

Module-3

Cloud Platform Architecture over Virtualized Datacenters: Cloud Computing and Service Models, Data Center Design and Interconnection Networks, Architectural Design of Compute and Storage Clouds, Public Cloud Platforms: GAE, AWS and Azure, Inter-Cloud Resource Management.

Textbook 1: Chapter 4: 4.1 to 4.5

4.1 Cloud Computing and Service Models**4.1.1 Public, Private, and Hybrid Clouds**

- **Public Cloud:**

- Hosted and maintained by third-party cloud service providers.
- Resources such as computing power, storage, and networking are shared among multiple customers.
- Offers cost-effectiveness, scalability, and flexibility.
- Examples: Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP).
- Disadvantages: Limited control over infrastructure, potential security concerns due to shared environments.

- **Private Cloud:**

- Dedicated cloud infrastructure for a single organization, either hosted on-premises or by a third-party provider.
- Provides enhanced security, compliance, and performance optimization.
- Ideal for industries with strict regulatory requirements such as healthcare and finance.
- Higher operational costs due to maintenance and hardware requirements.
- Examples: VMware vSphere, OpenStack-based private clouds.

- **Hybrid Cloud:**

- A combination of public and private cloud infrastructures to optimize performance and cost-efficiency.
- Workloads can be dynamically shifted between private and public clouds based on business needs.
- Benefits include greater flexibility, scalability, and disaster recovery options.
- Example: AWS Outposts, Microsoft Azure Stack.

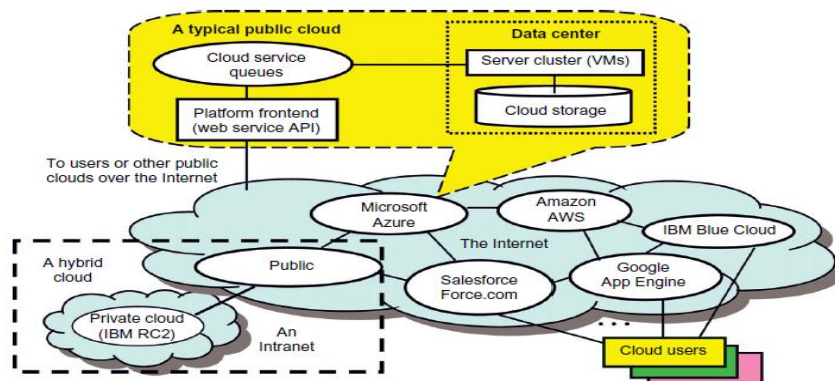


FIGURE 4.1

Public, private, and hybrid clouds illustrated by functional architecture and connectivity of representative clouds available by 2011.

4.1.1.5 Data-Center Networking Structure

- Cloud computing enables dynamic resource allocation through virtual clusters, with gateway nodes serving as access points and security controls. Unlike traditional grids, which rely on static resource allocation, cloud platforms handle fluctuating workloads dynamically. Private clouds, when well-designed, can efficiently meet these demands.
- Data centers and supercomputers share some similarities but differ significantly in scale, architecture, and networking. Data centers rely on large clusters of servers with integrated storage and cache, whereas supercomputers use separate data farms. Networking also differs: supercomputers utilize high-bandwidth, custom networks, while data centers rely on IP-based networks, such as 10 Gbps Ethernet.
- Private cloud examples include NASA's cloud for climate modeling and CERN's cloud for distributing research resources globally. These clouds require varying levels of performance, security, and data protection, governed by Service Level Agreements (SLAs). Cloud computing builds upon grid computing but focuses more on scalable, abstracted services rather than just storage and computing resources.

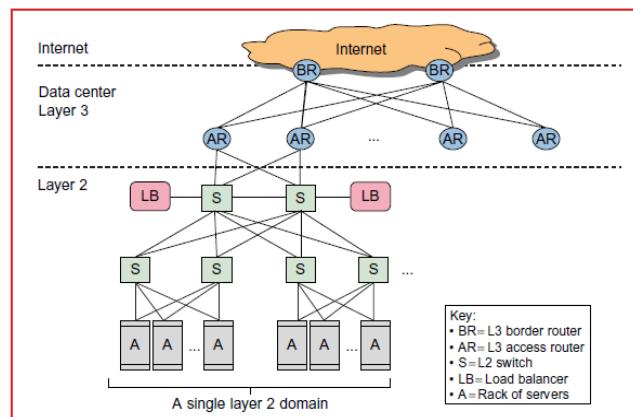


FIGURE 4.2

Standard data-center networking for the cloud to access the Internet.

Cloud Development Trends

Private clouds are expected to grow faster than public clouds due to their enhanced security and trust within organizations. As they mature, they may transition into public or hybrid clouds, blurring the distinction between cloud types. Hybrid clouds are likely to dominate the future.

Cloud services are categorized into different types of nodes: service-access nodes handle user interactions, runtime supporting service nodes assist cloud operations, and independent service nodes provide specialized services. Cloud computing optimizes performance by minimizing data movement, reducing Internet traffic, and addressing petascale I/O challenges. However, cloud performance, security, and QoS still require further validation through real-world applications.

4.1.2 Cloud Ecosystem and Enabling Technologies

Cloud computing aims to shift processing, storage, and software delivery from desktops to data centers, ensuring scalability, efficiency, and economic viability through a pay-as-you-go model. Key design objectives include:

1. Shifting Computing to Data Centers – Moving resources from local machines to centralized cloud infrastructure.
2. Service Provisioning & Economics – Efficient resource utilization with SLAs and cost-effective pricing models.
3. Scalability – Cloud platforms must support increasing user demands seamlessly.
4. Data Privacy Protection – Ensuring trust in cloud providers to handle sensitive data securely.
5. High-Quality Services – Standardizing QoS for interoperability across cloud providers.
6. New Standards & Interfaces – Addressing data lock-in issues with universal APIs and flexible access protocols.

Cloud computing reduces costs by eliminating capital expenses for hardware acquisition and shifting to a pay-per-use model. Traditional IT systems incur both fixed and increasing operational costs as users grow, whereas cloud computing minimizes upfront investment, making it attractive for startups and enterprises.

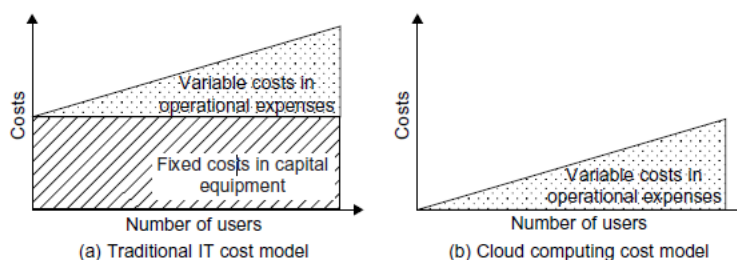


FIGURE 4.3

Computing economics between traditional IT users and cloud users.

• Cloud Ecosystem Components:

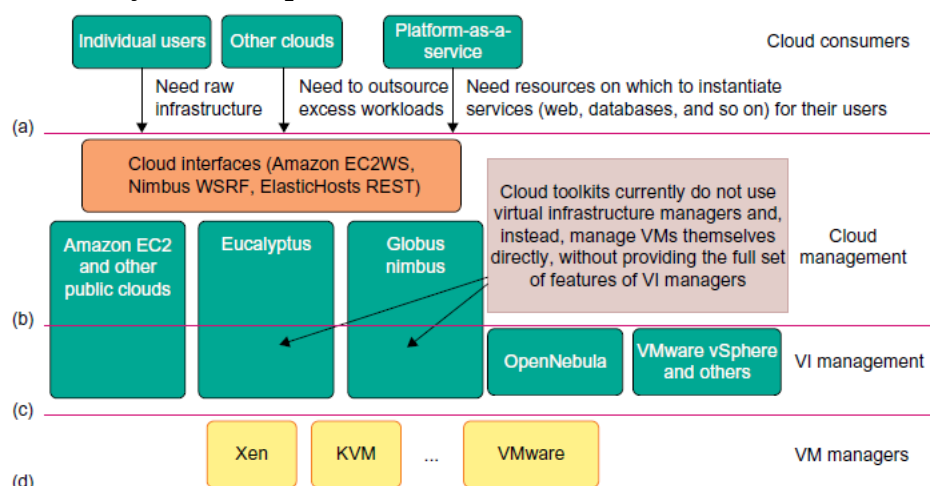


FIGURE 4.4

Cloud ecosystem for building private clouds: (a) Consumers demand a flexible platform; (b) Cloud manager provides virtualized resources over an IaaS platform; (c) VI manager allocates VMs; (d) VM managers handle VMs installed on servers.

- **Cloud Providers:** AWS, Google Cloud, Microsoft Azure, IBM Cloud, Oracle Cloud.
- **Cloud Consumers:** Businesses, developers, end-users leveraging cloud services.
- **Cloud Brokers:** Intermediaries that manage service usage and performance.
- **Security and Compliance Solutions:** Identity management, encryption, and regulatory compliance tools.
- **Key Enabling Technologies:**
 - **Virtualization:** Allows multiple virtual machines (VMs) to run on a single physical server, improving efficiency and flexibility.
 - **Containerization:** Docker and Kubernetes enable efficient application deployment across cloud environments.
 - **Microservices Architecture:** Supports modular application development, enhancing scalability and manageability.
 - **Artificial Intelligence (AI) & Machine Learning (ML):** Enables cloud-based predictive analytics, automation, and intelligent decision-making.
 - **Blockchain Technology:** Used for secure transactions, data integrity, and decentralized cloud services.

