Overview of Distributed and Cloud Computing
System Models and Enabling Technologies

The Evolution of Computer Systems and Applications

  - Early computer companies
  - Analog computers
  - Mainframe computers
    - Time-sharing
    - Real-time computing
  - Supercomputers
  - Minicomputers
  - Networking
  - Personal computers
  - Mobile computing
The Evolution of Computer Systems and Applications (cont.)

- Client-Server Computing
- Distributed Computing
- Virtualization and data centers
- Utility Computing
- Grid Computing
- Internet computing
- Web services
- Service-Oriented Computing (SOA)
- Mobile Computing
- Cloud Computing

Data Deluge

  - Small data
  - Big data
  - Data deluge
  - Data-intensive science – a new, 4th paradigm for scientific exploration, e-Science
  - Net information flow is growing faster than the IT investment required to “Store”, “Transmit,” “Analyze,” and “Manage” it
  - Where are these bottlenecks?
Coping with Data Deluge

- Data Deluge Enabling New Challenges

Interactions among 4 technical challenges
(source: Judy Qiu, Indiana University, 2011)

- Data Deluge
- Cloud Technology
- eScience,
  - Computational intensive science that is carried out in highly distributed network environments, or science that uses immense data sets that require Grid Computing. (source: http://en.wikipedia.org/wiki/E-Science)
- Multicore/Parallel Computing

(Courtesy of Judy Qiu, Indiana University, 2011)
1.1 Scalable Internet-based Computing

- **General Computing Trend**
  - Leverage shared web resources
  - Massive amount of data over the Internet

- **High Performance Computing (HPC)**
  - Supercomputers (massively parallel processors, MPP)
  - Clusters of cooperative computers; share computing resources
  - Physically connected in close range to one another

- **High Throughput Computing (HTC)**

---

**High Throughput Computing (HTC) & Applications**

- Peer-to-peer (P2P) networks – distributed file sharing and content delivery applications
- Web service platforms
- Cloud computing

**HTC Technologies**

- Improved batch processing speed
- Address acute problems at many data and enterprise computing centers
  - Cost, Energy saving, Security, Reliability
1.1 Scalable Internet-based Computing

- Three New Computing Paradigms
  - Web 2.0 Services
  - Internet Clouds
  - Internet of Things

- Computing Paradigm Distinction
  - Centralized computing
  - Parallel computing
  - Distributed computing
  - Cloud computing

Cloud and Internet of Things (IOT)

1.1 Scalable Internet-based Computing

- Degrees of Parallelism
  - Bit-level parallelism (BLP)
  - Instruction-level parallelism (ILP)
    - Pipelining, Super scalar computing, VLIW (very long instruction word) architecture, Multithreading
  - Data-level parallelism (DLP)
    - Single-instruction multiple data (SIMD)
  - Task-level parallelism (TLP)
    - Multicore processor and Chip Multiprocessors (CMPs)
  - Job-level parallelism (JLP)

- HPC for Science and HTC for Business Applications

(Courtesy of Raj Buyya, University of Melbourne, 2011)
1.2 Technologies for Network Based Systems

- Multicore CPUs and Multithreading Technologies
- GPU Computing (Graphics Co-processor)
- Memory, Storage, and Wide Area Networking
- Virtual Machines and Virtualization Middleware
- Data Center Virtualization for Cloud Computing
1.2 Technologies for Network Based Systems

**Figure 1.4**

Improvement in processor and network technologies over 33 years.

**Figure 1.5 Multicore Processor**

```
Multicore Processor

Core 1
L1 Cache

Core 2
L1 Cache

... ...

Core n
L1 Cache

L2 Cache

L3 Cache / DRAM
```
Figure 1.6 Multithreading Technologies

Figure 1.7 Architecture of a Many-Core Multiprocessor
GPU Interacting with a CPU Processor

- CPU (Central Processor Unit)
- GPU (Graphics Processor Unit)
**Figure 1.8** NVIDIA Fermi GPU built with 16 streaming multiprocessors (SMs) of 32 CUDA core each

