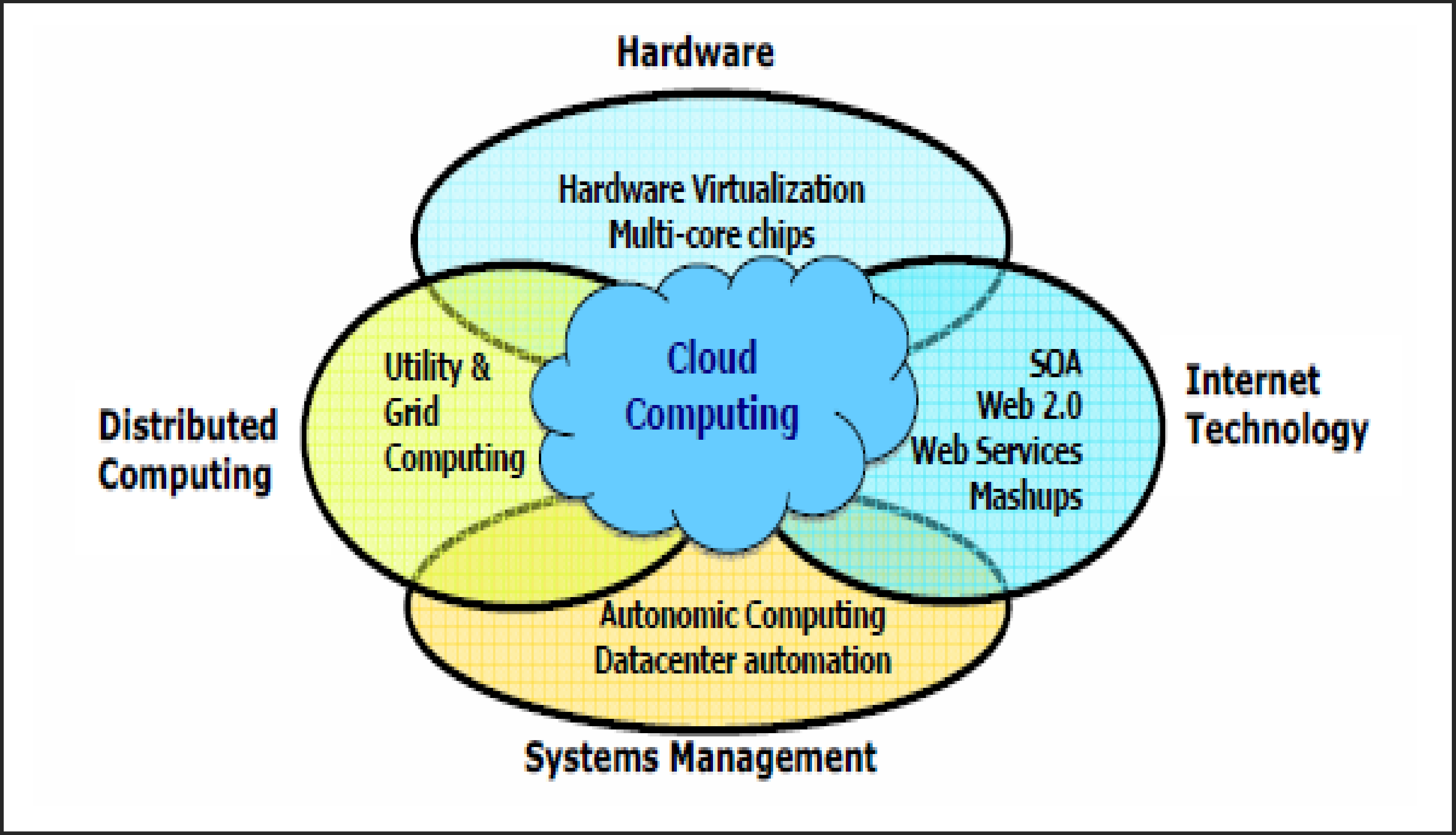
Distributed and Cloud Computing

K. Hwang, G. Fox and J. Dongarra

Chapter 1: Enabling Technologies and Distributed System Models

Data Deluge Enabling New Challenges

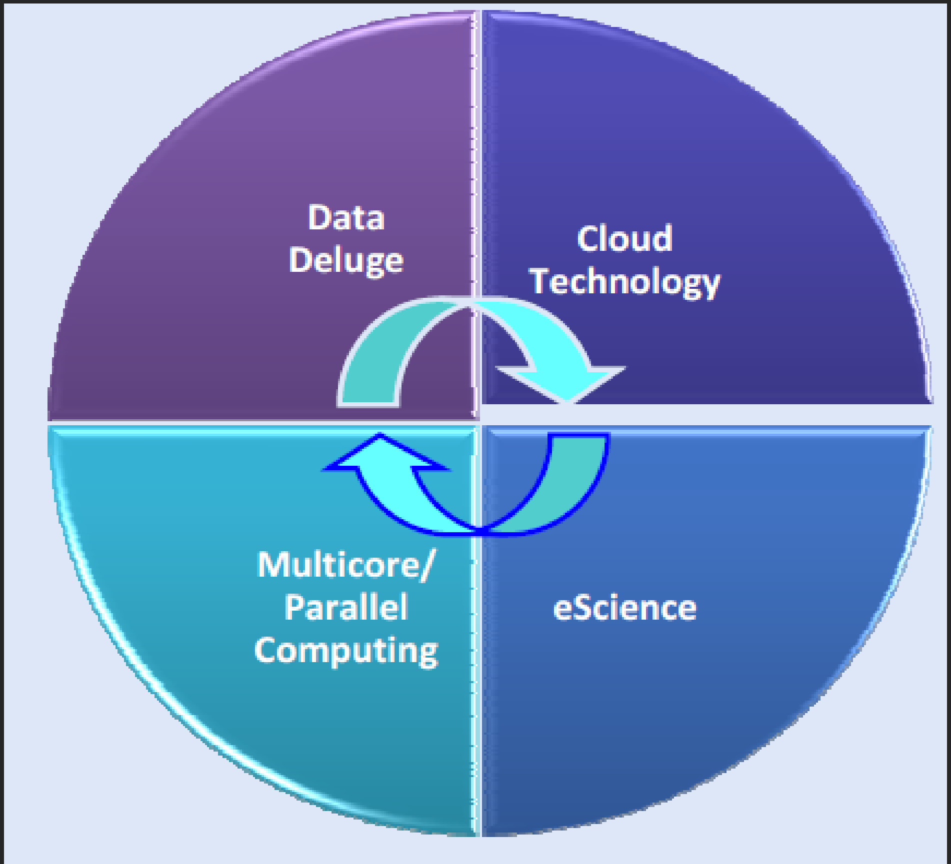


(Courtesy of Judy Qiu, Indiana University, 2011)

From Desktop/HPC/Grids to Internet Clouds in 30 Years

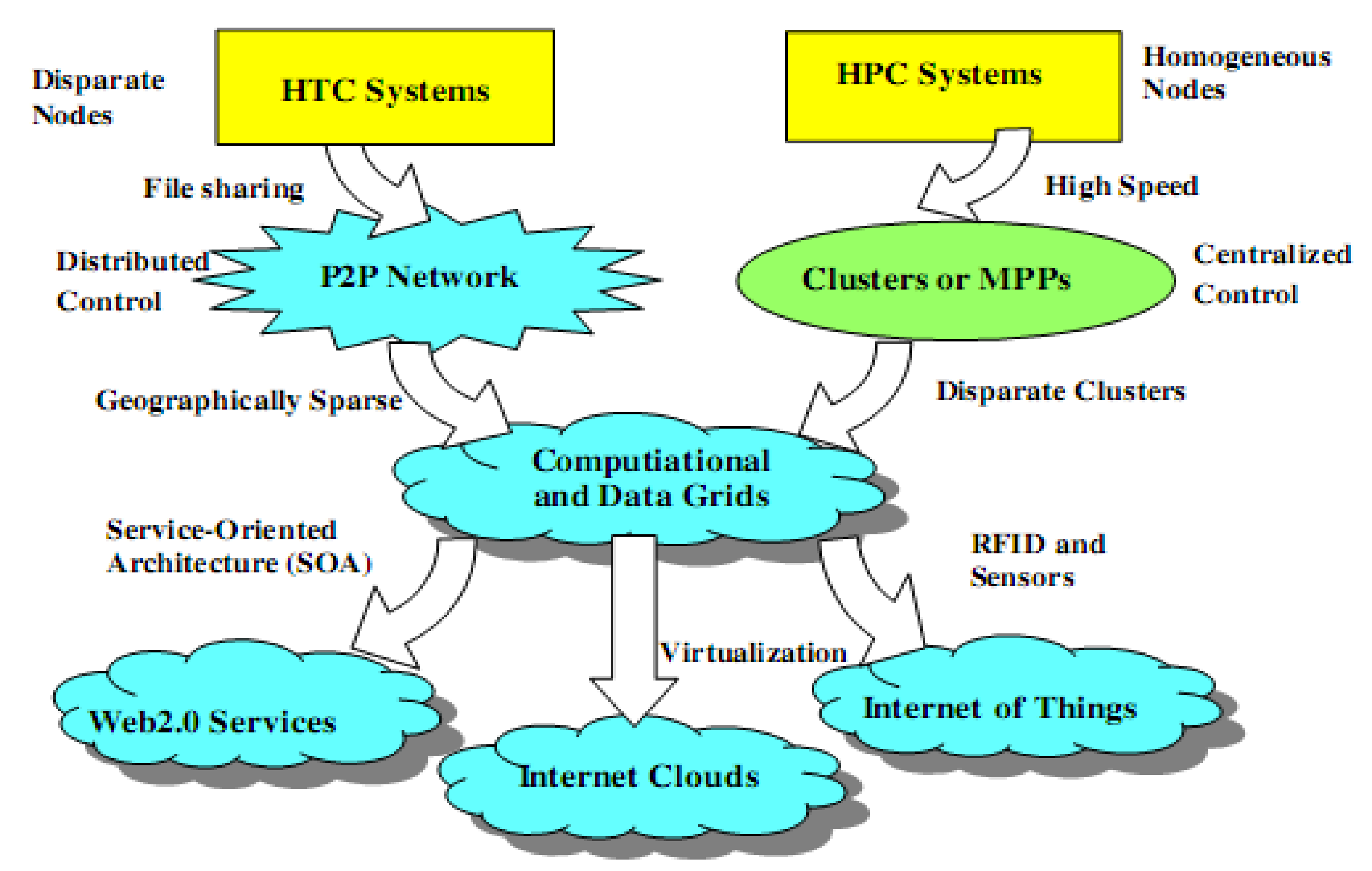
- HPC moving from centralized supercomputers to geographically distributed desktops, desksides, clusters, and grids to clouds over last 30 years
- R/D efforts on HPC, clusters, Grids, P2P, and virtual machines has laid the foundation of cloud computing that has been greatly advocated since 2007
- Location of computing infrastructure in areas with lower costs in hardware, software, datasets, space, and power requirements – moving from desktop computing to datacenter-based clouds

Interactions among 4 technical challenges : Data Deluge, Cloud Technology, eScience, and Multicore/Parallel Computing



(Courtesy of Judy Qiu, Indiana University, 2011)

Clouds and Internet of Things



HPC: High-Performance Computing

HTC: High-Throughput Computing

P2P: Peer to Peer

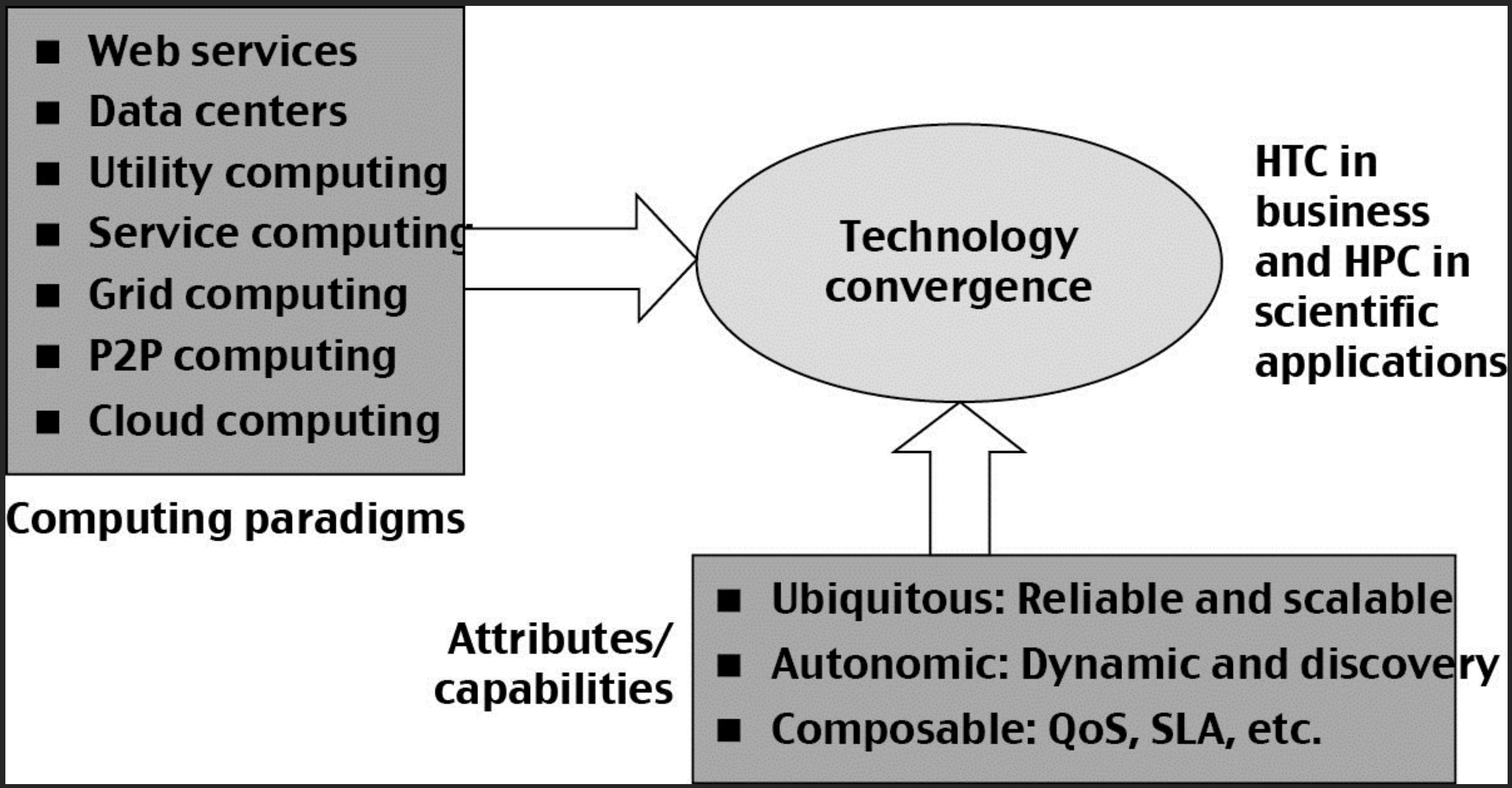
MPP: Massively Parallel Processors

Source: K. Hwang, G. Fox, and J. Dongarra, *Distributed and Cloud Computing,* Morgan Kaufmann, 2012.