4.1.3 Infrastructure-as-a-Service (IaaS)

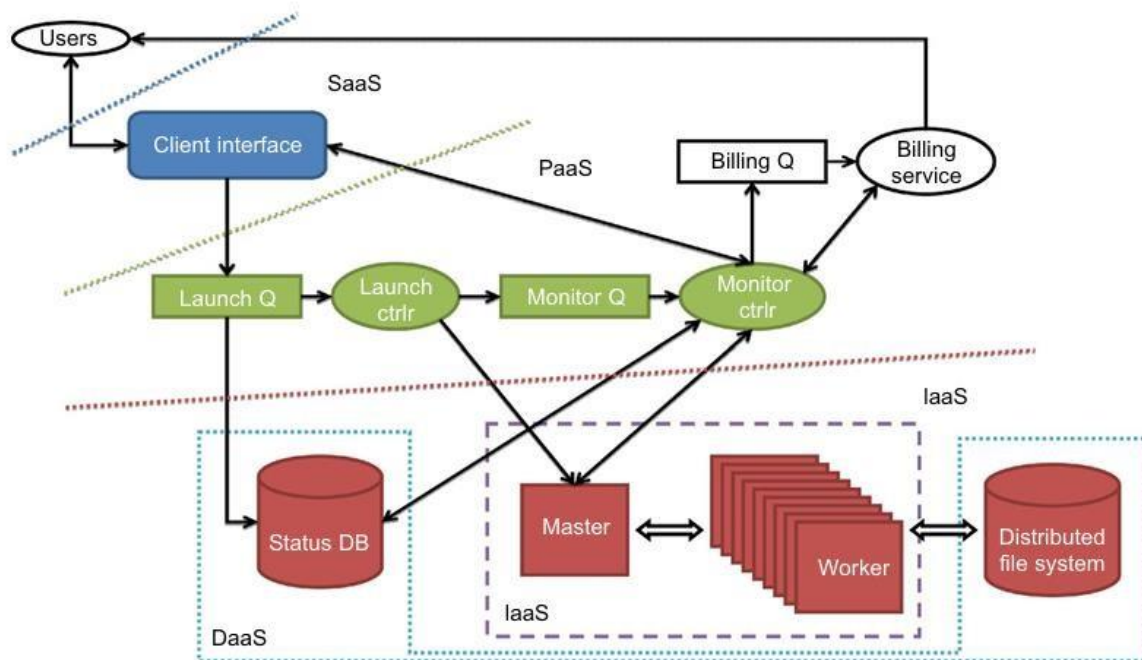


FIGURE 4.5

The IaaS, PaaS, and SaaS cloud service models at different service levels..

(Courtesy of J. Suh and S. Kang, USC)

- **Definition:**
 - Provides essential computing resources (compute, storage, networking) as virtualized services over the internet.
- **Key Characteristics:**

- Pay-as-you-go pricing model.
- Highly scalable infrastructure.
- Automated resource provisioning.
- **Examples:**

Cloud Name	VM Instance Capacity	API and Access Tools	Hypervisor, Guest OS
Amazon EC2	Each instance has 1–20 EC2 processors, 1.7–15 GB of memory, and 160–1.69 TB of storage.	CLI or web Service (WS) portal	Xen, Linux, Windows
GoGrid	Each instance has 1–6 CPUs, 0.5–8 GB of memory, and 30–480 GB of storage.	REST, Java, PHP, Python, Ruby	Xen, Linux, Windows
Rackspace Cloud	Each instance has a four-core CPU, 0.25–16 GB of memory, and 10–620 GB of storage.	REST, Python, PHP, Java, C#, .NET	Xen, Linux
FlexiScale in the UK	Each instance has 1–4 CPUs, 0.5–16 GB of memory, and 20–270 GB of storage.	web console	Xen, Linux, Windows
Joyent Cloud	Each instance has up to eight CPUs, 0.25–32 GB of memory, and 30–480 GB of storage.	No specific API, SSH, Virtual/Min	OS-level virtualization, OpenSolaris

- AWS EC2: Provides virtual machines with flexible configurations.

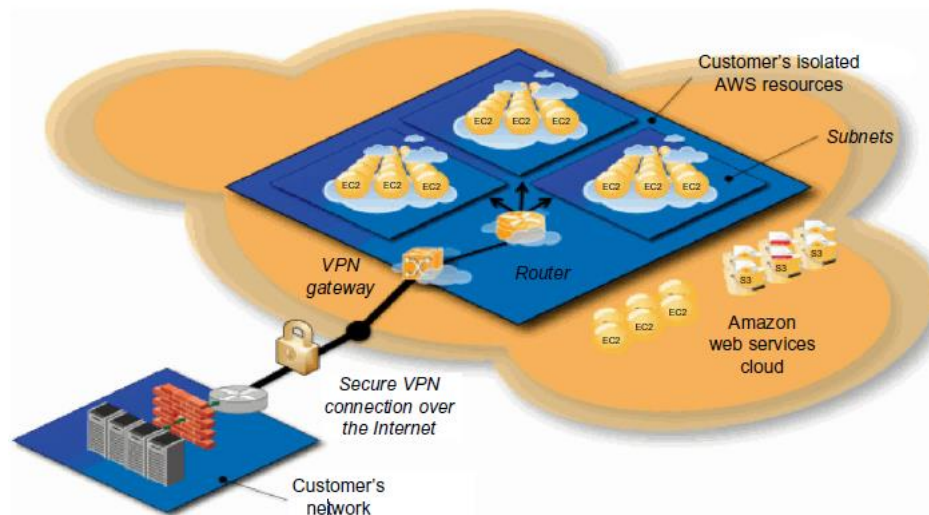


FIGURE 4.6

- Amazon VPC (virtual private cloud).
- Google Compute Engine (GCE): Scalable VM instances for cloud workloads.
- Microsoft Azure Virtual Machines: Supports both Windows and Linux environments.
- **Advantages:**
 - Eliminates the need for physical hardware investment.
 - Provides high availability and disaster recovery solutions.
 - Offers global reach and reduced latency through distributed data centers.
- **Challenges:**
 - Requires advanced cloud expertise for configuration and management.
 - Potential security risks if not properly configured.

4.1.4 Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS)

- **Platform-as-a-Service (PaaS):**

- Provides a cloud-based environment for application development and deployment.
- Includes integrated tools such as databases, runtime environments, and development frameworks.
- Examples:

Table 4.2 Five Public Cloud Offerings of PaaS [10,18]

Cloud Name	Languages and Developer Tools	Programming Models Supported by Provider	Target Applications and Storage Option
Google App Engine	Python, Java, and Eclipse-based IDE	MapReduce, web programming on demand	Web applications and BigTable storage
Salesforce.com's Force.com	Apex, Eclipse-based IDE, web-based Wizard	Workflow, Excel-like formula, Web programming on demand	Business applications such as CRM
Microsoft Azure	.NET, Azure tools for MS Visual Studio	Unrestricted model	Enterprise and web applications
Amazon Elastic MapReduce	Hive, Pig, Cascading, Java, Ruby, Perl, Python, PHP, R, C++	MapReduce	Data processing and e-commerce
Aneka	.NET, stand-alone SDK	Threads, task, MapReduce	.NET enterprise applications, HPC

- - Google App Engine: Scalable application hosting.
 - AWS Elastic Beanstalk: Automated deployment and scaling of web applications.
 - Microsoft Azure App Services: PaaS offering for .NET, Java, Python, and more.

▪

- **Benefits:**
 - Reduces the complexity of software development.
 - Allows developers to focus on writing code instead of managing infrastructure.
 - Scalable and flexible resource allocation.
- **Challenges:**

- Limited control over the underlying infrastructure.
 - Vendor lock-in concerns.
- **Software-as-a-Service (SaaS):**
 - Delivers software applications over the internet without requiring installation on local machines.
 - Users access software via web browsers on a subscription basis.
 - Examples:
 - Google Workspace (Docs, Sheets, Gmail): Cloud-based productivity tools.
 - Microsoft 365: Office applications and collaboration tools.
 - Salesforce: Customer relationship management (CRM) software.
 - **Benefits:**
 - Easy access from anywhere with an internet connection.
 - Automatic updates and maintenance handled by the provider.
 - Lower upfront costs compared to traditional software licensing.
 - **Challenges:**
 - Data security and privacy concerns.
 - Dependence on internet connectivity.
 - Limited customization compared to on-premises solutions.

Mashup of Cloud Services

At the time of this writing, public clouds are in use by a growing number of users. Due to the lack of trust in leaking sensitive data in the business world, more and more enterprises, organizations, and communities are developing private clouds that demand deep customization. An enterprise cloud is used by multiple users within an organization. Each user may build some strategic applications on the cloud, and demands customized partitioning of the data, logic, and database in the metadata representation. More private clouds may appear in the future.

Based on a 2010 Google search survey, interest in grid computing is declining rapidly. Cloud

mashups have resulted from the need to use multiple clouds simultaneously or in sequence. For example, an industrial supply chain may involve the use of different cloud resources or services at different stages of the chain. Some public repository provides thousands of service APIs and mashups for web commerce services. Popular APIs are provided by Google Maps, Twitter, YouTube, Amazon eCommerce, Salesforce.com, etc

4.2 Data-Center Design and Interconnection Networks

4.2.1 Warehouse-Scale Data-Center Design

- **Definition:** Large-scale data centers that host thousands of servers to support cloud operations.
- **Key Design Factors:**
 - **Redundancy:** Ensures fault tolerance and high availability.
 - **Scalability:** Allows the data center to expand resources efficiently.
 - **Energy efficiency:** Uses cooling techniques such as liquid cooling and free air cooling.