**Number Prefix Used in Computer Technologies**

<table>
<thead>
<tr>
<th>Number Prefix Used</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yotta</td>
<td>Y</td>
</tr>
<tr>
<td>Zetta</td>
<td>Z</td>
</tr>
<tr>
<td>Exa</td>
<td>E</td>
</tr>
<tr>
<td>Peta</td>
<td>P</td>
</tr>
<tr>
<td>Tera</td>
<td>T</td>
</tr>
<tr>
<td>Giga</td>
<td>G</td>
</tr>
<tr>
<td>Mega</td>
<td>M</td>
</tr>
<tr>
<td>Kilo</td>
<td>k</td>
</tr>
<tr>
<td>hecto</td>
<td>h</td>
</tr>
<tr>
<td>deca</td>
<td>da</td>
</tr>
<tr>
<td>deci</td>
<td>d</td>
</tr>
<tr>
<td>centi</td>
<td>c</td>
</tr>
<tr>
<td>milli</td>
<td>m</td>
</tr>
</tbody>
</table>

| Factor |  |  |
|--------|  |  |
| $10^{24}$ | or | E24  |
| $10^{21}$ | or | E21  |
| $10^{18}$ | or | E18  |
| $10^{15}$ | or | E15  |
| $10^{12}$ | or | E12  |
| $10^9$  | or  | E9   |
| $10^6$  | or  | E6   |
| $10^3$  | or  | E3   |
| $10^2$  | or  | E2   |
| $10^1$  | or  | E1   |
| $10^{-1}$ | or | E-1  |
| $10^{-2}$ | or | E-2  |
| $10^{-3}$ | or | E-3  |
## Number Prefix Used in Computer Technologies

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Symbol</th>
<th>Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>micro</td>
<td>µ</td>
<td>(10^{-6}) or E-6</td>
</tr>
<tr>
<td>nano</td>
<td>n</td>
<td>(10^{-9}) or E-9</td>
</tr>
<tr>
<td>pico</td>
<td>p</td>
<td>(10^{-12}) or E-12</td>
</tr>
<tr>
<td>femto</td>
<td>f</td>
<td>(10^{-15}) or E-15</td>
</tr>
<tr>
<td>atto</td>
<td>a</td>
<td>(10^{-18}) or E-18</td>
</tr>
<tr>
<td>zepto</td>
<td>z</td>
<td>(10^{-21}) or E-21</td>
</tr>
<tr>
<td>yocto</td>
<td>y</td>
<td>(10^{-24}) or E-24</td>
</tr>
</tbody>
</table>

**Figure 1.10 Improvement in memory and disk technologies over 33 years**

![Image showing memory and disk capacity improvement over 33 years](image)

> **FIGURE 1.10**
> Improvement in memory and disk technologies over 33 years. The Seagate Barracuda XT disk has a capacity of 3.1TB in 2011.
> (Courtesy of Xiaowei Lou and Likong Chen of University of Southern California, 2011)
Figure 1.9 GPU, CPU Performance Comparison

GPU and CPU performance in GFlops/Watt/core, compared with 60 GFlops/Watt/core projected in future Exascale systems.

Figure 1.14 Datacenter and server cost distribution

Customer spending (SB) vs. Millions installed servers for different categories of spending.
Low Cost Design: Datacenter

- IDC 2009 Datacenter Report
- Data Center Cost
  - 30% - purchasing IT equipment; 33% - Chillers
  - 18% - Uninterruptable power supply; 9% - computer room HVAC; 7% - power distribution, lighting, transformer costs
- Low-Cost Design Philosophy
  - About 60 percent of the cost is allocated to Management & Maintenance
  - The server purchase cost did not increase much with time
  - Use commodity switches and networks
  - Use commodity x86 servers
  - The software layer handles
    - Network traffic balancing
    - Fault tolerance
    - Expandability

Datacenter Growth and Cost Breakdown

- U.S. Datacenter 2012-2016 Forecast (Doc # 237070)
  - From 2.94 million in 2012 to 2.89 million in 2016
  - From 611.4 million square feet in 2012 to more than 700 million square feet in 2016
- IDC Find Growth, Consolidation, and Changing Ownership Patterns in Worldwide Datacenter Forecast, Nov. 10, 2014,
Cloud Computing Enabling Technologies

- Convergence of Technologies
  1) Hardware virtualization and multi-core chips
  2) Utility and grid computing
  3) SOA (Service-Oriented Architecture), Web 2.0, and WS mashups (Web services)
  4) Atonomic computing and data center automation


Virtual Machine Architecture
(source VMWare, 2010)