Computing Paradigm Distinctions

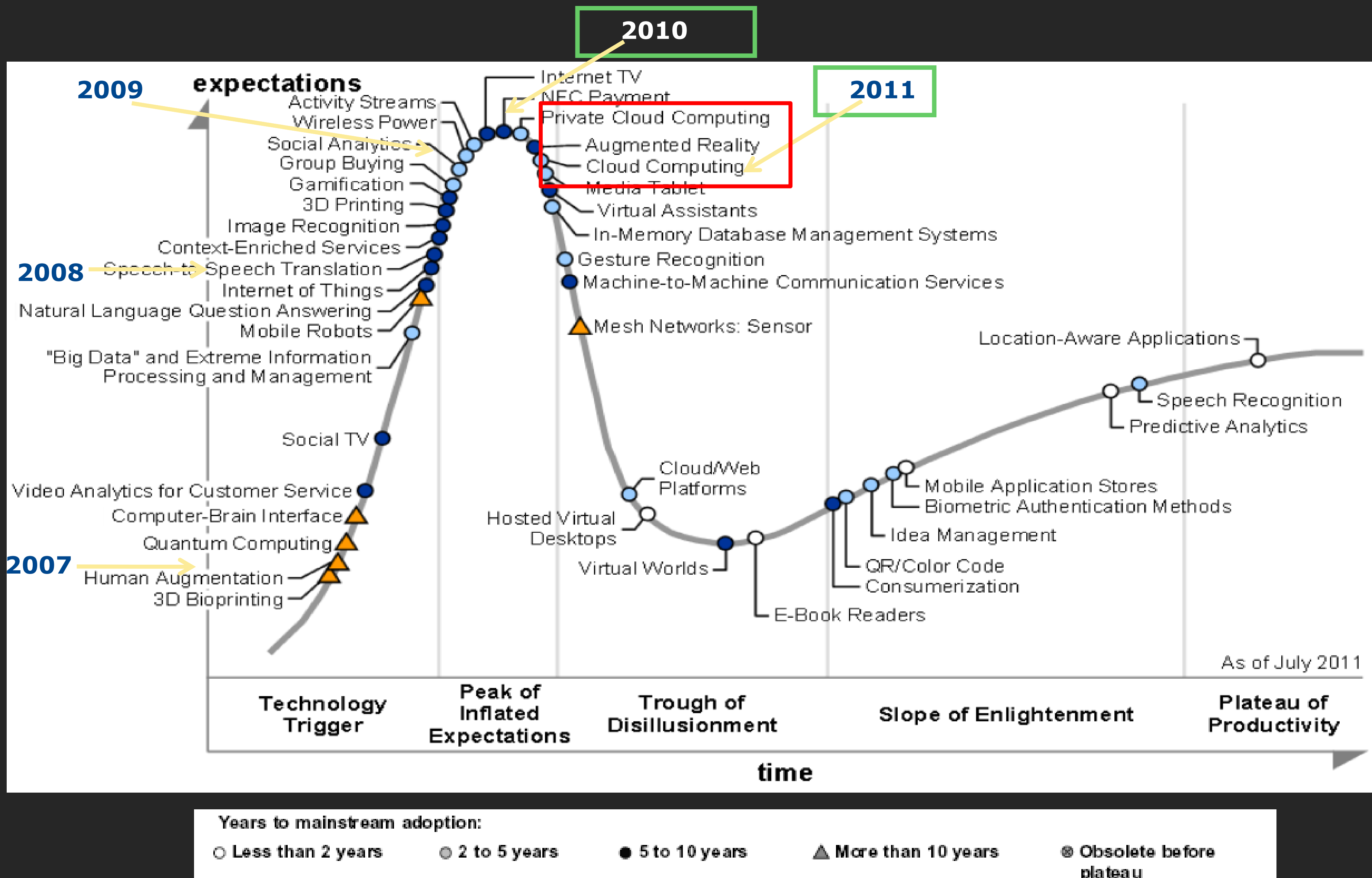
- Centralized Computing
  - All computer resources are centralized in one physical system.
- Parallel Computing
  - All processors are either tightly coupled with central shard memory or loosely coupled with distributed memory
- Distributed Computing
  - Field of CS/CE that studies distributed systems. A distributed system consists of multiple autonomous computers, each with its own private memory, communicating over a network.
- Cloud Computing
  - An Internet cloud of resources that may be either centralized or decentralized. The cloud apples to parallel or distributed computing or both. Clouds may be built from physical or virtualized resources.

Technology Convergence toward HPC for Science and HTC for Business: *Utility Computing*



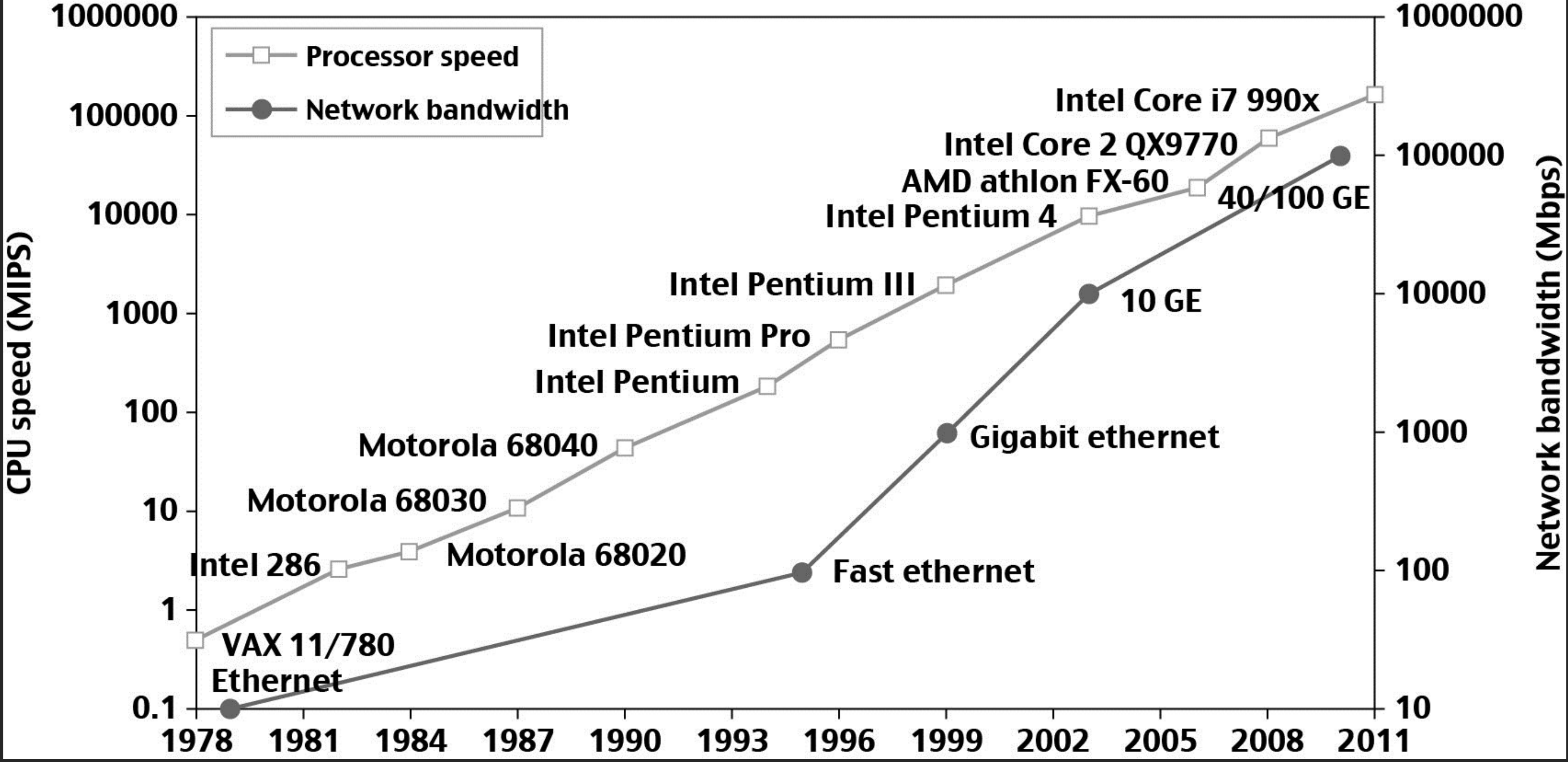
(Courtesy of Raj Buyya, University of Melbourne, 2011)