FIGURE 4.8

- A huge data center that is 11 times the size of a football field, housing 400,000 to 1 million servers.
- **Security measures:** Includes biometric access control, surveillance, and fire suppression systems.
- **Example:** Google's warehouse-scale computing infrastructure, which features custom-built hardware and software optimization.
- Data centers use raised floors to manage power cables and distribute cool air efficiently. The **cooling system** relies on **Computer Room Air Conditioning (CRAC) units**, which pressurize the raised floor plenum with cold air. Perforated tiles release this air in front of server racks, which are arranged in alternating **cold and hot aisles** to prevent heat mixing. Hot air from servers is recirculated back to the CRAC units, cooled, and reintroduced into the system.
- Modern data centers enhance cooling with **water-based free cooling** and **cooling towers**, which pre-cool condenser water before it reaches the chiller, improving efficiency and reducing energy consumption.

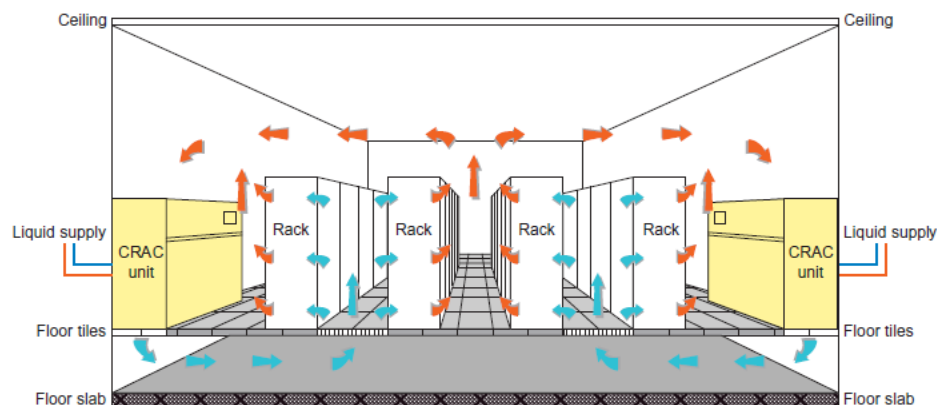


FIGURE 4.9

The cooling system in a raised-floor data center with hot-cold air circulation supporting water heat exchange facilities.

4.2.2 Data-Center Interconnection Networks

- **Purpose:** Facilitates communication between servers, storage, and network devices.
- **Common Network Topologies:**
 - **Fat-tree architecture:** Provides high bandwidth and low latency, used in cloud data centers.

- **Clos network:** Ensures efficient routing and redundancy.
- **Software-Defined Networking (SDN):** Enables dynamic network configuration and automation.
- **Load Balancing:** Distributes network traffic evenly to prevent bottlenecks.
- **Fault Tolerance:** Ensures network resilience through redundant connections.

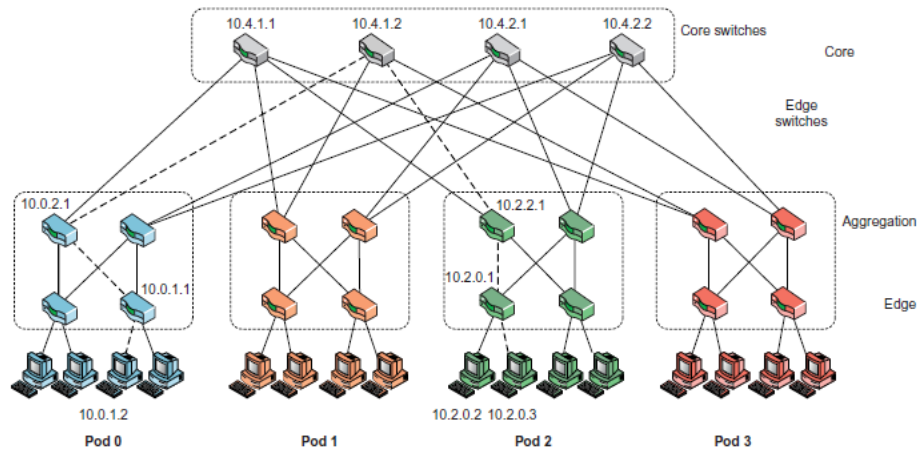


FIGURE 4.10

A fat-tree interconnection topology for scalable data-center construction.

4.2.3 Modular Data Centers in Shipping Containers

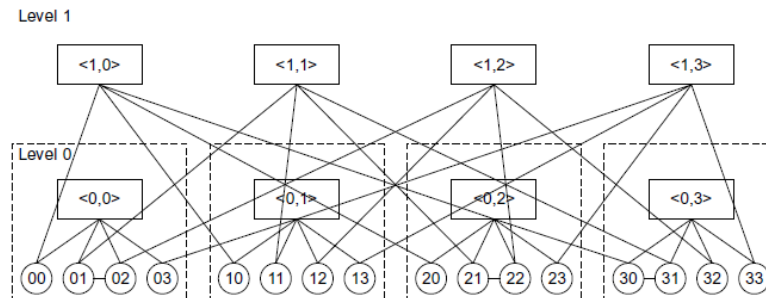
- **Definition:** Prefabricated data centers housed in shipping containers.
- **Advantages:**
 - **Portability:** Can be transported to different locations as needed.
 - **Scalability:** Easy to add more modules for additional capacity.
 - **Energy Efficiency:** Designed with advanced cooling systems to reduce power consumption.
- **Use Cases:** Disaster recovery, military applications, rapid deployment in remote areas.
- **Example:** Microsoft's Azure Modular Data Centers, which support cloud workloads in various locations.

4.2.4 Interconnection of Modular Data Centers

- **Techniques Used:**
 - **Fiber-optic networking:** Ensures high-speed data transfer between modules.
 - **SDN integration:** Manages dynamic network reconfigurations.
 - **Data replication strategies:** Ensures data consistency across distributed centers.
- **Challenges:**
 - **Latency issues:** Distance between modular centers can impact performance.
 - **Security risks:** Data transmission over long distances requires encryption

Example 4.5 A Server-Centric Network for a Modular Data Center

Guo, et al. [30] have developed a server-centric BCube network (Figure 4.12) for interconnecting modular data centers. The servers are represented by circles, and switches by rectangles. The BCube provides a layered structure. The bottom layer contains all the server nodes and they form Level 0. Level 1 switches form the top layer of BCube₀. BCube is a recursively constructed structure. The BCube₀ consists of n servers connecting to an n -port switch. The BCube _{k} ($k \geq 1$) is structured from n BCube _{$k-1$} with n^k n -port switches. The example of BCube₁ is illustrated in Figure 4.12, where the connection rule is that the i -th server in the j -th BCube₀ connects to the j -th port of the i -th Level 1 switch. The servers in the BCube have multiple ports attached. This allows extra devices to be used in the server.

**FIGURE 4.12**

BCube, a high-performance, server-centric network for building modular data centers.

4.2.5 Data-Center Management Issues

Here are basic requirements for managing the resources of a data center. These suggestions have resulted from the design and operational experiences of many data centers in the IT and service industries.

- Making common users happy The data center should be designed to provide quality service to the majority of users for at least 30 years.
- Controlled information flow Information flow should be streamlined. Sustained services and high availability (HA) are the primary goals.
- Multiuser manageability The system must be managed to support all functions of a data center, including traffic flow, database updating, and server maintenance.
- Scalability to prepare for database growth The system should allow growth as workload increases. The storage, processing, I/O, power, and cooling subsystems should be scalable.
- Reliability in virtualized infrastructure Failover, fault tolerance, and VM live migration should be integrated to enable recovery of critical applications from failures or disasters.
- Low cost to both users and providers The cost to users and providers of the cloud system built over the data centers should be reduced, including all operational costs.
- Security enforcement and data protection Data privacy and security defense mechanisms must be deployed to protect the data center against network attacks and system interrupts and to maintain data integrity from user abuses or network attacks.
- Green information technology Saving power consumption and upgrading energy efficiency are in high demand when designing and operating current and future data centers.
- **Key Challenges:**
 - **Resource allocation:** Ensuring optimal usage of compute, storage, and network resources.

- **Security and compliance:** Adhering to industry standards such as GDPR and HIPAA.
- **Energy efficiency:** Implementing cooling solutions and renewable energy sources.
- **Infrastructure monitoring:** Using AI-powered analytics to predict failures and optimize performance.
- **Automation:** Implementing orchestration tools like Kubernetes for workload management.

4.3 Architectural Design of Compute and Storage Clouds

Cloud Platform Design Goals: Cloud computing platforms are designed with four key goals: scalability, virtualization, efficiency, and reliability. They support Web 2.0 applications by managing user requests, allocating resources, and provisioning services across both physical and virtual machines. Security in shared environments remains a critical challenge.

To establish a large-scale HPC infrastructure, cloud platforms integrate hardware and software for seamless operation. Scalability is achieved through cluster architecture, allowing for easy expansion of processing power, storage, and bandwidth. Reliability is enhanced by storing data in multiple geographically distributed locations, ensuring accessibility even in case of failures. Cloud architectures can scale efficiently by adding more servers and expanding network connectivity as needed.

Enabling Technologies for Clouds

Cloud computing is driven by advancements in broadband and wireless networking, declining storage costs, and improvements in Internet computing software. These technologies enable on-demand capacity scaling, cost reduction, and service experimentation for users, while providers benefit from higher system utilization through multiplexing, virtualization, and dynamic resource provisioning.

Clouds rely on continuous progress in hardware, software, and networking, which collectively enhance performance, efficiency, and scalability. These advancements ensure that cloud platforms can dynamically allocate resources while maintaining cost-effectiveness and flexibility. Cloud computing has become a reality due to advancements in **hardware, virtualization, and service-oriented architecture (SOA)**. **Multicore CPUs, memory chips, and disk arrays** enable faster data centers with vast storage capacity. **Virtualization** supports rapid cloud deployment and disaster recovery, while **SOA** enhances cloud service integration.