After Virtualization:
- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines
Figure 1.12 Three VM Architecture

- VMM – Virtual Machine Monitor
- (a) VMM (hypervisor) in the privileged mode; bare-metal VM

Primitive Operations in Virtual Machines

VM multiplexing, suspension, provision, and migration in a distributed computing environment

(Courtesy of W. Rivestab, Keynote address ACM ASPLOS 2006 (441))
Concepts of Virtual Clusters

Prof. Paul Lin

Fig. 1. A Campus Area Grid

Fig. 2. Virtual machines in a cluster environment


1.3 System Models for Distributed and Cloud Computing Systems

- Four Groups of Massive Computer Systems: Clusters, P2P networks, computing grids, Internet clouds

Table 1.2 Classification of Distributed Parallel Computing Systems

<table>
<thead>
<tr>
<th>Functionality, Applications</th>
<th>Multicomputer Clusters [7, 33]</th>
<th>Peer-to-Peer Networks [40]</th>
<th>Data/Computational Grids [5, 42]</th>
<th>Cloud Platforms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architecture, Network, Connectivity and Size</td>
<td>Flexible network of client machines interconnected by high-speed network links over selected resource sites</td>
<td>Heterogeneous clusters interconnected by high-speed network links over selected resource sites</td>
<td>Virtualized cluster of servers over datacenters via service-level agreement</td>
<td></td>
</tr>
<tr>
<td>Control and Resources Management</td>
<td>Autonomous client nodes, free in and out with distributed self-organization</td>
<td>Centralized control, server oriented with authenticated security, and static resources</td>
<td>Dynamic resource provisioning of servers, storage, and networks over massive datasets</td>
<td></td>
</tr>
<tr>
<td>Applications and network-centric services</td>
<td>High-performance computing, search engines, and web services, etc.</td>
<td>Most appealing to business file sharing, content delivery, and social networking</td>
<td>Distributed super-computing, global problem solving, and datacenter services</td>
<td>Upgraded web search, utility computing, and outsourced computing services</td>
</tr>
<tr>
<td>Representative Operational Systems</td>
<td>Google search engine, SunBlade, IBM Road Runner, Cray XT4, etc.</td>
<td>Snort, eMule, BitTorrent, Napster, Kazaa, Skype, JXTA, and .NET</td>
<td>TeraGrid, GGFyN, UK EEE, D-Grid, ChinaGrid, etc.</td>
<td>Google App Engine, IBM Bluemix, Amazon Web Service(AWS), and Microsoft Azure</td>
</tr>
</tbody>
</table>
A Typical Cluster Architecture

- SAN: Storage Area Network
- LAN: Local Area Network

A cluster of servers interconnected by a high-bandwidth SAN or LAN with shared I/O devices and disk arrays; the cluster acts as a single computer attached to the Internet.

A Computational Grid

Computational grid or data grid providing computing utility, data and information services through resource sharing and cooperation among participating organizations.
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])

P2P Structure

FIGURE 1.17
The structure of a P2P System by mapping a physical IP network to an overlay network built with virtual links.

(Courtesy of Zhenyu Li, Institute of Computing Technology, Chinese Academy of Sciences, 2003)


### P2P (Peer-to-Peer) Network Families

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

---

**Figure 1.18 Basic Concept of Internet Clouds**

- Cloud Computing over the Internet
- Virtualized resource from data centers to form an Internet cloud, provisioning with hardware, software, storage, networks, and services for paid users to run their applications.
Figure 1.19 Three Cloud Service

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

Eight Reasons to Adapt to the Cloud for Upgraded Internet Applications and Web Services, page 36

1. Desired locations in areas with **protected space and high energy efficiency**
2. Sharing of peak-load capacity among a large pool of users, improving overall utilization
3. Separation of infrastructure maintenance duties from domain-specific application development
4. Significant reduction in cloud computing cost, compared with traditional computing paradigms
5. Cloud computing programming and application development
6. Service and data discovery and content/service distribution
7. Privacy, security, copyright, and reliability issues
8. Service agreements, business models, and pricing policies
1.4 Software Environments for Distributed Systems and Clouds