2011 Gartner "IT Hype Cycle" for Emerging Technologies

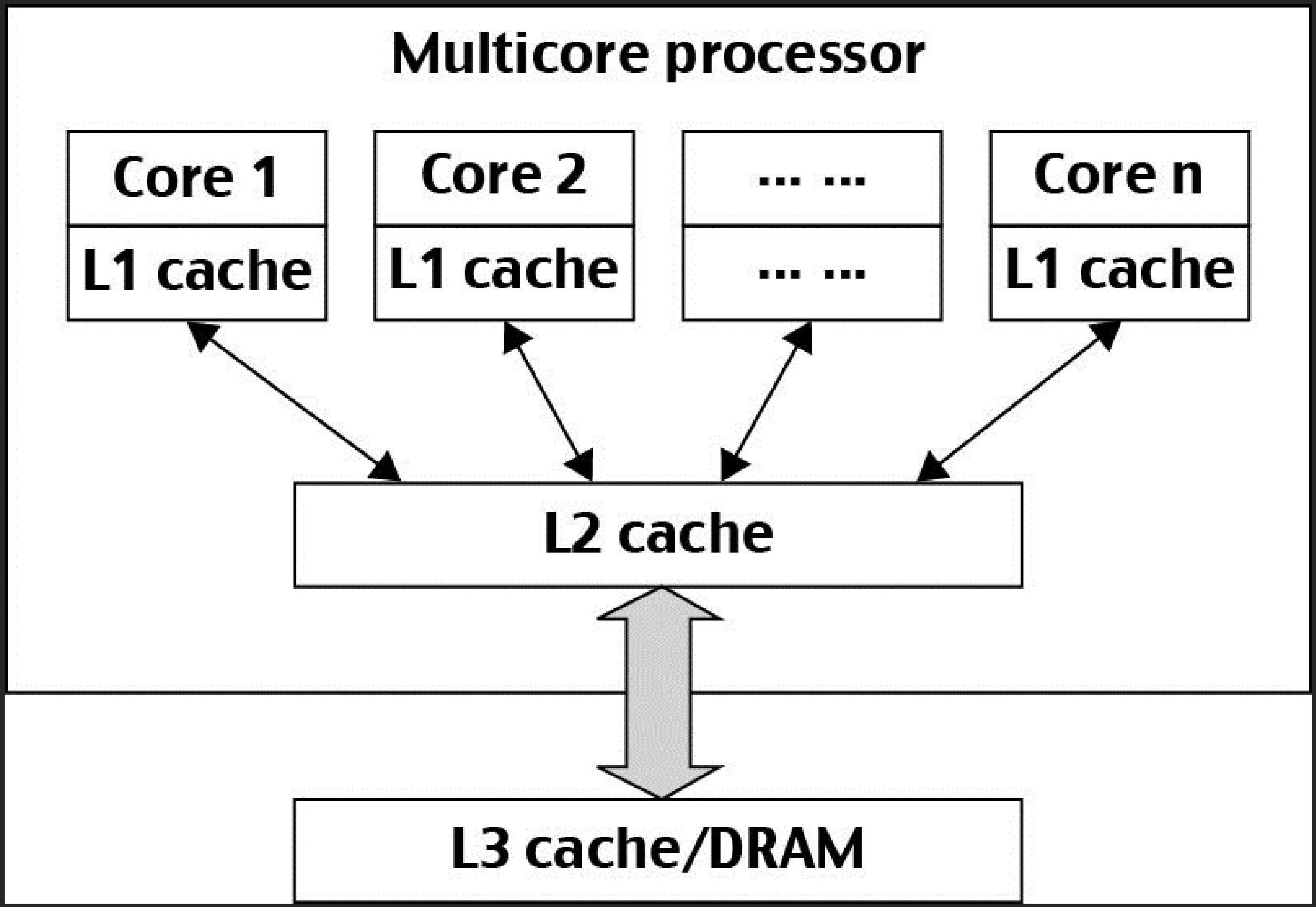


Technologies for Network-based Systems

33 year Improvement in Processor and Network Technologies



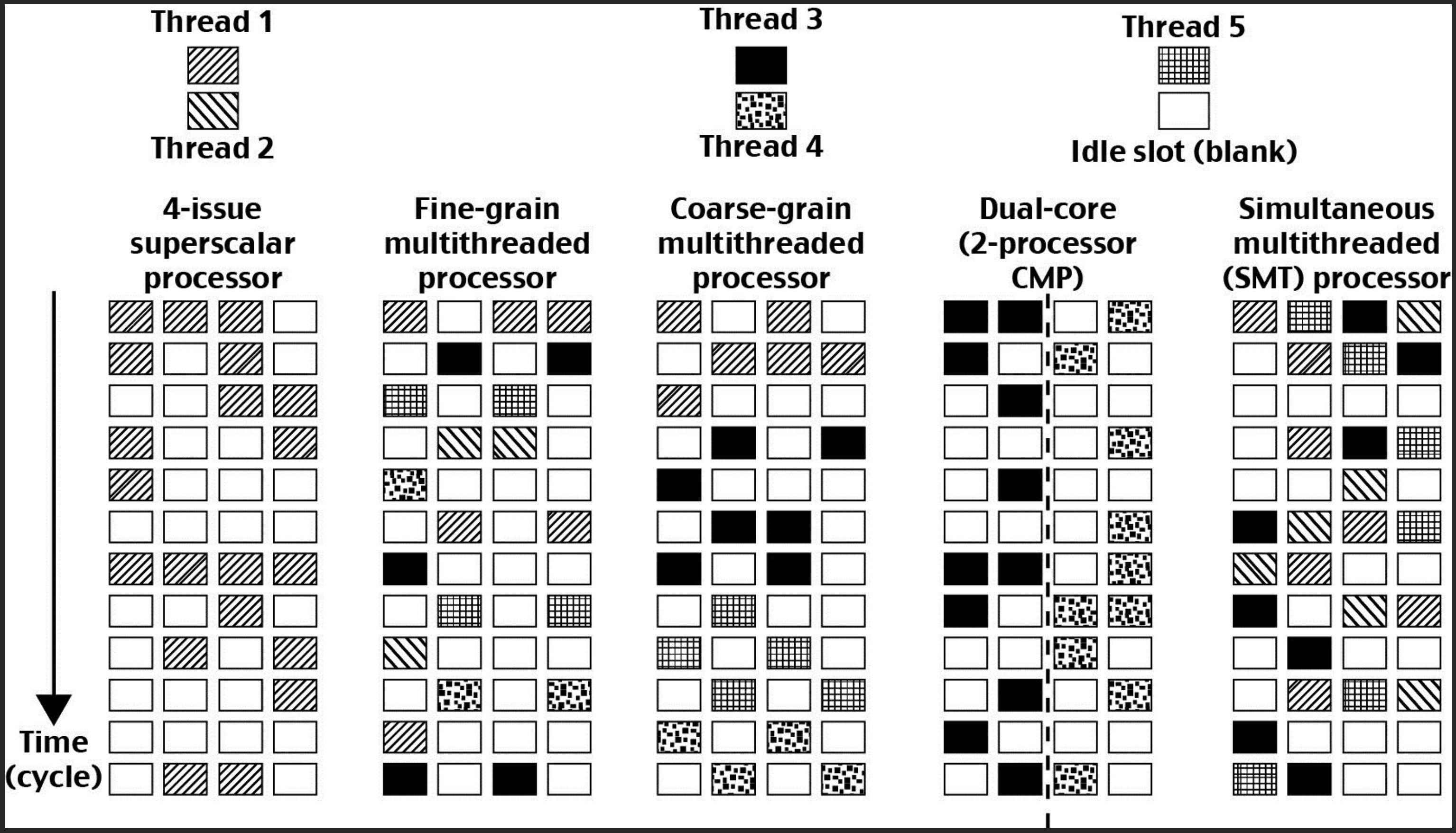
Modern Multi-core CPU Chip



Multi-threading Processors

- Four-issue superscalar (e.g. Sun Ultrasparc I)
  - Implements instruction level parallelism (ILP) within a single processor.
  - Executes more than one instruction during a clock cycle by sending multiple instructions to redundant functional units.
- Fine-grain multithreaded processor
  - Switch threads after each cycle
  - Interleave instruction execution
  - If one thread stalls, others are executed
- Coarse-grain multithreaded processor
  - Executes a single thread until it reaches certain situations
- Simultaneous multithread processor (SMT)
  - Instructions from more than one thread can execute in any given pipeline stage at a time.

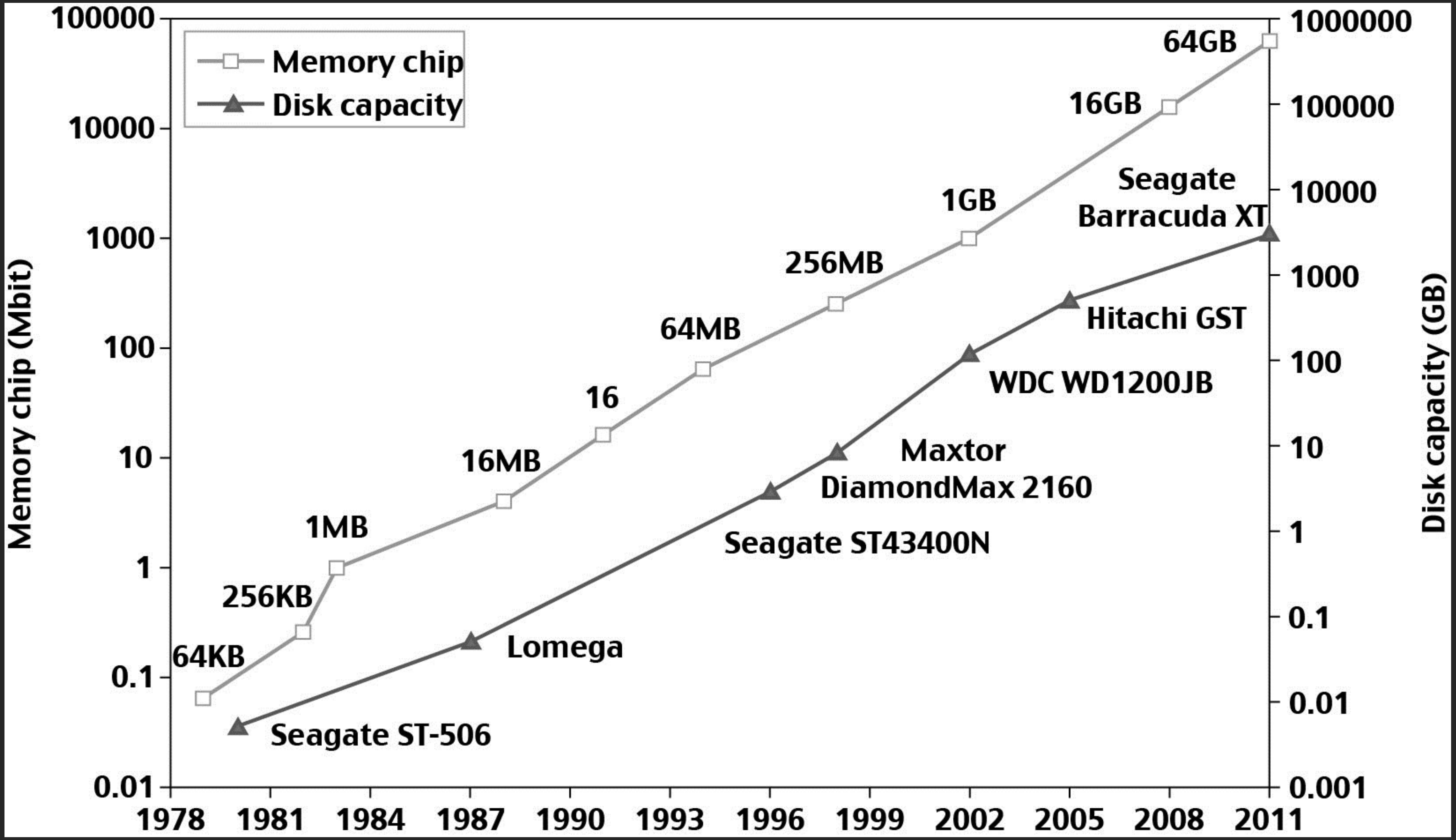
5 Micro-architectures of CPUs



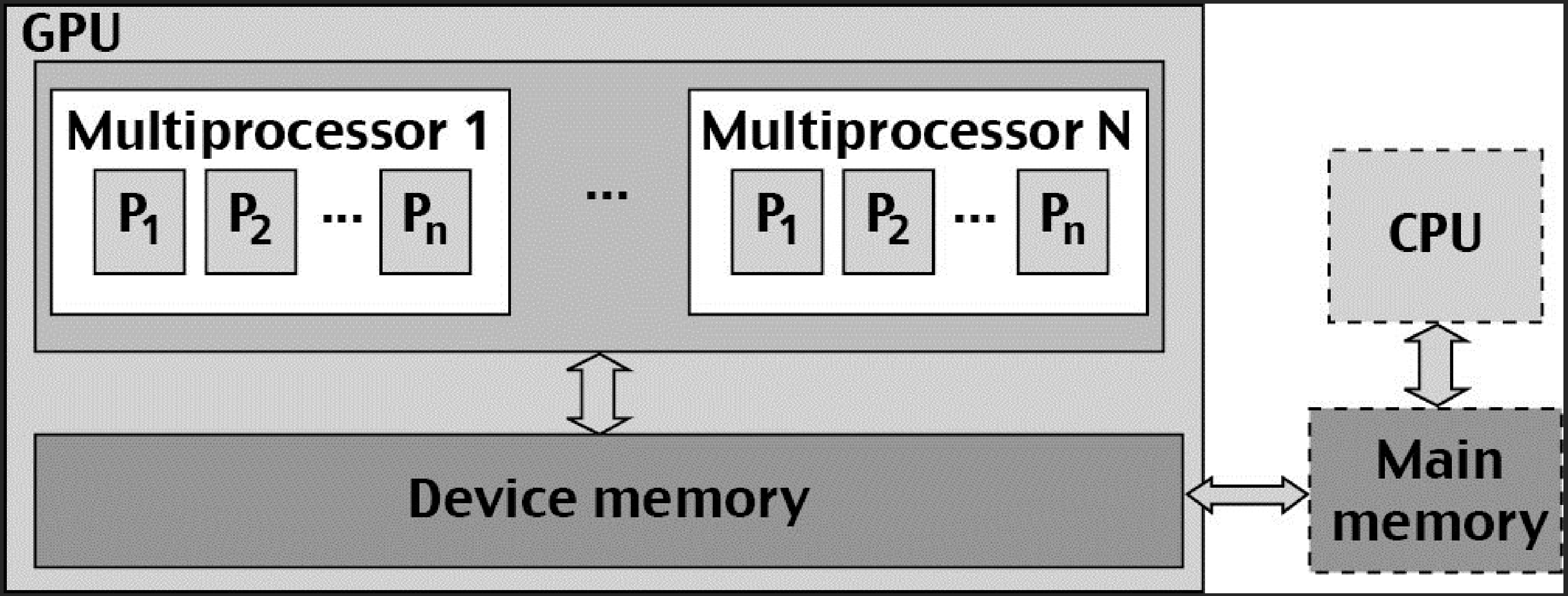
Each row represents the issue slots for a single execution cycle:

- A filled box indicates that the processor found an instruction to execute in that issue slot on that cycle;
- An empty box denotes an unused slot.

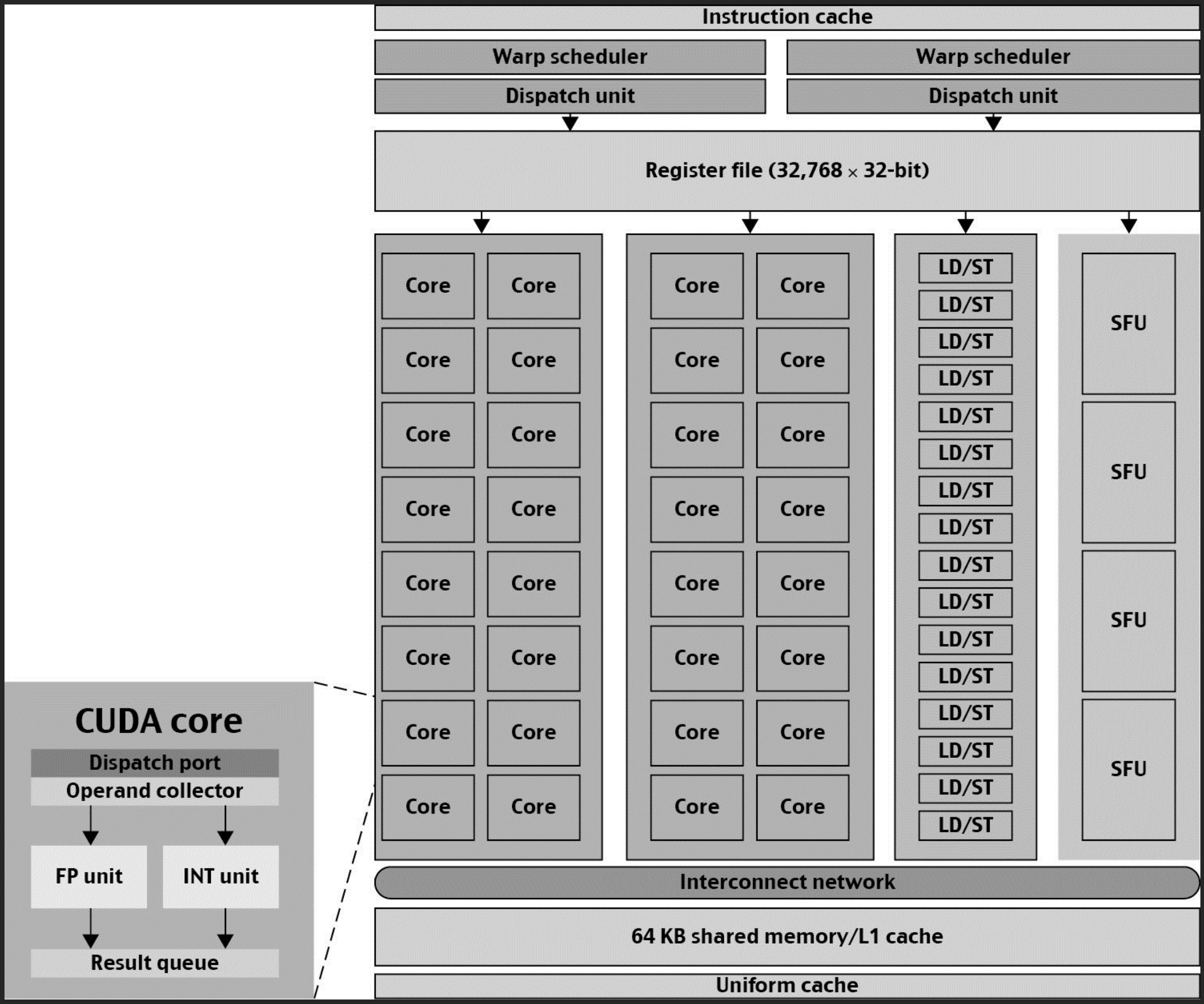
33 year Improvement in Memory and Disk Technologies



