FUTURE VISION BIE

One Stop for All Study Materials

& Lab Programs

Future Vision

By K B Hemanth Raj

Scan the QR Code to Visit the Web Page

Or

Visit : https://hemanthrajhemu.github.io

Gain Access to All Study Materials according to VTU,
CSE – Computer Science Engineering,
ISE – Information Science Engineering,
ECE - Electronics and Communication Engineering
& MORE...

MODULE – I
INTRODUCTION, VIRTUALIZATION
Chapters – 1 & 3.

1. Introduction.

Computing is being transformed into a model consisting of services that are commoditized and delivered in a manner similar to utilities such as water, electricity, gas, and telephony. In such a model, users access services based on their requirements, regardless of where the services are hosted.

Cloud computing is a technological advancement that focuses on the way we design computing systems, develop applications, and leverage existing services for building software. It is based on the concept of dynamic provisioning, which is applied not only to services but also to compute capability, storage, networking, and information technology (IT) infrastructure in general.

1.1 Cloud computing at a glance

In 1969, Leonard Kleinrock, one of the chief scientists of the original Advanced Research Projects Agency Network (ARPANET), which seeded the Internet, said:

"As of now, computer networks are still in their infancy, but as they grow up and become sophisti-cated, we will probably see the spread of 'computer utilities' which, like present electric and telephone utilities, will service individual homes and offices across the country."

Cloud computing allows renting infrastructure, runtime environments, and services on a pay-per-use basis. This principle finds several practical applications and then gives different images of cloud computing to different people. Chief information and technology officers of large enterprises see opportunities for scaling their infrastructure on demand and sizing it according to their business needs. End users leveraging cloud computing services can access their documents and data anytime,anywhere, and from any device connected to the Internet. Many other points of view exist.

One of the most diffuse views of cloud computing can be summarized as follows:

"I don't care where my servers are, who manages them, where my documents are stored, or where my applications are hosted. I just want them always available and access them from any device connected through Internet. And I am willing to pay for this service for as a long as I need it."

1.1.1 The vision of cloud computing

Cloud computing allows anyone with a credit card to provision virtual hardware, runtime environments, and services. These are used for as long as needed, with no up-front commitments required.

The entire stack of a computing system is transformed into a collection of utilities, which can be provisioned and composed together to deploy systems in hours rather than days and with virtually no maintenance costs.

The long-term vision of cloud computing is that IT services are traded as utilities in an open market, without technological and legal barriers. In this cloud marketplace, cloud service providers and consumers, trading cloud services as utilities, play a central role.

Many of the technological elements contributing to this vision already exist. Different stake-holders leverage clouds for a variety of services. The need for ubiquitous storage and compute power on demand is the most common reason to consider cloud computing. A scalable runtime for applications is an attractive option for application and system developers that do not have infrastructure or cannot afford any further expansion of existing infrastructure.

This approach provides opportunities for optimizing datacenter facilities and fully utilizing their capabilities to serve multiple users. This consolidation model will reduce the waste of energy and carbon emissions, thus contributing to a greener IT on one end and increasing revenue on the other end.

FIGURE 1.1 : Cloud computing vision.

1.1.2 Defining a cloud

The term cloud has historically been used in the telecommunications industry as an abstraction of the network in system diagrams. It then became the symbol of the most popular computer network, the Internet. This meaning also applies to cloud computing, which refers to an Internet-centric way of computing. The Internet plays a fundamental role in cloud computing, since it represents either the medium or the platform through which many cloud computing services are delivered and made accessible. This aspect is also reflected in the definition given by Armbrust et al.:

"Cloud computing refers to both the applications delivered as services over the Internet and the hardware and system software in the datacenters that provide those services."

The notion of multiple parties using a shared cloud computing environment is highlighted in a definition proposed by the U.S. National Institute of Standards and Technology (NIST):

"Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."

According to Reese,we can define three criteria to discriminate whether a service is delivered in the cloud computing style:

- The service is accessible via a Web browser (nonproprietary) or a Web services application programming interface (API).
- Zero capital expenditure is necessary to get started.
- You pay only for what you use as you use it.