- Service-Oriented Architecture (SOA)
  - Layered architecture for web services and grids
  - Web services and tools:
    - XML
    - Web services: SOAP (Simple Object Access Protocol), WSDL (Web Service Description Language)
    - REST (Representational State Transfer)
- Trends toward Distributed Operating Systems
- Parallel and Distributed Programming Models
  - Message-Passing Interface (MPI)
  - MapReduce
  - Hadoop
- Performance, Security, and Energy Efficiency
**Figure 1.20 Layered Architecture for Web Services and the Grids**

- Application specific services/ grids
- Generally useful services and grids
- Workflow
- Service management
- Service discovery and information
- Service Internet transport → Protocol
- Service Interfaces

**Base hosting environment**
- Protocol HTTP FTP DNS ...
- Presentation XDR ...
- Session SSH ...
- Transport TCP UDP ...
- Network IP ...
- Data link/Physical

HTTP – Hypertext Transfer Protocol, FTP – File Transfer Protocol
DNS – Domain Name Service
XDR – External Data Representation (FRC-1014)

**Figure 1.21 The Evolution of SOA** (ss – sensor services, fs – filter or transforming service)

Raw data → Data → Information → Knowledge → Wisdom → Decisions

Another grid
Another service
Database

Filter cloud
Discovery cloud
Portal
Inter-service messages

Traditional grid with exposed services
Sensor or data interchange service
Figure 1.22 A Ideal Model for Cloud Computing: A transparent computing environment that separates the user data, application, OS, and hardware in time and space.

Table 1.6 Feature Comparison of Three Distributed Operating Systems

<table>
<thead>
<tr>
<th>Distributed OS Functionality</th>
<th>AMOEBA developed at Vrije University [46]</th>
<th>DCE as OSF/1 by Open Software Foundation [7]</th>
<th>MOSIX for Linux Cluster at Hebrew University [3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>History and Current System Status</td>
<td>Written in C and tested in the European community; version 5.2 released in 1996</td>
<td>Built as a user extension on top of UNIX, VMS, Windows, OS/2, etc.</td>
<td>Developed since 1977, now called MOSIX2, used in HPC Linux and GPU clusters</td>
</tr>
<tr>
<td>Distributed OS Architecture</td>
<td>Microkernel-based and location-transparent, uses many servers to handle file, directory, replication, run, boot, and TCP/IP services</td>
<td>Middleware OS providing a platform for running distributed applications. The system supports RPC, security, and threads</td>
<td>A distributed OS with resource discovery, process migration, runtime support, load balancing, fault control, configuration, etc.</td>
</tr>
<tr>
<td>OS Kernel, Middleware, and Virtualization Support</td>
<td>A special microkernel that handles low-level processes, memory, I/O, and communication functions</td>
<td>DCE packages handle file, time, directory, security services, RPC, and authentication at middleware or user space</td>
<td>MOSIX2 runs with Linux 2.6; extensions for use in multiple clusters and clouds with provisioned VMs</td>
</tr>
<tr>
<td>Communication Mechanisms</td>
<td>Uses a network-layer PUP protocol and RPC to implement point-to-point and group communication</td>
<td>RPC supports authenticated communication and other security services in user programs</td>
<td>Using PVM, MPI in collective communications, priority process control, and queueing services</td>
</tr>
</tbody>
</table>
### Parallel and Distributed Programming

**Table 1.7 Parallel and Distributed Programming Models and Tool Sets**

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

### Grid Standards and Middleware

**Table 1.9 Grid Standards and Toolkits for scientific and Engineering Applications**

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

- Network bandwidth (Mbps – Million bits per second)
- CPU Speed – MIPS (Million Instructions Per Second)
- System throughput - Performance Metrics
  - MIPS (Million Instruction Per Second) – CPU speed
  - Tflops (Tera floating-point operations per second; Tera = 10^{12})
  - TPSs (Transactions per Second)
  - Job response time
  - Network latency
  - System overhead: OS boot time, compile time, I/O data rate, runtime support system
- QoS – Internet and Web services
- System availability, dependability, security resilience
Scalability Analysis

- Dimensions of Scalability
  - **Size Scalability**
    - Achieve higher performance by increasing machine size
    - Machine size (Ex: 512 processors in 1997 => 65,000 processors)
  - **Software Scalability**
    - Upgrade OS, Compilers Libraries, new software, etc.
  - **Application Scalability**
    - Problem size
    - Matching “Problem size” with “Machine size”
  - **Technology Scalability**
    - Time (generation scalability), Space (packaging and energy concerns), and Heterogeneity (use hardware components or software packages from different vendors)