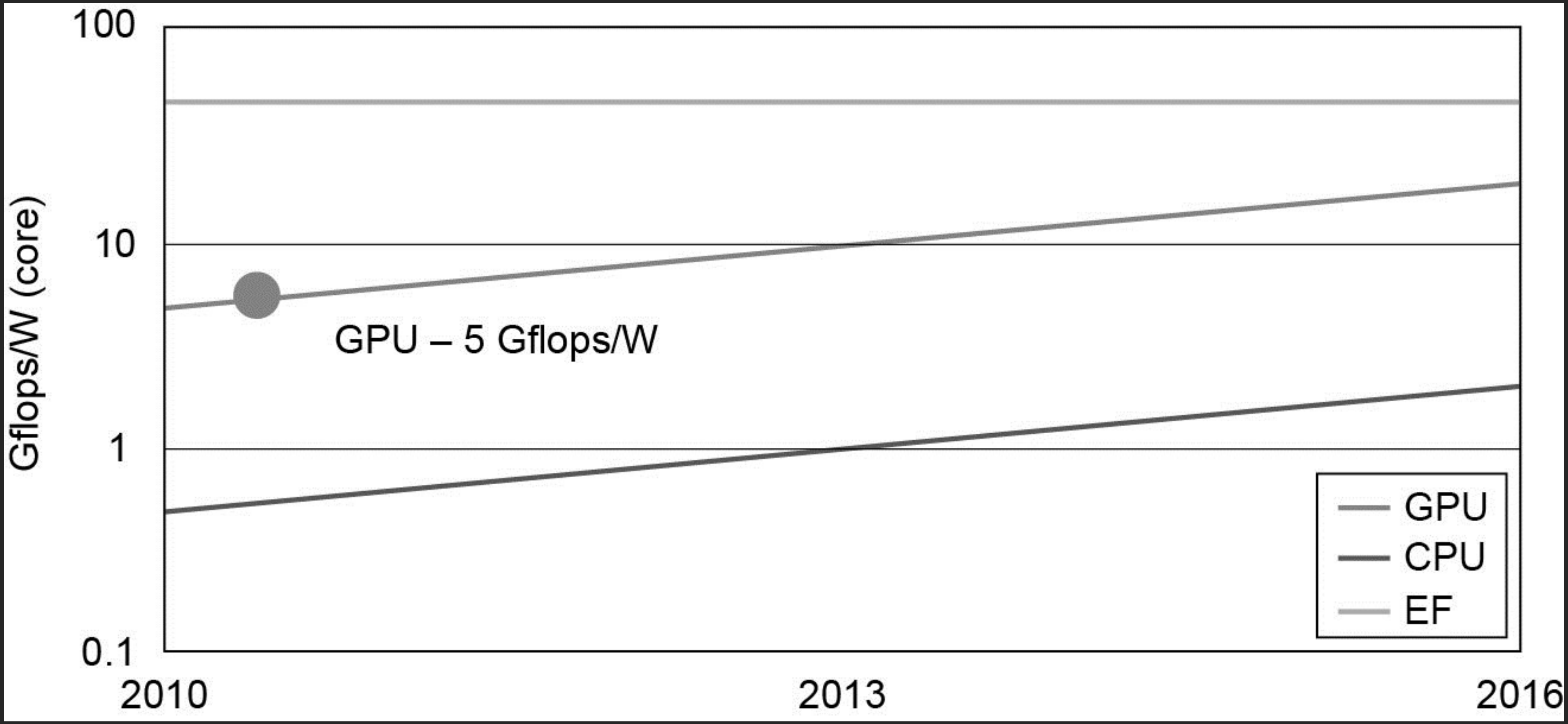
Architecture of A Many-Core Multiprocessor GPU interacting with a CPU Processor



NVIDIA Fermi GPU

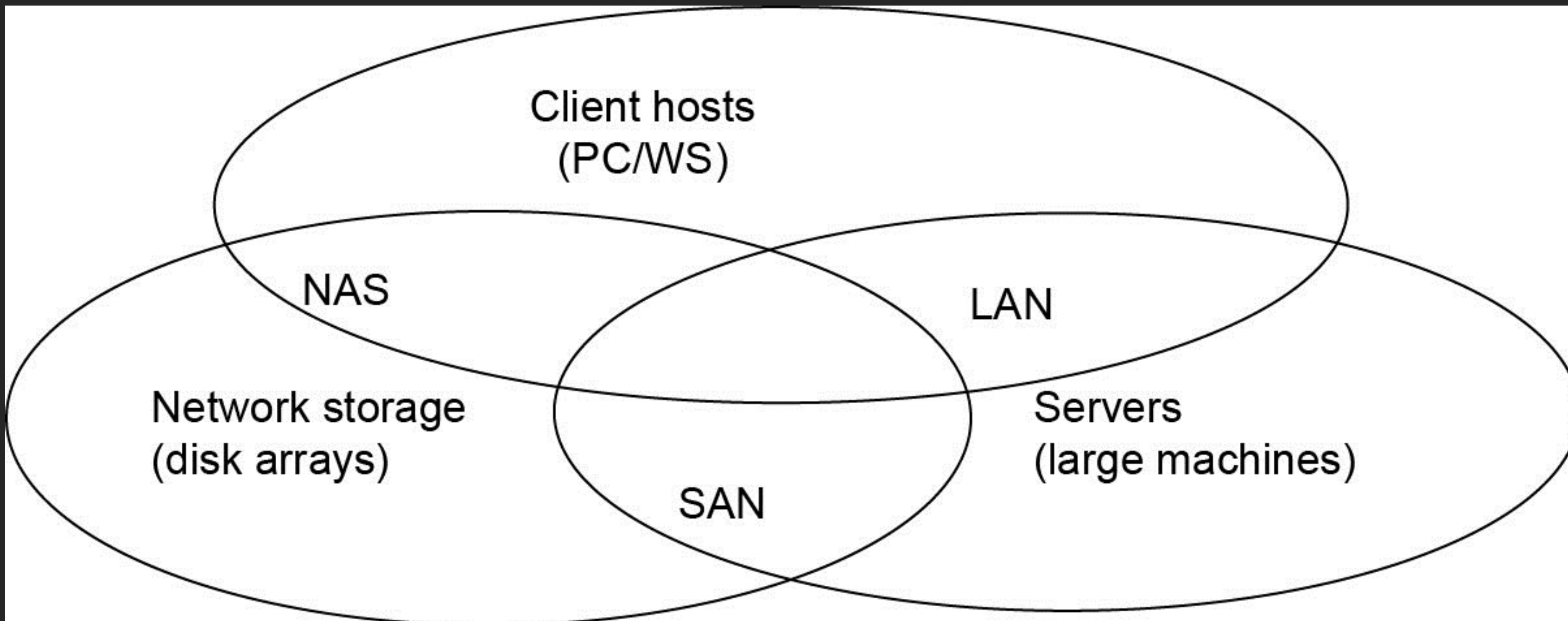


GPU Performance



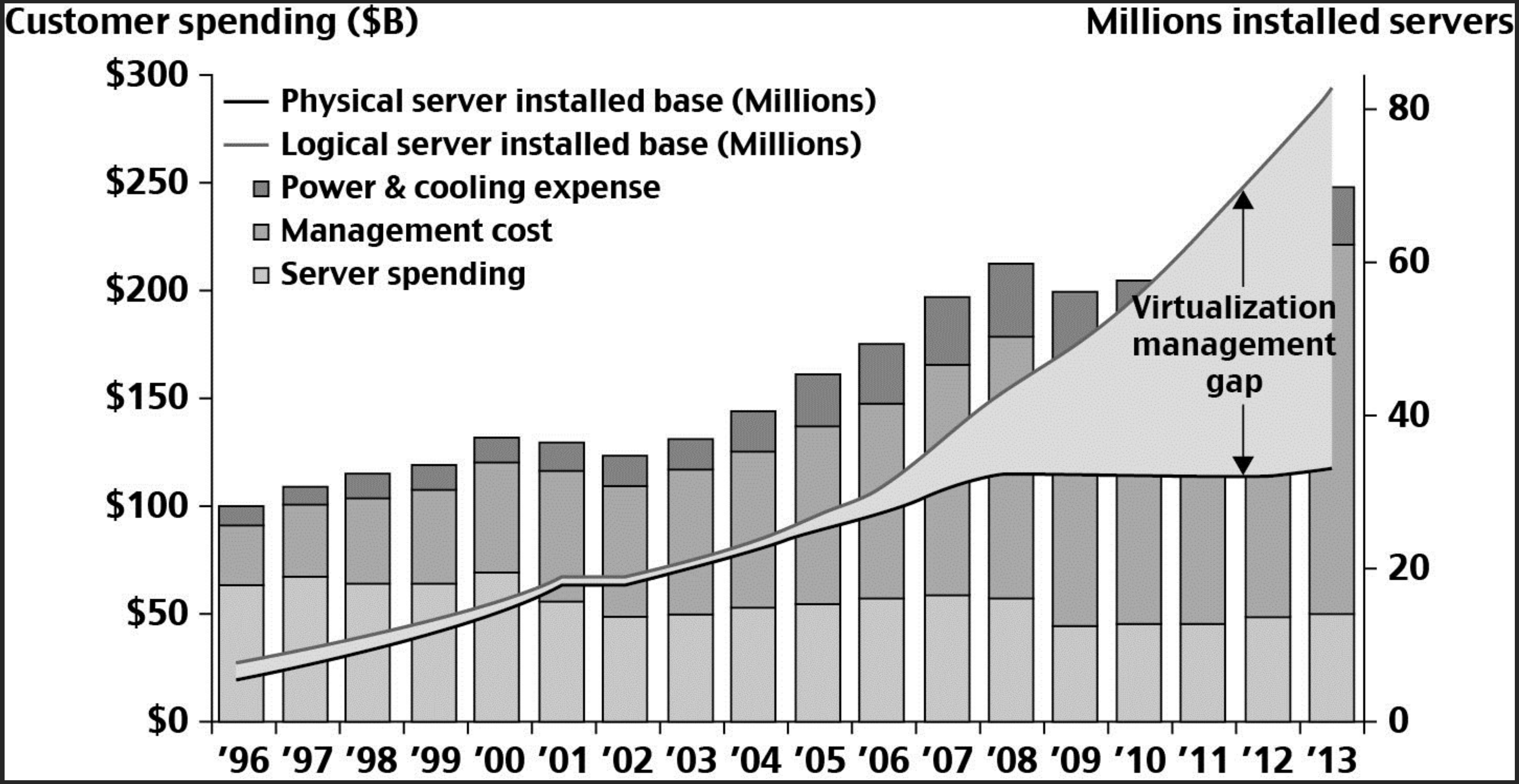
Bottom – CPU - 0.8 Gflops/W/Core (2011)
Middle – GPU - 5 Gflops/W/Core (2011)
Top - EF – Exascale computing (10^18 Flops)

Interconnection Networks



- SAN (storage area network) - connects servers with disk arrays
- LAN (local area network) – connects clients, hosts, and servers
- NAS (network attached storage) – connects clients with large storage systems

Datacenter and Server Cost Distribution

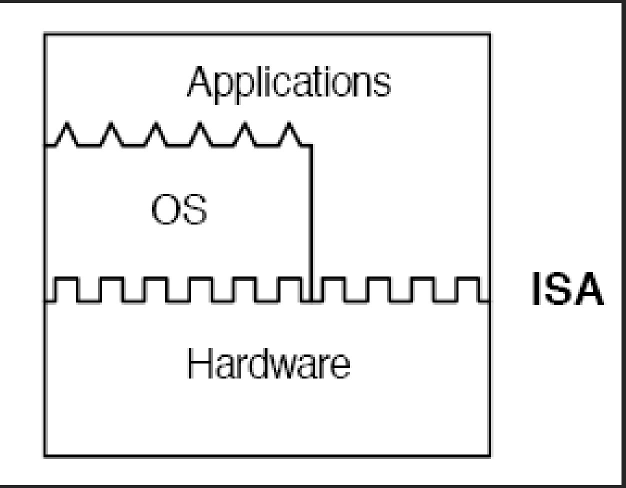


Virtual Machines

- Eliminate real machine constraint
  - Increases portability and flexibility
- Virtual machine adds software to a physical machine to give it the appearance of a different platform or multiple platforms.
- Benefits
  - Cross platform compatibility
  - Increase Security
  - Enhance Performance
  - Simplify software migration

Initial Hardware Model

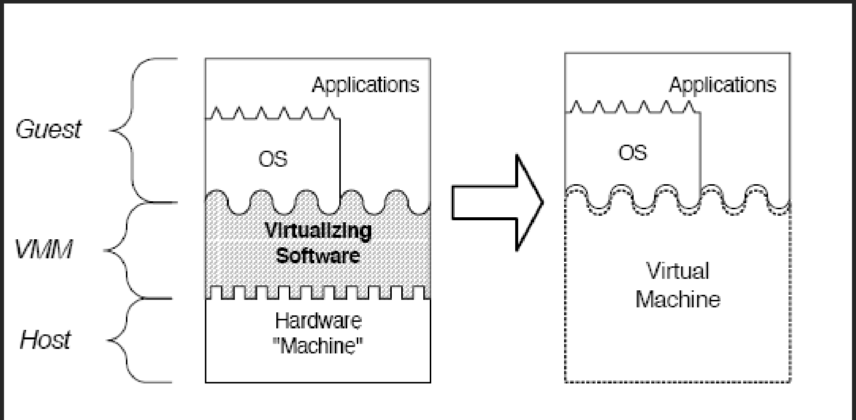
- All applications access hardware resources (i.e. memory, i/o) through system calls to operating system (privileged instructions)



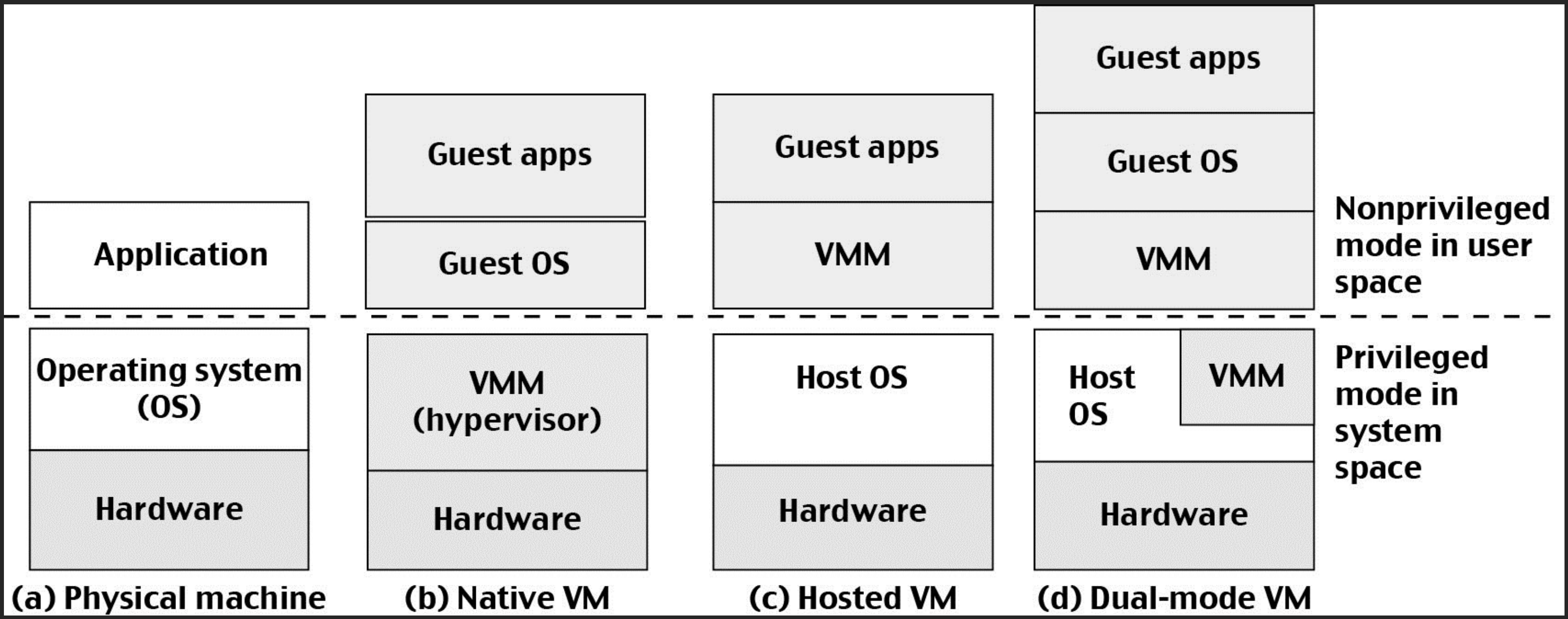
- Advantages
  - Design is decoupled (i.e. OS people can develop OS separate of Hardware people developing hardware)
  - Hardware and software can be upgraded without notifying the Application programs
- Disadvantage
  - Application compiled on one ISA will not run on another ISA..
    - Applications compiled for Mac use different operating system calls then application designed for windows.
  - ISA's must support old software
    - Can often be inhibiting in terms of performance
  - Since software is developed separately from hardware… Software is not necessarily optimized for hardware.