Progress in **SaaS, Web 2.0 standards, and Internet performance** has driven cloud adoption, allowing for large-scale, multi-tenant services over massive datasets. **Distributed storage systems** form the backbone of modern data centers, while improvements in **license management and automatic billing** further streamline cloud operations.

Table 4.3 Cloud-Enabling Technologies in Hardware, Software, and Networking	
Technology	Requirements and Benefits
Fast platform deployment	Fast, efficient, and flexible deployment of cloud resources to provide dynamic computing environment to users
Virtual clusters on demand	Virtualized cluster of VMs provisioned to satisfy user demand and virtual cluster reconfigured as workload changes
Multitenant techniques	SaaS for distributing software to a large number of users for their simultaneous use and resource sharing if so desired
Massive data processing	Internet search and web services which often require massive data processing, especially to support personalized services
Web-scale communication	Support for e-commerce, distance education, telemedicine, social networking, digital government, and digital entertainment applications
Distributed storage	Large-scale storage of personal records and public archive information which demands distributed storage over the clouds
Licensing and billing services	License management and billing services which greatly benefit all types of cloud services in utility computing

4.3.1 A Generic Cloud Architecture Design

- **Three-layer architecture:**
 1. **Compute layer:** Manages virtual machines (VMs), containers, and serverless functions.
 2. **Storage layer:** Includes block storage, object storage, and file storage solutions.
 3. **Networking layer:** Handles firewalls, load balancers, and Software-Defined Networking (SDN).

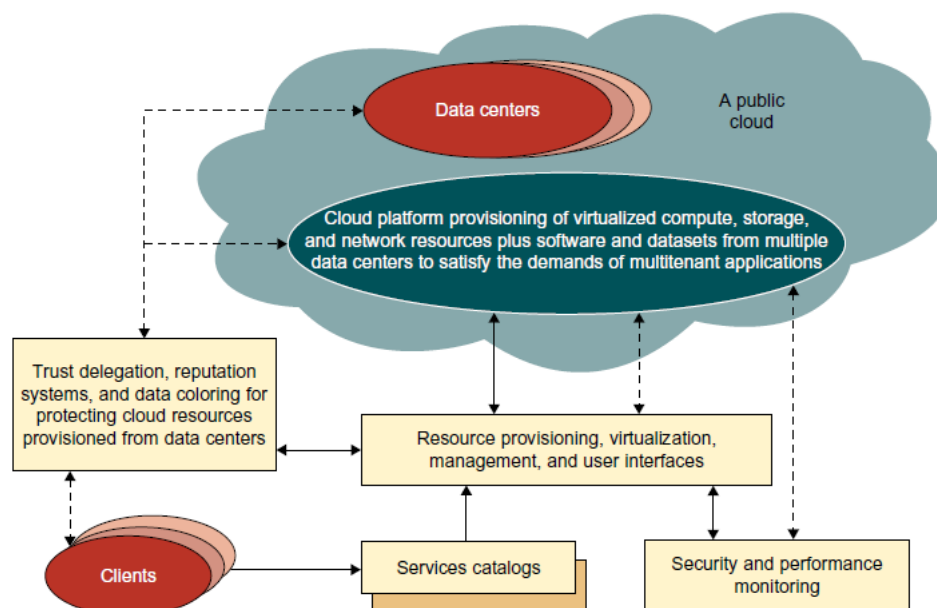


FIGURE 4.14

A security-aware cloud platform built with a virtual cluster of VMs, storage, and networking resources over the data-center servers operated by providers.

4.
 - **Key Characteristics:**
 - On-demand resource allocation.
 - Elastic scalability to meet workload demands.
 - Centralized management and orchestration.
 - Automated provisioning using APIs and infrastructure as code (IaC).

4.3.2 Layered Cloud Architectural Development

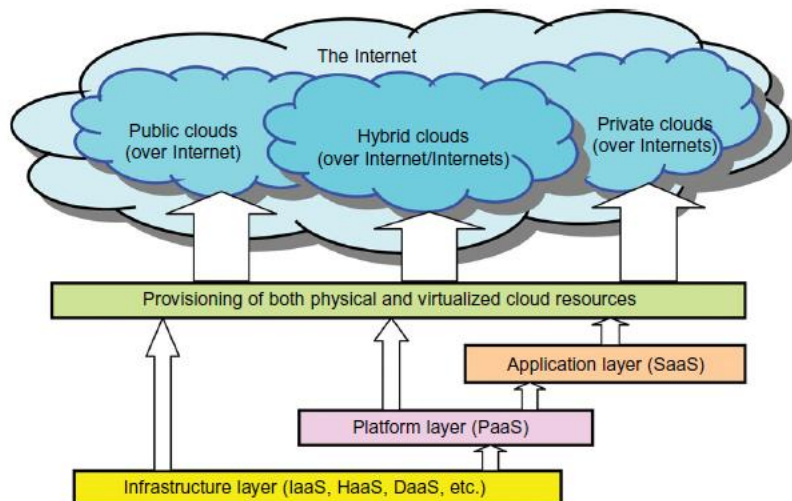


FIGURE 4.15

Layered architectural development of the cloud platform for IaaS, PaaS, and SaaS applications over the Internet.

- **Service-oriented architecture (SOA):** Enhances modular development and interoperability.
- **Multi-tenancy:** Supports multiple users in a shared environment while ensuring security.
- **Quality of Service (QoS) considerations:**
 - **Latency:** Minimizing delays in service response.
 - **Bandwidth:** Ensuring efficient data transmission.
 - **Performance optimization:** Using caching, load balancing, and autoscaling.

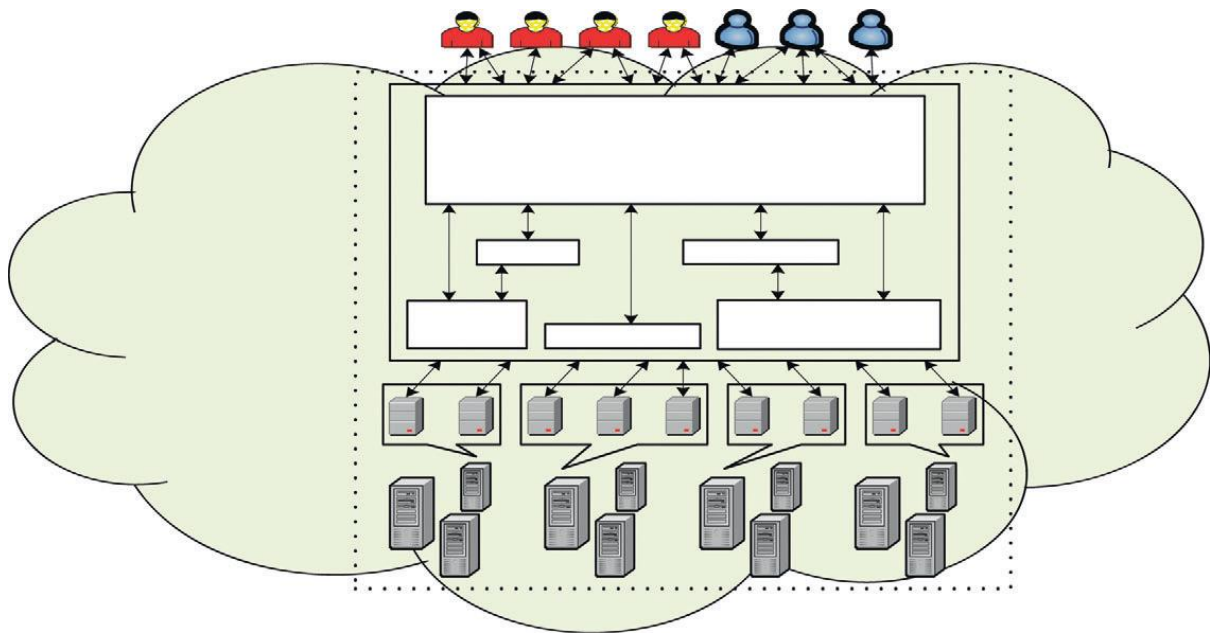
Market-Oriented Cloud Architecture

Cloud providers must ensure **QoS (Quality of Service)** to meet consumer demands, as defined in **SLAs (Service Level Agreements)**. Traditional **system-centric** resource management is insufficient; instead, **market-oriented** resource management is used to balance **supply and demand** of cloud resources.

The architecture includes:

- **Users/Brokers:** Submit service requests from anywhere.
- **SLA Resource Allocator:** Interfaces between cloud providers and users.
- **Service Request Examiner:** Interprets requests and prioritizes allocations.
- **Accounting Mechanism:** Tracks resource usage for billing and optimization.
- **VM Monitor:** Monitors VM availability and entitlements.
- **Dispatcher:** Assigns accepted service requests to VMs.
- **Service Request Monitor:** Tracks request execution progress.

Clouds dynamically allocate multiple **VMs on a single physical machine**, allowing **resource partitioning** and supporting different **OS environments**. The **Pricing Mechanism** determines costs based on factors like **time of submission (peak/off-peak)**, **fixed vs. dynamic rates**, and **resource availability**.



Quality of Service Factors

A cloud data center consists of multiple computing servers that allocate resources based on service demands. QoS (Quality of Service) parameters such as time, cost, reliability, and security are essential for commercial cloud offerings. Since business operations are dynamic, QoS requirements must adapt over time.