System Scalability vs. OS. Multiplicity

*Figure 1.23*

System scalability versus multiplicity of OS images based on 2010 technology.
Example 1-2. Energy efficiency in distributed power management

- **DVFS (Dynamic Voltage & Frequency Scaling) Method for Energy Efficiency**
  - Reducing clock frequency or voltage during slack time (idle time)
  - Relationship between Energy and Voltage Frequency in CMOS circuit: $E = C_{eff} f^2 t$, $f = K \frac{(v - v_t)^2}{v}$

![DVFS technique diagram](image)

Amdahl’s Laws

- **Amdahl’s Law** – a law governing the speed up of using parallel processors on a problem
- Execute a given program on a uniprocessor computer with a total execution time of T minutes.

Parallel Computing

- Ignore all system, I/O time, exception handling or communication overhead. The program is now parallelized or partitioned for parallel execution on a cluster of “many processing nodes”
- Assume that a fraction “Alpha $\alpha$” of the code must be executed sequentially – Sequential Bottleneck
- $(1 - \alpha)$ of the code can be compiled for execution by “N processors”
- The total execution time of the program $(T) = \text{Sequential Execution Time} + \text{Parallel Execution Time}$

$$\alpha T + (1 - \alpha)T/N$$
Amdahl’s Laws (cont.)

- Amdahl’s law states that the speedup factor of using the N-processor system over the use of a single processor is expressed by

\[
\text{Speedup} = S = \frac{T}{\alpha T + \frac{(1 - \alpha)T}{N}} = \frac{1}{\alpha + \frac{1 - \alpha}{N}}
\]

- Fixed workload speedup
- Maximum speedup: if \( \alpha \approx 0 \), or the code is fully parallelizable
- As cluster size increases \( N \approx \) Infinity; \( S \approx 1/\alpha \)
  - Upper bound is independent of cluster size \( N \)
  - If \( \alpha = 0.25 \), or \( 1 - \alpha = 0.75 \), the maximum speedup achieved is 4. (Even if hundreds of processors is used)
- We should make \( \alpha \) as small as possible. Increase the cluster size alone may not result in a good speedup

Problem with Fixed Workload

- To execute a fixed workload on \( n \) processors, parallel processing may lead to a system efficiency defined as:

\[
E = S = \frac{\alpha}{\alpha + \frac{1 - \alpha}{N}} = \frac{1}{[\alpha N + 1 - \alpha]}
\]

- \( \alpha = 0.25 \) \( N = 256 \) nodes, \( E = 1/[0.25*256 + 0.75] = 1.5% \)
- Only a few processors (say, 4) are kept busy, while the majority of the nodes are left idling.
- Very often the system efficiency is very low, especially when the cluster size is very large.
Gustafson’s Law

  - Scaling the problem size to match the “cluster capacity.”
  - Let $W = $ workload in a given program.
  - When using an $N$-processor system, the user scaled the workload to

$$W' = \alpha W + (1 - \alpha)NW$$

- Only the parallelizable portion of the workload is scaled $N$ times in the second term.

$$S' = \frac{W'}{W} = \frac{[\alpha W + (1 - \alpha)NW]}{W} = \alpha + (1 - \alpha)N$$

- By fixing the parallel execution time at level $W$, the following efficiency expression is obtained:

$$E' = S'/N = \frac{\alpha}{N} + (1 - \alpha)$$

Improved efficiency: a 256 node cluster, $\alpha = 0.25$,

$$E' = 0.25/256 + 0.75 = 0.751$$

Fault Tolerance and System Availability

- System Availability
  - $\text{System Availability} = \frac{\text{MTTF}}{\text{MTTF} + \text{MTTR}}$
- A system is highly available if it has
  - a long “Mean Time To Failure (MTTF)” and
  - a short “Mean Time to Repair (MTTR)”
System Availability vs. Configuration Size

Estimated system availability by system size of common configurations in 2010.

System Attacks & Network Threats

Various system attacks and network threats to the cyberspace.
Figure 1.22 Transparent Cloud Computing Environment

Figure 3  Transparent computing that separates the user data, application, OS, and hardware in time and space – an ideal model for future Cloud platform construction

Summary & Conclusion