Virtual Machine Basics

- Virtual software placed between underlying machine and conventional software
  - Conventional software sees different ISA from the one supported by the hardware
- Virtualization process involves:
  - Mapping of virtual resources (registers and memory) to real hardware resources
  - Using real machine instructions to carry out the actions specified by the virtual machine instructions



Three VM Architectures

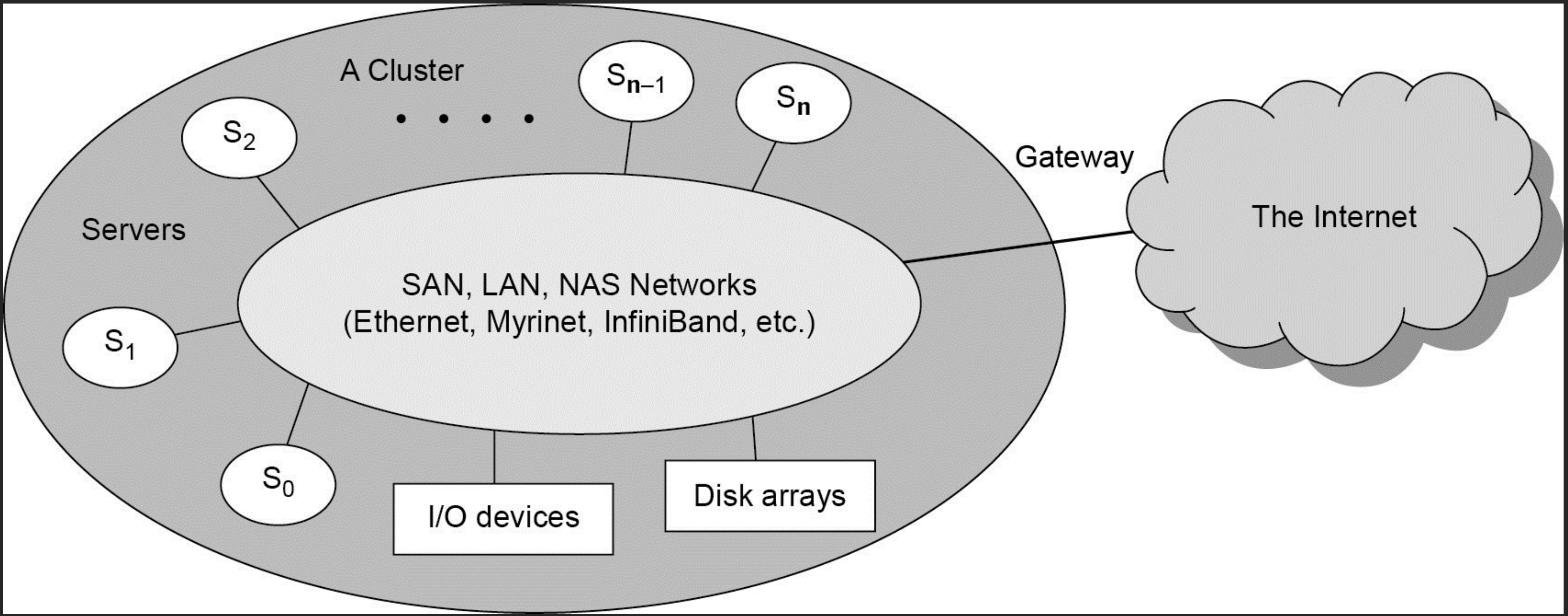


System Models for Distributed and Cloud Computing

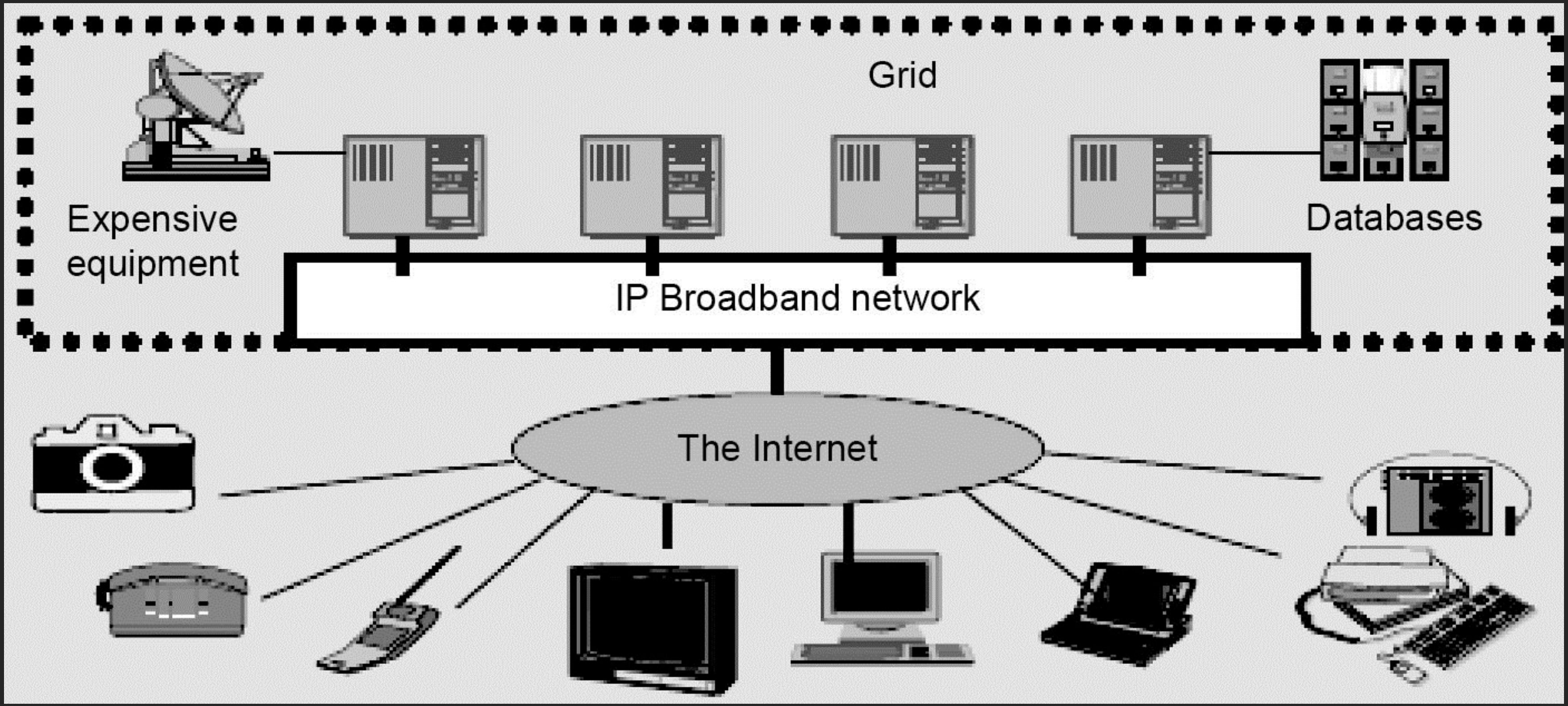
Table 1.2 Classification of Distributed Parallel Computing Systems

Functionality, Applications	Multicomputer Clusters [27, 33]	Peer-to-Peer Networks [40]	Data/Computational Grids [6, 42]	Cloud Platforms [1, 9, 12, 17, 29]
Architecture, Network Connectivity and Size	Network of compute nodes interconnected by SAN, LAN, or WAN, hierarchically	Flexible network of client machines logically connected by an overlay network	Heterogeneous clusters interconnected by high-speed network links over selected resource sites.	Virtualized cluster of servers over datacenters via service-level agreement
Control and Resources Management	Homogeneous nodes with distributed control, running Unix or Linux	Autonomous client nodes, free in and out, with distributed self-organization	Centralized control, server oriented with authenticated security, and static resources	Dynamic resource provisioning of servers, storage, and networks over massive datasets
Applications and network-centric services	High-performance computing, search engines, and web services, etc.	Most appealing to business file sharing, content delivery, and social networking	Distributed super-computing, global problem solving, and datacenter services	Upgraded web search, utility computing, and outsourced computing services
Representative Operational Systems	Google search engine, SunBlade, IBM Road Runner, Cray XT4, etc.	Gnutella, eMule, BitTorrent, Napster, KaZaA, Skype, JXTA, and .NET	TeraGrid, GriPhyN, UK EGEE, D-Grid, ChinaGrid, etc.	Google App Engine, IBM Bluecloud, Amazon Web Service(AWS), and Microsoft Azure,

A Typical Cluster Architecture



Computational or Data Grid



A Typical Computational Grid

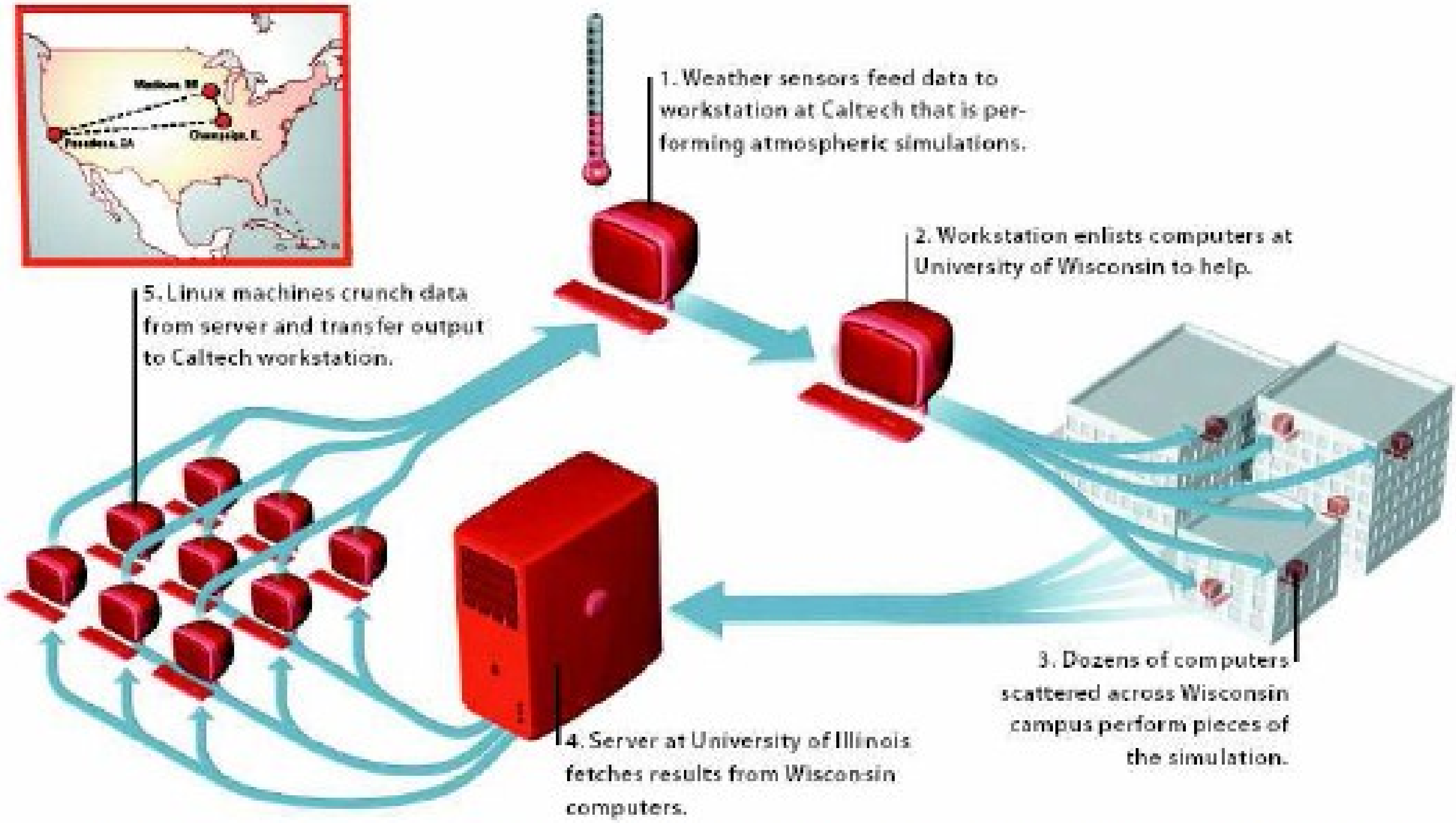
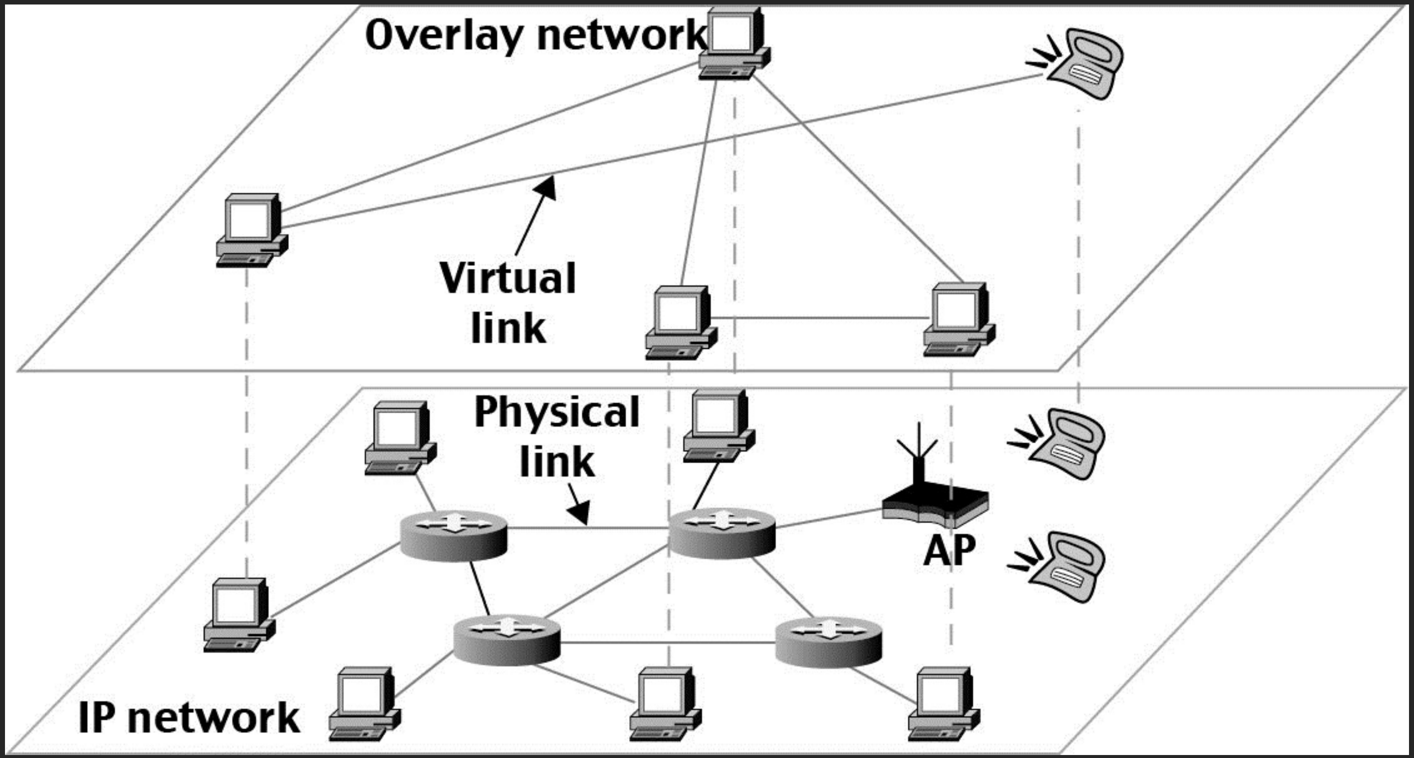