Key Components of Market-Oriented Cloud Architecture:

1. SLA Resource Allocator: Manages agreements between users and cloud providers.
2. Virtual & Physical Machines: Resources are allocated dynamically.
3. Pricing & Accounting: Tracks resource usage and charges users accordingly.
4. Dispatcher & Monitors: Assigns tasks to virtual machines (VMs) and tracks execution.
5. Service Request Examiner & Admission Control: Ensures efficient resource allocation without overloading.

Key Features:

- Customer-Driven Service Management: Adapts resources based on user requirements.
- Computational Risk Management: Identifies and mitigates risks in cloud operations.
- Autonomic Resource Management: Self-managing models adjust resource distribution dynamically.
- Dynamic SLA Negotiation: Supports real-time adjustments based on changing demands.

By leveraging VM technology, this architecture optimizes resource allocation, ensuring efficient, cost-effective, and reliable cloud services while adapting to real-time fluctuations in demand.

4.3.3 Virtualization Support and Disaster Recovery

System virtualization software plays a crucial role in cloud computing by simulating hardware execution and enabling the operation of multiple virtual environments on shared physical infrastructure.

Key Functions of Virtualization Software in Cloud Computing:

1. **Hardware Virtualization:** Allows unmodified operating systems to run on virtualized hardware, enabling legacy software support.
2. **Flexible Development & Deployment:**
 - Developers can use any OS or programming environment without infrastructure limitations.
 - The same environment is used for both development and deployment, reducing runtime errors.
3. **Multi-Tenant Isolation:**
 - Virtual Machines (VMs) securely separate users while maximizing resource utilization.
 - Unlike traditional cluster resource sharing, virtualization ensures better security and customization.
4. **Hosting Third-Party Applications:** Cloud platforms use VMs to host external applications, offering users complete control over their software stack.
5. **Scalability & Customization:** Users can configure VMs to suit their needs without interfering with other cloud tenants.

By leveraging virtualization, cloud computing platforms provide enhanced flexibility, security, and resource efficiency, making it easier to deploy, manage, and scale applications across diverse environments.

Virtualization Support in Public Clouds

Virtualization plays a significant role in cloud computing by abstracting physical resources, enhancing flexibility, and improving disaster recovery.

Comparison of Virtualization Support in Public Clouds

Cloud Provider	Virtualization Support	Key Feature
AWS (Amazon Web Services)	Extreme flexibility using VMs	Users can run custom applications
Microsoft Azure	Programming-level virtualization (.NET)	Supports application development
Google App Engine (GAE)	Application-level virtualization	Users can only build apps within Google's service framework

Each cloud provider supports different virtualization mechanisms based on their architecture and service models.

Storage Virtualization for Green Data Centers

- **Energy Consumption Concern:** IT power consumption in the U.S. has doubled, accounting for 3% of total energy use.
- **Corporate Response:** Fortune 500 companies are adopting energy-efficient strategies.
- **Impact of Virtualization:**
 - Reduces power usage by consolidating multiple workloads on fewer physical servers.
 - Lowers operational costs and improves energy efficiency.
 - Supports Green Computing initiatives.

Virtualization for IaaS (Infrastructure as a Service)

Cloud computing leverages VM technology for customizable environments, offering:

1. **Workload Consolidation:** Optimizes underutilized servers, reducing operational costs.
2. **Legacy Application Support:** Runs older software without compatibility issues.
3. **Security Enhancements:** Sandboxing prevents malware and application failures from affecting the entire system.
4. **Performance Isolation:** Enables cloud providers to ensure better QoS (Quality of Service) for users.

VM Cloning for Disaster Recovery

- **Traditional Recovery:** Slow and expensive, requiring reinstallation of hardware, OS, and software.
- **VM-Based Recovery:** Faster (40% of traditional recovery time), using snapshots for live migration.
- **Disaster Recovery (DR) Strategies:**
 1. **Data replication:** Synchronous and asynchronous replication techniques.
 2. **Snapshot backups:** Creating periodic copies of virtual machines and storage volumes.
 3. **Failover mechanisms:** Redundant systems automatically take over in case of failure.
 4. **Geographic redundancy:** Storing backup data across multiple cloud regions.

4.3.4 Architectural Design Challenges

- **Security risks:** Ensuring proper authentication, encryption, and compliance measures.
- **Scalability:** Handling increased demand while maintaining performance.
- **Interoperability:** Integrating different cloud service providers and platforms.
- **Data sovereignty and regulatory compliance:** Ensuring adherence to GDPR, HIPAA, and other regulations.

Challenges in Cloud Computing and Virtualization

Cloud computing has revolutionized IT infrastructure, but it presents multiple **challenges** related to availability, security, performance, and standardization.

Challenge 1: Service Availability and Data Lock-in

- **Single Point of Failure:** Cloud services managed by a single provider can be vulnerable to failures.
- **Multi-Cloud Solutions:** Using **multiple cloud providers** enhances **high availability (HA)**.
- **DDoS Attacks:** Can disrupt **SaaS providers**, requiring **rapid scale-ups** for defense.
- **Data Lock-in:**
 - APIs remain **proprietary**, restricting **data and application portability**.
 - **Standardizing APIs** would enable multi-cloud deployment and “surge computing” (using public clouds for extra workloads).

Challenge 2: Data Privacy and Security Concerns

- **Public Cloud Exposure:** Cloud platforms are open to security threats.
- **Encryption & Firewalls:** **Data encryption**, **VLANs**, and **network middleboxes** can mitigate risks.
- **Security Threats:**
 - **Traditional:** DoS attacks, malware, rootkits, spyware.
 - **Cloud-Specific:** Hypervisor malware, guest hopping, VM hijacking, **man-in-the-middle attacks** on VM migration.
- **Data Sovereignty Laws:** Many nations require **customer data storage within national boundaries**.

Challenge 3: Unpredictable Performance and Bottlenecks

- **I/O Bottlenecks:**
 - VMs efficiently share **CPU and memory**, but **I/O performance suffers**.
 - Example: **Amazon EC2**
 - **STREAM benchmark: 1,355 MB/sec bandwidth** needed.
 - **Disk writes: 55 MB/sec**, showing I/O limitations.
- **Data Placement Complexity:**
 - Data-intensive applications require **optimized data transfer strategies**.
 - Amazon **CloudFront** minimizes **data transport costs**.
- **Solutions:**
 - Improve **I/O virtualization architectures**.
 - Remove **bottleneck links and weak servers**.

Challenge 4: Distributed Storage and Software Bugs

- **Scalable Storage Needs:**
 - Cloud databases must scale **on-demand** while ensuring **high availability (HA)**.
 - **Distributed SANs** improve storage resilience.
- **Debugging Challenges:**
 - Large-scale cloud bugs **cannot be easily reproduced**.
 - **VM-based debugging** can help capture cloud runtime issues.
 - **Simulated debugging** (if well-designed) offers another solution.

Challenge 5: Cloud Scalability, Interoperability, and Standardization

- **Diverse Pricing Models:**
 - **GAE (Google App Engine)** charges **per cycle used**.
 - **AWS** charges **hourly for VM instances**, even if idle.
- **Standardization Efforts:**
 - **Open Virtualization Format (OVF):**
 - **Portable VM packaging** across platforms.
 - Allows **cross-hypervisor VM migration** (e.g., Intel ↔ AMD).
 - **Challenges:**
 - Need **hypervisor-agnostic VMs**.
 - **Support for legacy hardware** in cloud load balancing.

Challenge 6: Software Licensing and Reputation Sharing

- **Licensing Challenges:**
 - Open-source software dominates due to flexible licensing.
 - Commercial software vendors must adapt **pay-as-you-go** or **bulk-use** models.
- **Reputation Management:**
 - **Cloud-wide blacklisting** (e.g., spam-prevention services blacklisting EC2 IPs).
 - Potential **reputation-guarding services** to prevent cloud-wide bans.
- **Legal Liability:**
 - Cloud providers and customers debate **who holds legal liability** for failures or security breaches.
 - SLAs (Service Level Agreements) must clearly define liability.

4.4 Public Cloud Platforms: GAE, AWS, and Azure

4.4.1 Public Clouds and Service Offerings

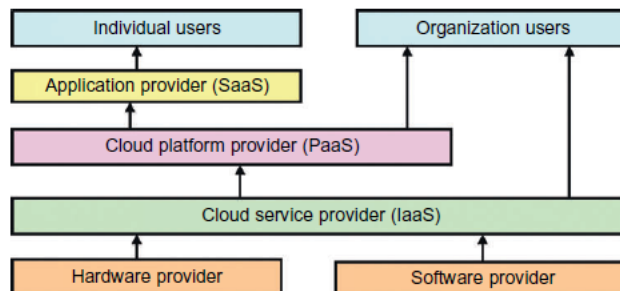


FIGURE 4.19

Roles of individual and organizational users and their interaction with cloud providers under various cloud service models.

- **Major Public Cloud Providers:**
 - **Amazon Web Services (AWS):** Leading IaaS and PaaS provider with services like EC2, S3, Lambda.
 - **Google Cloud Platform (GCP):** AI-driven cloud infrastructure with services like GKE, Cloud Run, and BigQuery.
 - **Microsoft Azure:** Strong hybrid cloud capabilities with services like Azure Virtual Machines, Azure Kubernetes Service, and Azure AI.
- **Common Service Offerings:**
 - Compute power (VMs, serverless functions, containers).
 - Storage solutions (object, block, file storage).
 - AI/ML services, big data analytics, and security tools.
 - Managed databases (SQL, NoSQL, cloud-native DBs).