Figure 1.17 An example computational Grid built over specialized computers at three resource sites at Wisconsin, Caltech, and Illinois. (Courtesy of Michel Waldrop, "Grid Computing", IEEE *Computer Magazine*, 2000. [42])

Peer-to-Peer (P2P) Network

- A distributed system architecture
- Each computer in the network can act as a client or server for other netwpork computers.
- No centralized control
- Typically many nodes, but unreliable and heterogeneous
- Nodes are symmetric in function
- Take advantage of distributed, shared resources (bandwidth, CPU, storage) on peer-nodes
- Fault-tolerant, self-organizing
- Operate in dynamic environment, frequent join and leave is the norm

Peer-to-Peer (P2P) Network



Overlay network - computer network built on top of another network.

- Nodes in the overlay can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links, in the underlying network.
- For example, distributed systems such as cloud computing, peer-to-peer networks, and client-server applications are overlay networks because their nodes run on top of the Internet.

Table 1.5 Major Categories of P2P Network Families [42]

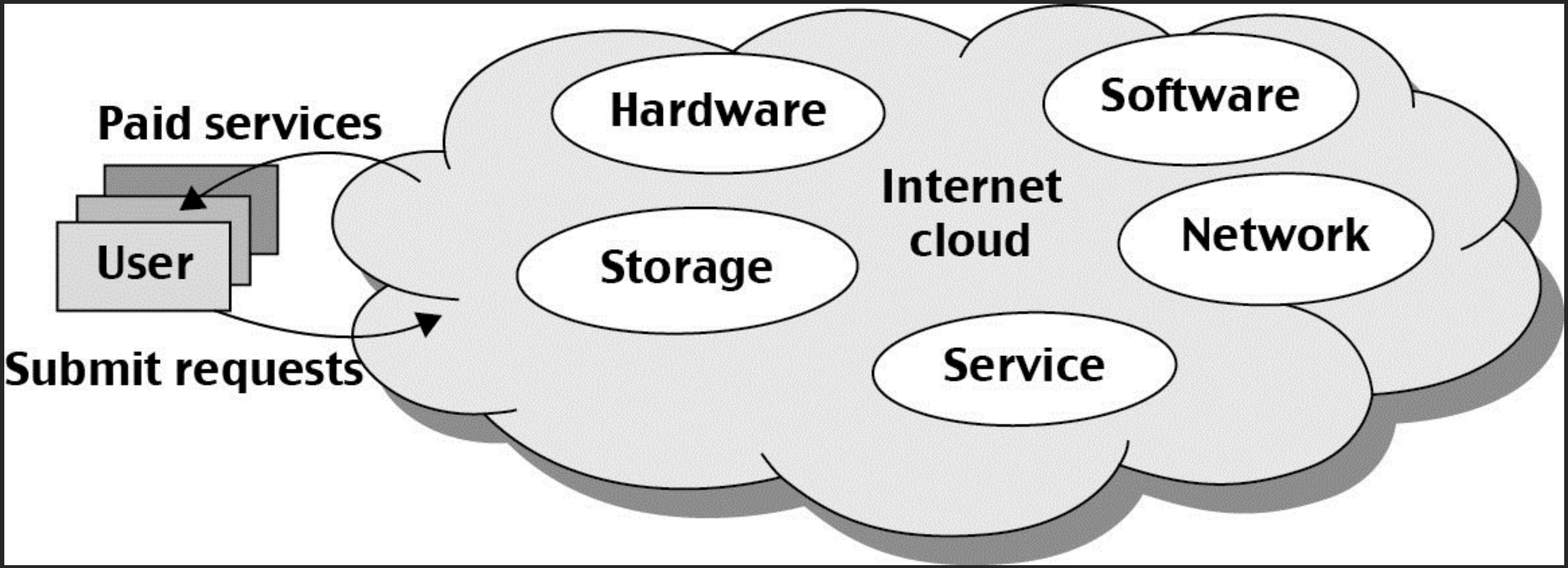
System Features	Distributed File Sharing	Collaborative Platform	Distributed P2P Computing	P2P Platform
Attractive Applications	Content distribution of MP3 music, video, open software, etc.	Instant messaging, collaborative design and gaming	Scientific exploration and social networking	Open networks for public resources
Operational Problems	Loose security and serious online copyright violations	Lack of trust, disturbed by spam, privacy, and peer collusion	Security holes, selfish partners, and peer collusion	Lack of standards or protection protocols
Example Systems	Gnutella, Napster, eMule, BitTorrent, Aimster, KaZaA, etc.	ICQ, AIM, Groove, Magi, Multiplayer Games, Skype, etc.	SETI@home, Geonome@home, etc.	JXTA, .NET, FightingAid@home, etc.

The Cloud

- Historical roots in today's Internet apps
  - Search, email, social networks
  - File storage (Live Mesh, Mobile Me, Flicker, …)
- A cloud infrastructure provides a framework to manage scalable, reliable, on-demand access to applications
- A cloud is the "invisible" backend to many of our mobile applications
- A model of computation and data storage based on "pay as you go" access to "unlimited" remote data center capabilities

Basic Concept of Internet Clouds

- Cloud computing is the use of computing resources (hardware and software) that are delivered as a service over a network (typically the Internet).
- The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams.
- Cloud computing entrusts remote services with a user's data, software and computation.



The Next Revolution in IT
Cloud Computing

- **Classical Computing**
  - Buy & Own
    - Hardware, System Software, Applications often to meet peak needs.
  - Install, Configure, Test, Verify, Evaluate
  - Manage
  - ..
  - Finally, use it
  - $$$$....$(High CapEx)

Every 18 months?

- **Cloud Computing**
  - Subscribe
  - Use

  - $ - pay for what you use, based on QoS

(Courtesy of Raj Buyya, 2012)

Cloud Service Models (1)

Infrastructure as a service (IaaS)

- Most basic cloud service model
- Cloud providers offer computers, as physical or more often as virtual machines, and other resources.
- Virtual machines are run as guests by a hypervisor, such as Xen or KVM.
- Cloud users deploy their applications by then installing operating system images on the machines as well as their application software.
- Cloud providers typically bill IaaS services on a utility computing basis, that is, cost will reflect the amount of resources allocated and consumed.
- Examples of IaaS include: Amazon CloudFormation (and underlying services such as Amazon EC2), Rackspace Cloud, Terremark, and Google Compute Engine.

Cloud Service Models (2)

Platform as a service (PaaS)

- Cloud providers deliver a computing platform typically including operating system, programming language execution environment, database, and web server.
- Application developers develop and run their software on a cloud platform without the cost and complexity of buying and managing the underlying hardware and software layers.
- Examples of PaaS include: Amazon Elastic Beanstalk, Cloud Foundry, Heroku, Force.com, EngineYard, Mendix, Google App Engine, Microsoft Azure and OrangeScape.

Cloud Service Models (3)

Software as a service (SaaS)

- Cloud providers install and operate application software in the cloud and cloud users access the software from cloud clients.
- The pricing model for SaaS applications is typically a monthly or yearly flat fee per user, so price is scalable and adjustable if users are added or removed at any point.
- Examples of SaaS include: Google Apps, innkeypos, Quickbooks Online, Limelight Video Platform, Salesforce.com, and Microsoft Office 365.

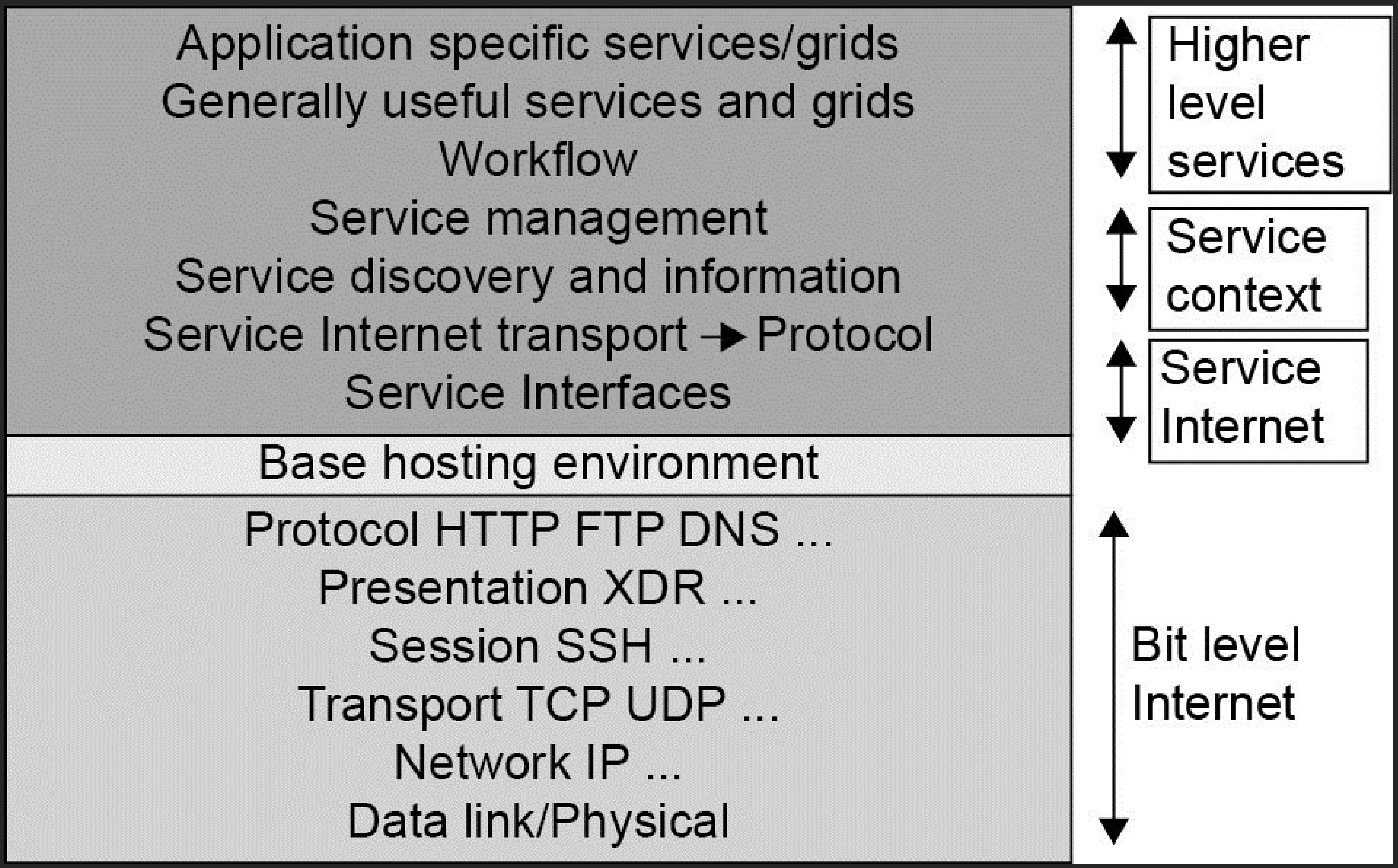
Service-oriented architecture (SOA)

- SOA is an evolution of distributed computing based on the request/reply design paradigm for synchronous and asynchronous applications.
- An application's business logic or individual functions are modularized and presented as services for consumer/client applications.
- Key to these services - their loosely coupled nature;
  - i.e., the service interface is independent of the implementation.
- Application developers or system integrators can build applications by composing one or more services without knowing the services' underlying implementations.
  - For example, a service can be implemented either in .Net or J2EE, and the application consuming the service can be on a different platform or language.