Table 4.5 Five Major Cloud Platforms and Their Service Offerings [36]					
Model	IBM	Amazon	Google	Microsoft	Salesforce
PaaS	BlueCloud, WCA, RC2		App Engine (GAE)	Windows Azure	Force.com
IaaS	Ensembles	AWS		Windows Azure	
SaaS	Lotus Live		Gmail, Docs	.NET service, Dynamic CRM	Online CRM, Gifttag
Virtualization		OS and Xen	Application Container	OS level/ Hypel-V	
Service Offerings	SOA, B2, TSAM, RAD, Web 2.0	EC2, S3, SQS, SimpleDB	GFS, Chubby, BigTable, MapReduce	Live, SQL Hotmail	Apex, visual force, record security
Security Features	WebSphere2 and PowerVM tuned for protection	PKI, VPN, EBS to recover from failure	Chubby locks for security enforcement	Replicated data, rule-based access control	Admin./record security, uses metadata API
User Interfaces		EC2 command-line tools	Web-based admin. console	Windows Azure portal	
Web API	Yes	Yes	Yes	Yes	Yes
Programming Support	AMI		Python	.NET Framework	

Note: WCA: WebSphere CloudBurst Appliance; RC2: Research Compute Cloud; RAD: Rational Application Developer; SOA: Service-Oriented Architecture; TSAM: Tivoli Service Automation Manager; EC2: Elastic Compute Cloud; S3: Simple Storage Service; SQS: Simple Queue Service; GAE: Google App Engine; AWS: Amazon Web Services; SQL: Structured Query Language; EBS: Elastic Block Store; CRM: Consumer Relationship Management.

4.4.2 Google App Engine (GAE)

- **PaaS solution** that allows developers to build and deploy web applications without managing the infrastructure.
- **Key Features:**
 - Automatic scaling of applications based on demand.
 - Supports multiple programming languages (Python, Java, Go, Node.js, PHP, etc.).
 - Integrated development tools and monitoring.
 - Pay-as-you-go pricing model.

Google App Engine (GAE) Architecture

Google App Engine (GAE) is a **Platform-as-a-Service (PaaS)** that allows developers to build and deploy applications on Google's cloud infrastructure. It eliminates the need for server maintenance and provides scalable cloud services.

Key Building Blocks of Google Cloud Infrastructure

1. **Google File System (GFS):** Manages large-scale data storage.
2. **MapReduce:** Supports distributed computing for application development.
3. **BigTable:** Provides structured and semi-structured data storage.
4. **Chubby:** A distributed lock service for synchronization.

These components form the foundation of Google's cloud services, enabling seamless data storage, processing, and application management.

GAE Features and Architecture

- GAE **runs user programs** on Google's infrastructure, offering **scalability and maintenance-free operation** for developers.
- The **frontend** serves as an **application framework**, similar to **ASP, J2EE, and JSP**, supporting **Python and Java** for application development.

- The **GAE infrastructure** supports multiple clusters of servers running GFS, MapReduce, BigTable, and Chubby to provide efficient resource management.
- **Web-based interactions** allow both users and third-party developers to access Google applications seamlessly.

This architecture ensures that applications hosted on GAE can handle large-scale data and traffic efficiently while leveraging Google's high-performance cloud infrastructure.

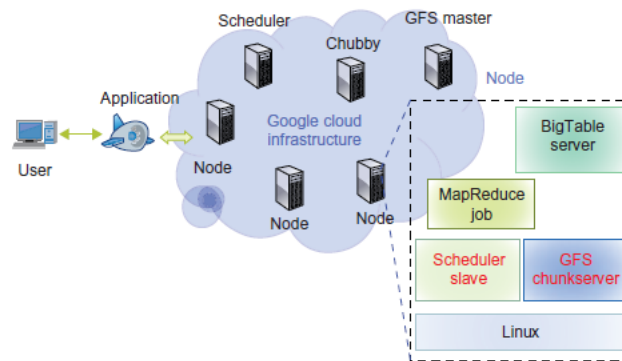


FIGURE 4.20

- Google cloud platform and major building blocks, the blocks shown are large clusters of low-cost servers.

Summary of GAE Components and Services

Google App Engine (GAE) is an **application development platform** rather than an infrastructure platform, providing tools and services for building and deploying cloud applications.

Major Components of GAE

1. **Datastore:**
 - Provides **object-oriented, distributed, structured data storage** based on **BigTable**.
 - Ensures **secure data management operations**.
2. **Application Runtime Environment:**
 - Supports **scalable web programming and execution**.
 - Compatible with **Python and Java** for development.
3. **Software Development Kit (SDK):**
 - Facilitates **local application development and testing**.
 - Allows developers to **upload application code** seamlessly.
4. **Administration Console:**
 - Simplifies the **management of application development cycles**.
 - Focuses on **software management** rather than physical infrastructure.
5. **GAE Web Service Infrastructure:**
 - Provides **special interfaces** for **efficient storage and network resource management**.

GAE Services and Usage

- **Free Access for Gmail Users:**
 - Users can register using a Gmail account and utilize **free services within a quota**.

- Exceeding the quota requires a **paid plan**.
- **Programming Language Support:**
 - GAE supports **Python, Ruby, and Java**, but **does not provide Infrastructure-as-a-Service (IaaS)**.
 - Unlike **AWS (which offers IaaS and PaaS)**, GAE is strictly a **PaaS model** for **deploying user-built applications**.
- **Comparison with Other Cloud Platforms:**
 - **GAE:** Focuses on **application hosting and development**.
 - **AWS:** Provides both **IaaS (infrastructure)** and **PaaS (platform)** services.
 - **Azure:** Supports a **similar model to GAE** but focuses on **.NET applications**.

GAE simplifies **cloud application deployment** by **abstracting infrastructure complexities**, making it an ideal choice for **developers focused on web and cloud-based applications**.

4.4.3 Amazon Web Services (AWS)

AWS Components and Services

Amazon Web Services (AWS) is a leading provider of public cloud services using the Infrastructure-as-a-Service (IaaS) model. AWS enables flexible and secure computing resource sharing through Virtual Machines (VMs).

Key AWS Components:

1. EC2 (Elastic Compute Cloud):
 - Provides virtualized computing platforms for running cloud applications.
2. S3 (Simple Storage Service):
 - Offers object-oriented storage for data management.
3. EBS (Elastic Block Service):
 - Supports block storage for traditional applications.
4. SQS (Simple Queue Service):
 - Ensures reliable message processing between processes.
 - Messages persist even if the receiver process is inactive.
5. ELB (Elastic Load Balancing):
 - Distributes incoming traffic across multiple EC2 instances.
 - Prevents overloading and manages fault tolerance.
6. CloudWatch:
 - Monitors AWS resources (e.g., CPU, disk I/O, network traffic).
 - Supports autoscaling and load balancing via ELB.
7. RDS (Relational Database Service):
 - Provides a fully managed relational database service.
8. Elastic MapReduce (EMR):
 - Equivalent to Hadoop on EC2 for big data processing.
9. AWS Import/Export:
 - Allows physical data transfer via shipping disks for high-bandwidth data migration.
10. CloudFront:
 - Implements content distribution for faster web application delivery.
11. FPS (Flexible Payment Service):
 - Enables developers to charge AWS users for paid services.

12. FWS (Fulfillment Web Service):

- Provides order fulfillment services through Amazon's logistics.

13. MPI Clusters & Cluster Compute Instances (July 2010):

- AWS introduced hardware-assisted virtualization for high-performance computing (HPC).
- Uses EBS-based booting instead of para-virtualization.

AWS provides a flexible cloud computing environment, making it ideal for small and medium businesses looking to scale their operations while serving large numbers of users efficiently.

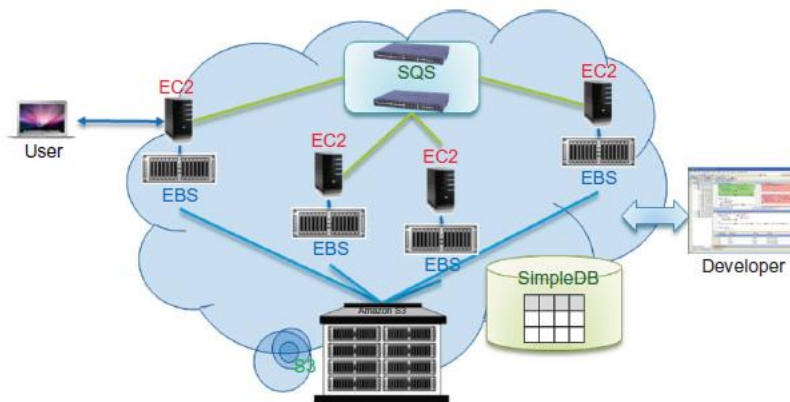


FIGURE 4.21

Amazon cloud computing infrastructure (Key services are identified here; many more are listed in Table 4.6).

- **Market leader in cloud services**, providing comprehensive solutions for enterprises.
- **Advantages:**
 - Global infrastructure with multiple availability zones.
 - Broad range of services catering to various industry needs.
 - Strong security and compliance measures.