SOA key characteristics:

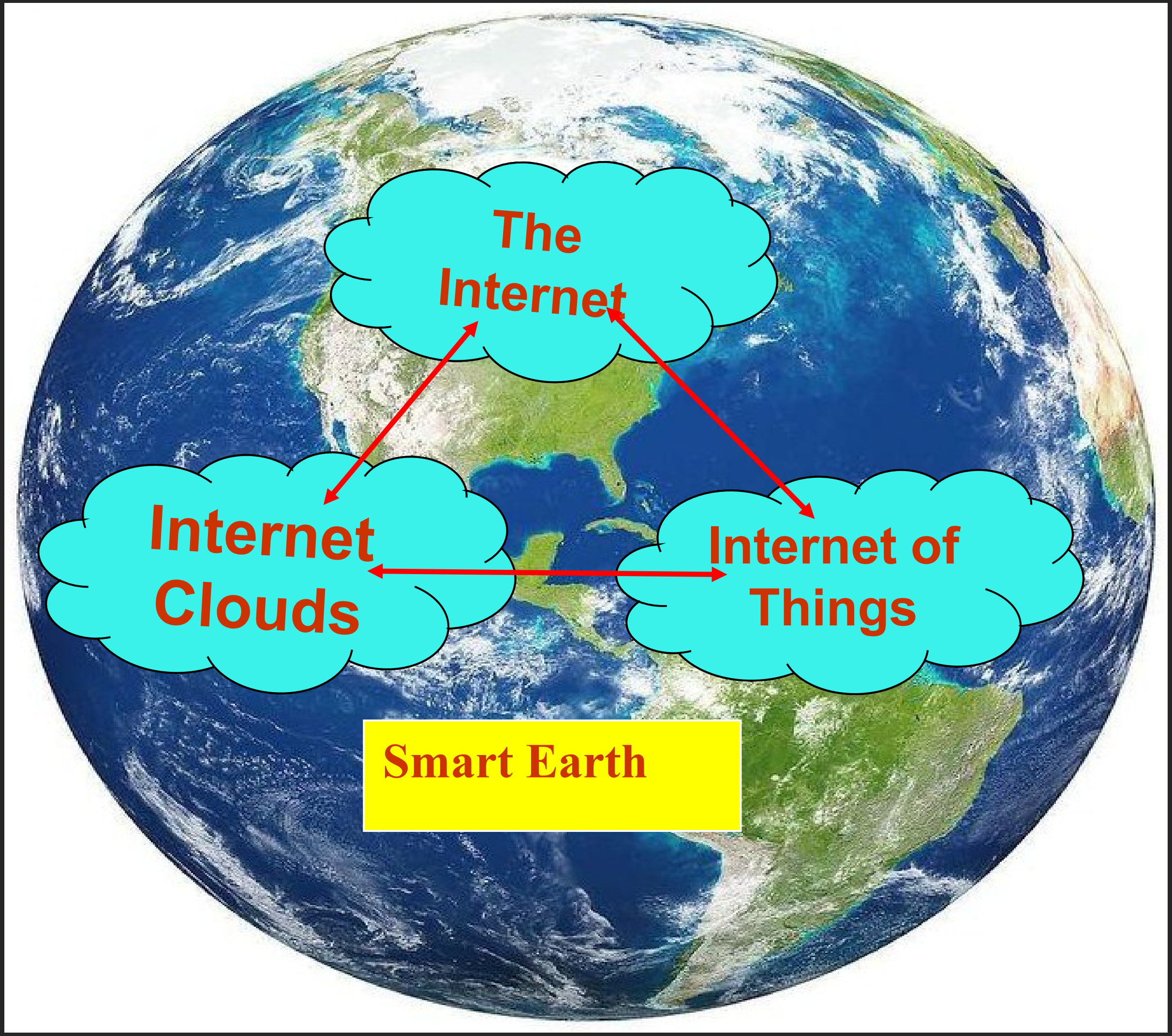
- SOA services have self-describing interfaces in platform-independent XML documents.
  - Web Services Description Language (WSDL) is the standard used to describe the services.
- SOA services communicate with messages formally defined via XML Schema (also called XSD).
  - Communication among consumers and providers or services typically happens in heterogeneous environments, with little or no knowledge about the provider.
  - Messages between services can be viewed as key business documents processed in an enterprise.
- SOA services are maintained in the enterprise by a registry that acts as a directory listing.
  - Applications can look up the services in the registry and invoke the service.
  - Universal Description, Definition, and Integration (UDDI) is the standard used for service registry.
- Each SOA service has a quality of service (QoS) associated with it.
  - Some of the key QoS elements are security requirements, such as authentication and authorization, reliable messaging, and policies regarding who can invoke services.

Layered Architecture for Web Services



Cloud Computing Challenges:
Dealing with too many issues (Courtesy of R. Buyya)

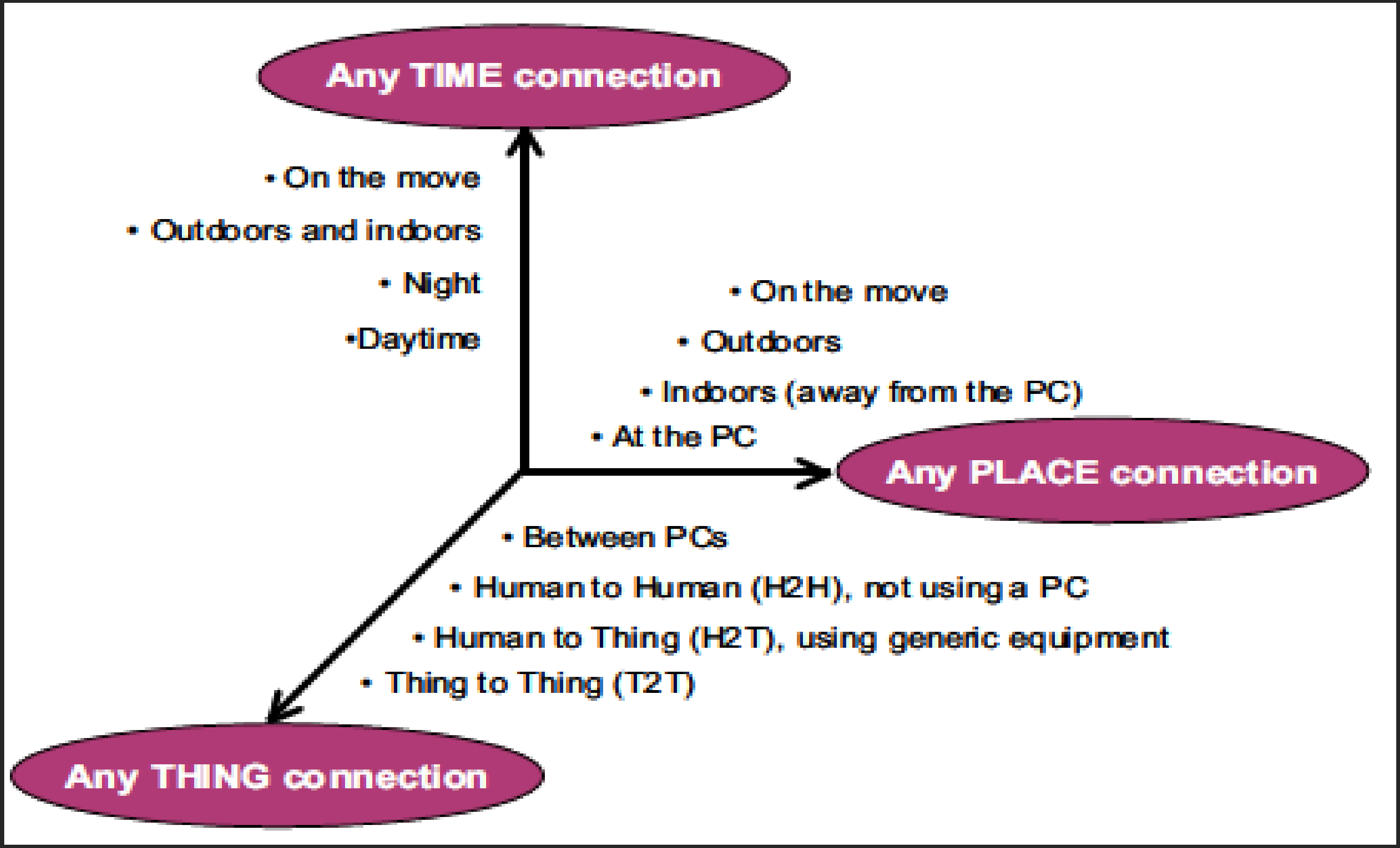
The Internet of Things (IoT)



Smart Earth:

An IBM Dream

Opportunities of IoT in 3 Dimensions

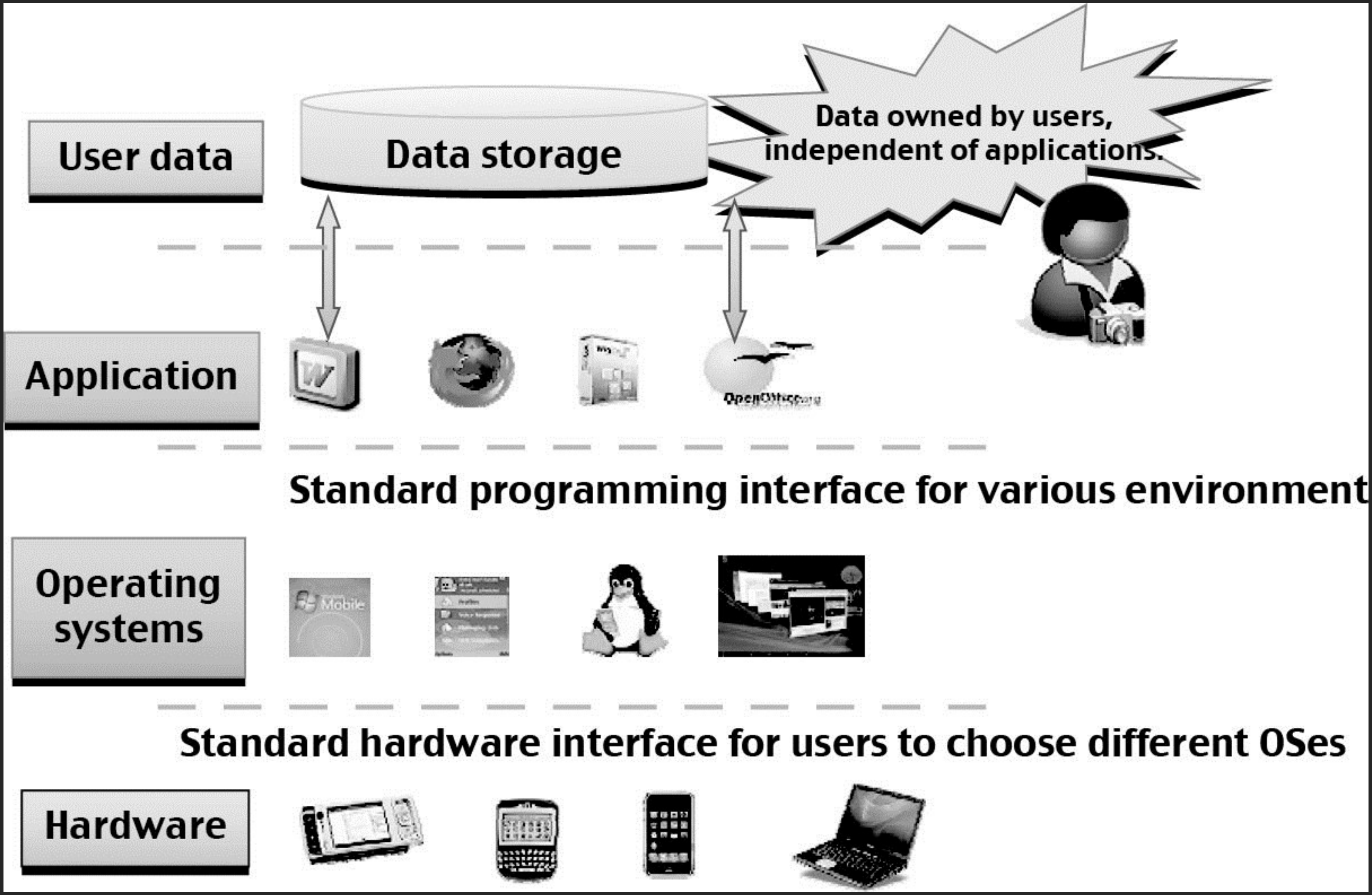


(courtesy of Wikipedia, 2010)

Table 1.6 Feature Comparison of Three Distributed Operating Systems

Distributed OS Functionality	AMOEBA developed at Vrije University [46]	DCE as OSF/1 by Open Software Foundation [7]	MOSIX for Linux Clusters at Hebrew University [3]
History and Current System Status	Written in C and tested in the European community; version 5.2 released in 1995	Built as a user extension on top of UNIX, VMS, Windows, OS/2, etc.	Developed since 1977, now called MOSIX2 used in HPC Linux and GPU clusters
Distributed OS Architecture	Microkernel-based and location-transparent, uses many servers to handle files, directory, replication, run, boot, and TCP/IP services	Middleware OS providing a platform for running distributed applications; The system supports RPC, security, and threads	A distributed OS with resource discovery, process migration, runtime support, load balancing, flood control, configuration, etc.
OS Kernel, Middleware, and Virtualization Support	A special microkernel that handles low-level process, memory, I/O, and communication functions	DCE packages handle file,time, directory, security services, RPC, and authentication at middleware or user space	MOSIX2 runs with Linux 2.6; extensions for use in multiple clusters and clouds with provisioned VMs
Communication Mechanisms	Uses a network-layer FLIP protocol and RPC to implement point-to-point and group communication	RPC supports authenticated communication and other security services in user programs	Using PVM, MPI in collective communications, priority process control, and queuing services

Transparent Cloud Computing Environment



Separates user data, application, OS, and space – good for cloud computing.

Parallel and Distributed Programming

Table 1.7 Parallel and Distributed Programming Models and Tool Sets

Model	Description	Features
MPI	A library of subprograms that can be called from C or FORTRAN to write parallel programs running on distributed computer systems [6,28,42]	Specify synchronous or asynchronous point-to-point and collective communication commands and I/O operations in user programs for message-passing execution
MapReduce	A Web programming model for scalable data processing on large clusters over large data sets, or in Web search operations [16]	*Map* function generates a set of intermediate key/value pairs; *Reduce* function merges all intermediate values with the same key
Hadoop	A software library to write and run large user applications on vast data sets in business applications (http://hadoop.apache.org/core)	A scalable, economical, efficient, and reliable tool for providing users with easy access of commercial clusters

Grid Standards and Middleware :

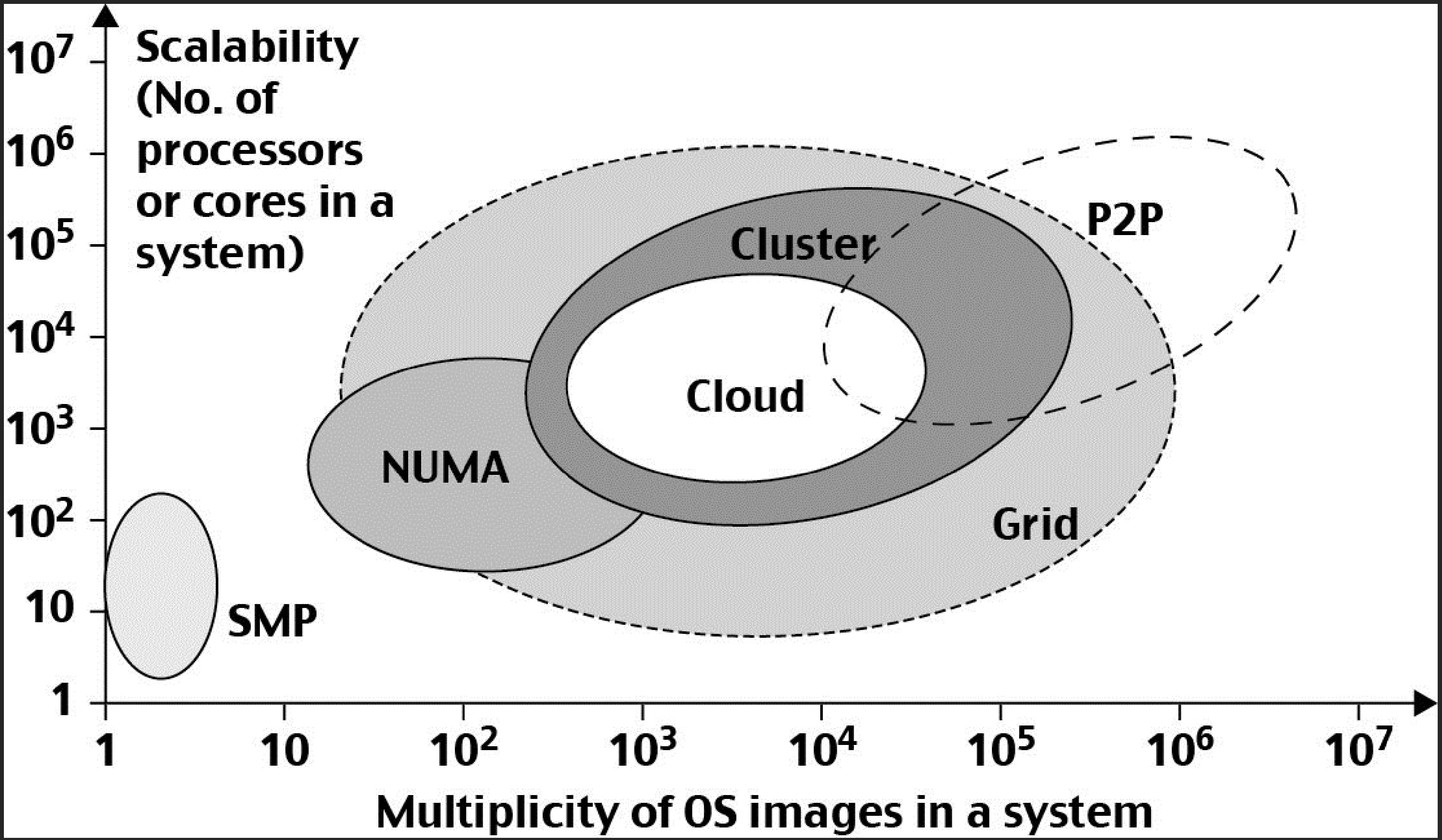
Table 1.9 Grid Standards and Toolkits for scientific and Engineering Applications

Grid Standards	Major Grid Service Functionalities	Key Features and Security Infrastructure
OGSA Standard	Open Grid Service Architecture offers common grid service standards for general public use	Support heterogeneous distributed environment, bridging CA, multiple trusted intermediaries, dynamic policies, multiple security mechanisms, etc.
Globus Toolkits	Resource allocation, Globus security infrastructure (GSI), and generic security service API	Sign-in multi-site authentication with PKI, Kerberos, SSL, Proxy, delegation, and GSS API for message integrity and confidentiality
IBM Grid Toolbox	AIX and Linux grids built on top of Globus Toolkit, autonomic computing, Replica services	Using simple CA, granting access, grid service (ReGS), supporting Grid application for Java (GAF4J), GridMap in IntraGrid for security update.

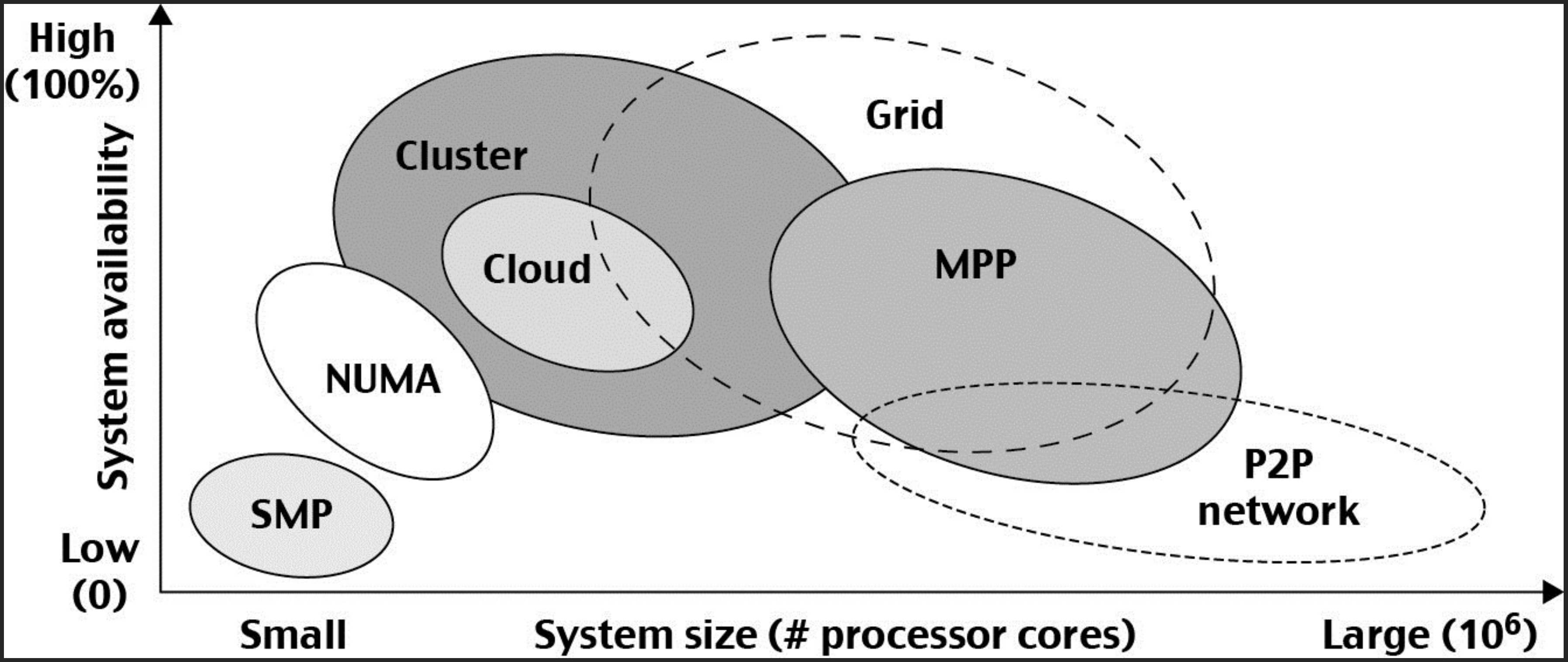
Dimensions of Scalability

- Size – increasing performance by increasing machine size
- Software – upgrade to OS, libraries, new apps.
- Application – matching problem size with machine size
- Technology – adapting system to new technologies

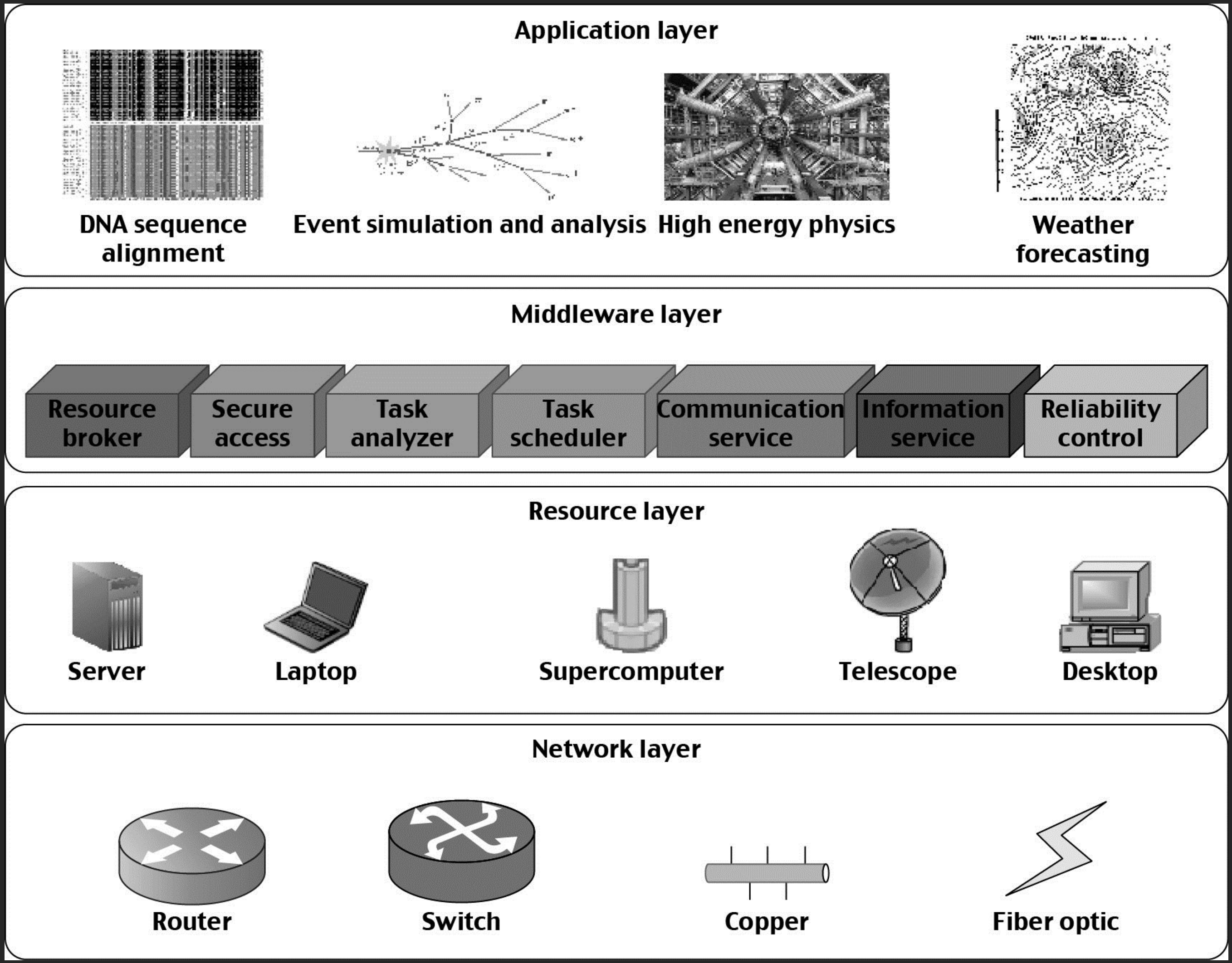
System Scalability vs. OS Multiplicity



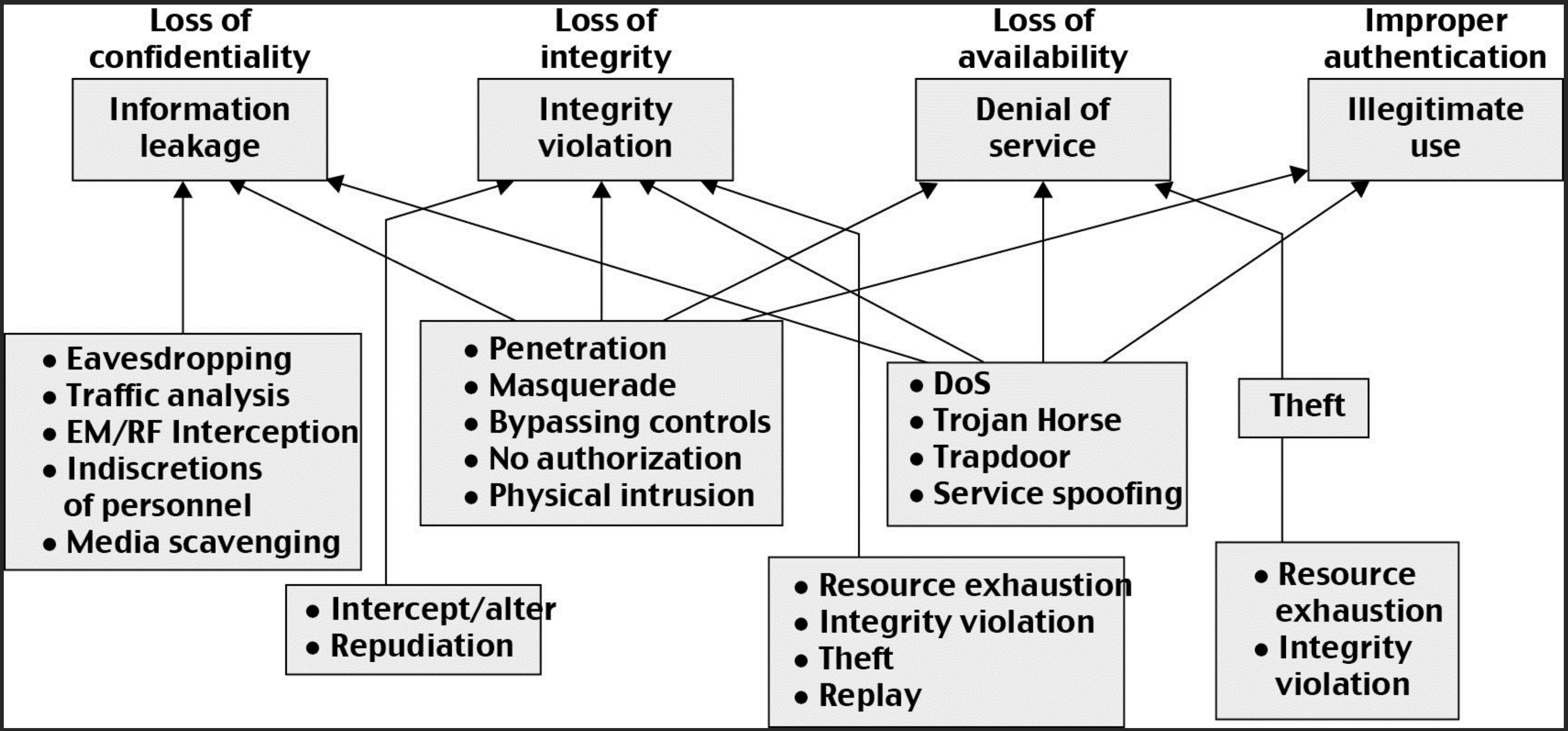
System Availability vs. Configuration Size :



Operational Layers of Distributed Computing System



Security: System Attacks and Network Threads



Four Reference Books:

1. K. Hwang, G. Fox, and J. Dongarra, *Distributed and Cloud Computing: from Parallel Processing to the Internet of Things* *Morgan Kauffmann Publishers*, 2011
2. R. Buyya, J. Broberg, and A. Goscinski (eds), Cloud Computing: Principles and Paradigms, ISBN-13: 978-0470887998, Wiley Press, USA, February 2011.
3. T. Chou, *Introduction to Cloud Computing: Business and Technology,* Lecture Notes at Stanford University and at Tsinghua University, Active Book Press, 2010.
4. T. Hey, Tansley and Tolle (Editors), *The Fourth Paradigm : Data-Intensive Scientific Discovery,* Microsoft Research, 2009.