4.4.4 Microsoft Azure

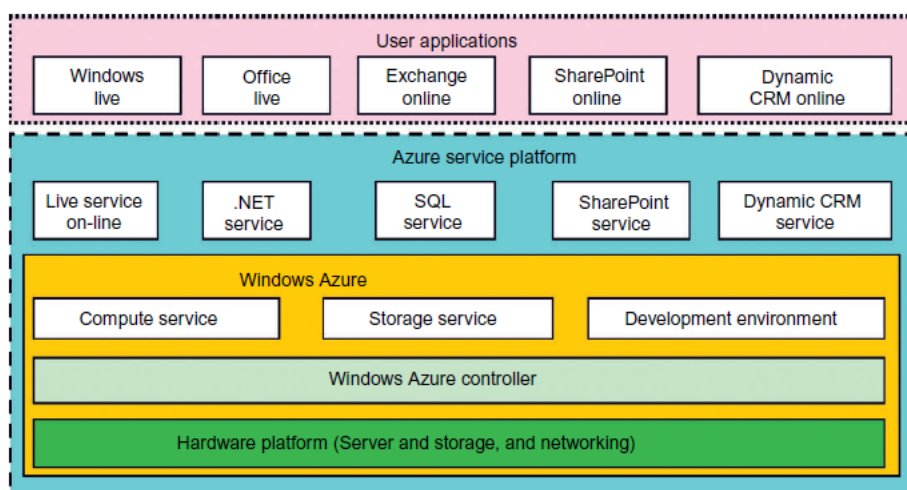


FIGURE 4.22

Microsoft Windows Azure platform for cloud computing.

Microsoft Azure Cloud Platform

Launch

Year:

2008

Objective: To address cloud computing challenges using Microsoft data centers.

Architecture: Built on Windows OS and Microsoft virtualization technology.

Key Components of Azure:

1. Windows Azure:
 - Provides a cloud platform based on Windows OS.
 - Manages VMs, storage, and network resources in data centers.
2. Core Azure Services:
 - Live Service: Supports Microsoft Live applications and multi-machine data access.
 - .NET Service: Enables local application development and cloud execution.
 - SQL Azure: Provides cloud-based relational database services with SQL Server.
 - SharePoint Service: Helps in scalable business application development.
 - Dynamic CRM Service: Supports business management applications (finance, marketing, sales).
3. Integration with Microsoft Products:
 - Azure services work seamlessly with Windows Live, Office Live, Exchange Online, SharePoint Online, and Dynamic CRM Online.
4. Communication Protocols:
 - Uses SOAP and REST for web-based communication and third-party cloud integration.
5. Azure Development Kit (SDK):
 - Allows developers to run a local version of Azure, develop applications, and debug them on Windows hosts.

Key Differentiators of Azure:

- Seamless integration with Microsoft products.
- Strong support for enterprise applications (SharePoint, CRM, SQL).
- Developer-friendly SDK for local development and testing.
- Hybrid cloud capabilities with third-party cloud integration.

Microsoft Azure offers a powerful and scalable cloud solution, making it a preferred choice for enterprises that rely on Microsoft technologies.

INTER-CLOUD RESOURCE MANAGEMENT

Extended Cloud Computing Services : Cloud computing is structured into different service layers, each catering to distinct functionalities and user needs. These services range from hardware and networking to software applications and runtime support.

Cloud application (SaaS)			Concur, RightNOW, Teleo, Kenexa, Webex, Blackbaud, salesforce.com, Netsuite, Kenexa, etc.
Cloud software environment (PaaS)			Force.com, App Engine, Facebook, MS Azure, NetSuite, IBM BlueCloud, SGI Cyclone, eBay
Cloud software infrastructure			Amazon AWS, OpSource Cloud, IBM Ensembles, Rackspace cloud, Windows Azure, HP, Banknorth
Computational resources (IaaS)	Storage (DaaS)	Communications (Caas)	
Collocation cloud services (LaaS)			Savvis, Internap, NTTCommunications, Digital Realty Trust, 365 Main
Network cloud services (NaaS)			Owest, AT&T, AboveNet
Hardware/Virtualization cloud services (HaaS)			VMware, Intel, IBM, XenEnterprise

FIGURE 4.23

A stack of six layers of cloud services and their providers.

Six Layers of Cloud Services

1. Hardware as a Service (HaaS)
 - Provides physical infrastructure (servers, storage, networking).
 - Examples: VMware, Intel, IBM, XenEnterprise.
2. Network as a Service (NaaS)
 - Manages network connectivity and virtual LANs.
 - Examples: AT&T, Qwest, AboveNet.
3. Location as a Service (LaaS)
 - Offers data center collocation services (housing, power, security).
 - Sometimes referred to as Security as a Service (SaaS).
 - Examples: Savvis, Internap, Digital Realty Trust.
4. Infrastructure as a Service (IaaS)
 - Provides compute, storage, and networking resources on demand.
 - Examples: Amazon AWS, Microsoft Azure, Rackspace Cloud, IBM Ensembles.
5. Platform as a Service (PaaS)
 - Offers development environments, frameworks, and APIs.
 - Examples: Google App Engine, Force.com, Microsoft Azure, IBM BlueCloud.
6. Software as a Service (SaaS)
 - Delivers fully managed applications over the internet.
 - Examples: Salesforce, Webex, Concur, RightNOW, Teleo, Netsuite.

Roles of Cloud Service Players

Cloud Players	IaaS Role	PaaS Role	SaaS Role
IT Administrators/ Cloud Providers	Monitor SLAs	Monitor SLAs & enable platforms	Monitor SLAs & deploy software
Software Developers/ Vendors	Deploy & store data	Enable platforms via APIs	Develop & deploy software
End Users/ Business Users	Deploy & store data	Develop & test web applications	Use business software

Trends in Cloud Services

- SaaS Expansion: Initially led by CRM applications, now extends to HR, finance, and distributed collaboration.
- PaaS Adoption: Increasing with platforms like Google App Engine, Salesforce, Facebook, Microsoft Azure.
- IaaS Growth: Driven by AWS, Azure, and Rackspace, providing scalable compute and storage.
- Vertical Cloud Services: Multiple integrated services working together in a cloud mashup.

Cloud Software Stack

Cloud platforms are designed with high throughput, fault tolerance, and high availability (HA). The software stack consists of:

1. Virtualization Layer – Flexible infrastructure using VMs.
2. Storage Layer – Manages large-scale data storage.
3. Database Layer – Supports structured and unstructured data.
4. Compute Layer – Executes cloud applications.
5. Application Layer – Provides user-facing software services.

Runtime Support & Cloud Scheduling

- Cluster Monitoring: Tracks the status of cloud resources.
- Job Management System: Schedules tasks efficiently across cloud nodes.
- MapReduce Scheduling: Specialized for big data processing in distributed cloud environments.

Advantages of SaaS Model

- No upfront costs for hardware and software.
- Lower operational costs for providers compared to traditional hosting.
- Cloud storage options (vendor-specific or public cloud).
- Scalability and flexibility for businesses and developers.

Summary of Resource Provisioning and Platform Deployment in Cloud Computing

The emergence of cloud computing has led to significant changes in software and hardware architecture, emphasizing processor cores, VM instances, and parallelism at the cluster node level. Effective resource provisioning and platform deployment are essential for optimizing cloud infrastructure performance.

1. Provisioning of Compute Resources (VMs)

Cloud providers allocate resources through **Service Level Agreements (SLAs)**, ensuring CPU, memory, and bandwidth availability for a preset period. Balancing **underprovisioning** (risking SLA violations) and **overprovisioning** (causing resource waste) is a key challenge. Efficient provisioning involves **VM installation, live migration, and failure recovery**. Examples include **Amazon EC2**, IBM's **Blue Cloud**, and Microsoft **Azure**, all of which rely on virtualization technologies.

2. Resource Provisioning Methods

Three primary resource provisioning methods exist:

- **Demand-Driven Provisioning:** Adjusts resources based on utilization thresholds (e.g., Amazon EC2 auto-scaling). It is simple but ineffective for abrupt workload changes.

- **Event-Driven Provisioning:** Allocates resources based on predicted workload spikes during specific events (e.g., seasonal sales). This method minimizes Quality of Service (QoS) loss if predictions are accurate.
- **Popularity-Driven Provisioning:** Allocates resources based on **Internet search trends**. It anticipates traffic surges but may waste resources if popularity forecasts are incorrect.

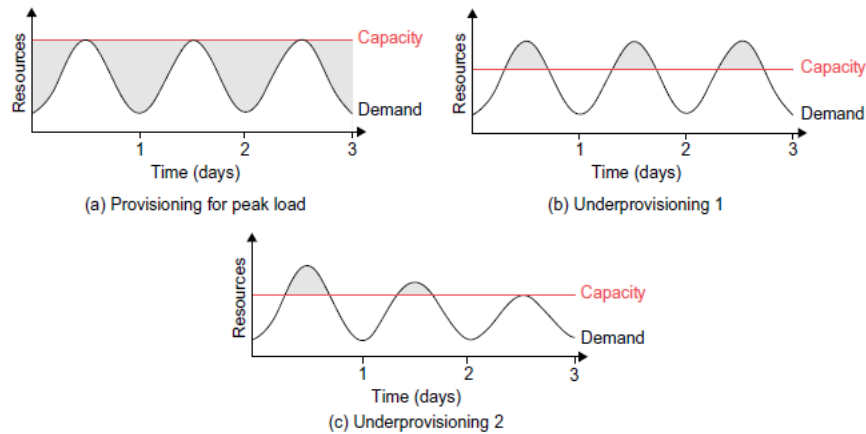


FIGURE 4.24

Three cases of cloud resource provisioning without elasticity: (a) heavy waste due to overprovisioning, (b) underprovisioning and (c) under- and then overprovisioning.

-

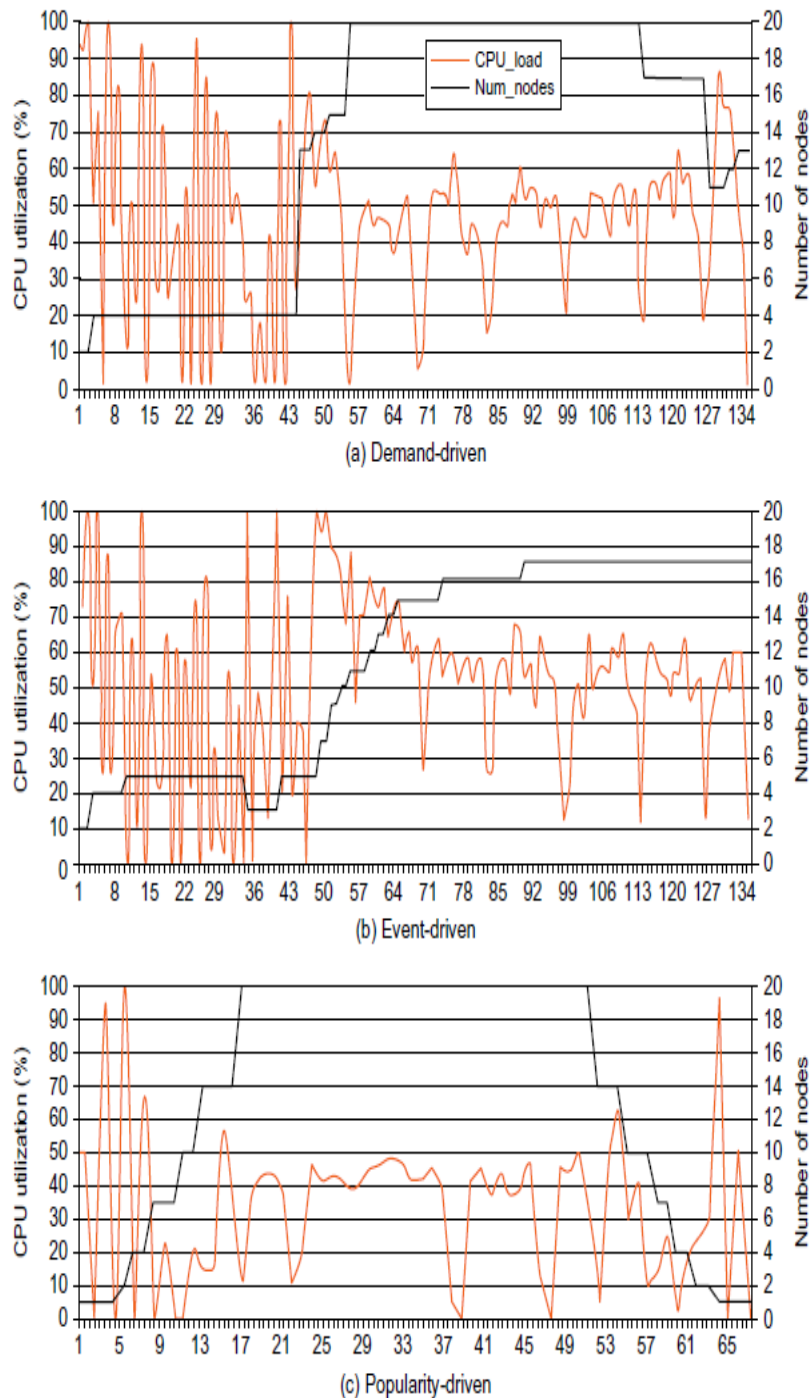


FIGURE 4.25

EC2 performance results on the AWS EC2 platform, collected from experiments at the University of Southern California using three resource provisioning methods.

3. Dynamic Resource Deployment

Dynamic provisioning allows cloud systems to scale across multiple resource sites. The **InterGrid infrastructure**, developed by Melbourne University, enables cross-grid resource allocation via **InterGrid Gateways (IGGs)**, allowing interaction between local clusters and external cloud providers.

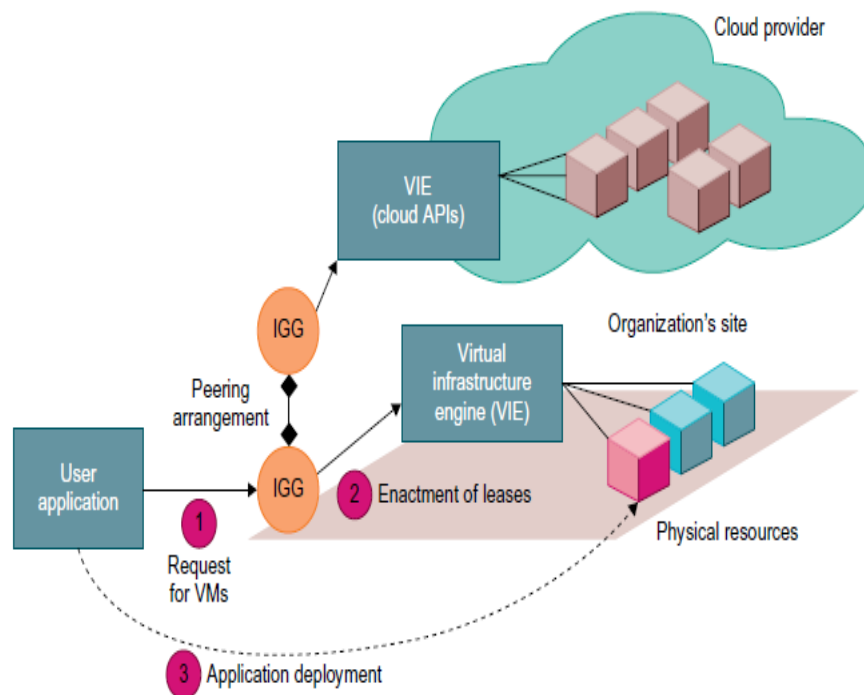


FIGURE 4.26

Cloud resource deployment using an IGG (intergrid gateway) to allocate the VMs from a Local cluster to interact with the IGG of a public cloud provider.

4. Provisioning of Storage Resources

Cloud storage is built on **physical or virtual servers** and is essential for scalable applications like email systems and web search engines. Future storage solutions will likely integrate **solid-state drives (SSDs)** for higher performance. Common cloud storage systems include:

- **Google File System (GFS)** – High bandwidth for continuous access.
- **Hadoop Distributed File System (HDFS)** – Open-source alternative to GFS.
- **Amazon S3 & EBS** – Remote data storage and virtual disk services.

Cloud databases (e.g., **Google BigTable**, **Amazon SimpleDB**, **Microsoft Azure SQL Service**) enable structured and semi-structured data management, allowing developers to build applications efficiently.

Conclusion

Efficient cloud resource provisioning is critical for balancing performance, cost, and scalability. By leveraging various provisioning strategies and storage solutions, cloud platforms optimize their computing environments while maintaining service quality and efficiency.

Virtual Machine Creation and Management

This section discusses key aspects of cloud infrastructure management, including resource management for independent services, execution of third-party applications, and various components involved in virtual machine (VM) management.

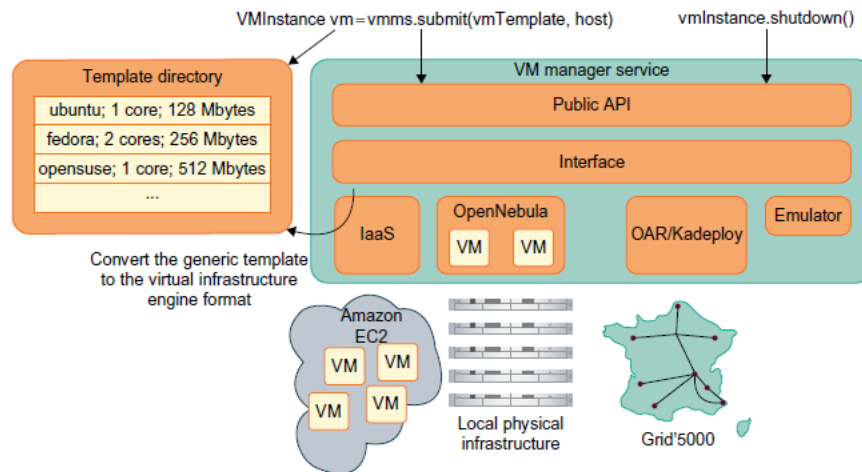


FIGURE 4.27

Interactions among VM managers for cloud creation and management; the manager provides a public API for users to submit and control the VMs.

1. **Independent Service Management:** Cloud infrastructures provide APIs for running independent services. Amazon's Simple Queue Service (SQS) facilitates reliable communication between service providers. This enables multiple cloud applications to run simultaneously.
2. **Running Third-Party Applications:** Cloud platforms support third-party applications through web services, allowing developers to build applications with APIs instead of traditional runtime libraries. Examples include Google App Engine (GAE), Microsoft Azure, and IBM's WebSphere.
3. **Virtual Machine Manager (VMM):** The VMM links gateways to resources, managing VMs using virtualization technologies. It supports multiple environments, including OpenNebula, Amazon EC2, and Grid'5000. The manager enables remote VM deployment using hypervisors like Xen.
4. **Virtual Machine Templates:** VM templates define configuration parameters such as processor allocation, memory, disk image, OS kernel, and pricing. Administrators maintain a repository of templates to ensure consistency across the infrastructure.
5. **Distributed VM Management:** The InterGrid platform uses distributed VM management, allowing gateways to request, allocate, and redirect VM resources across multiple sites. A peering policy ensures workload balancing across cloud sites.
6. **InterCloud Resource Exchange:** To meet global demand, cloud providers establish data centers in different geographical locations. The Melbourne group's **InterCloud** architecture enables seamless integration of multiple cloud providers for dynamic resource allocation and load balancing.
7. **Cloud Exchange (CEx):** The CEx acts as a marketplace for cloud resources, allowing providers and consumers to negotiate service agreements based on economic models such as auctions and commodity markets. It ensures QoS-driven service delivery and secure financial transactions through SLA-based contracts.

This emphasizes the need for efficient VM management, workload balancing, and inter-cloud resource sharing to optimize cloud computing performance and scalability.