1.1.3 A closer look

Cloud computing is helping enterprises, governments, public and private institutions, and research organizations shape more effective and demand-driven computing systems. Access to, as well as integration of, cloud computing resources and systems is now as easy as performing a credit card transaction over the Internet. Practical examples of such systems exist across all market segments:

- Large enterprises can offload some of their activities to cloud-based systems.
- Small enterprises and start-ups can afford to translate their ideas into business results more quickly, without excessive up-front costs.
- System developers can concentrate on the business logic rather than dealing with the complexity of infrastructure management and scalability.
- End users can have their documents accessible from everywhere and any device.

Cloud computing does not only contribute with the opportunity of easily accessing IT services on demand, it also introduces a new way of thinking about IT services and resources: as utilities. A bird's-eye view of a cloud computing environment is shown in Figure 1.3.

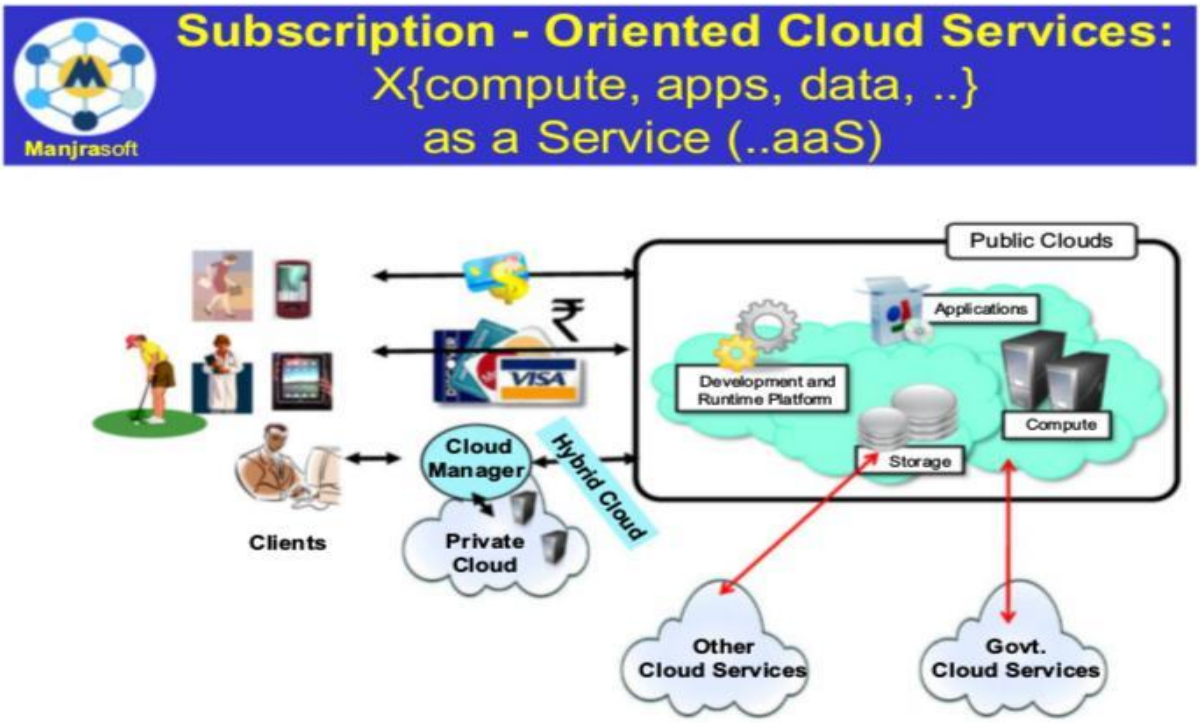

FIGURE 1.3

A bird's-eye view of cloud computing.

1.1.4 The cloud computing reference model

A fundamental characteristic of cloud computing is the capability to deliver, on demand, a variety of

IT services that are quite diverse from each other. cloud computing services offerings into three major categories: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), and Software-as-a-Service (SaaS). These categories are related to each other as described in Figure 1.5, which provides an organic view of cloud computing.
At the base of the stack, Infrastructure- as-a-Service solutions deliver infrastructure on demand in the form of virtual hardware, storage, and networking. Virtual hardware is utilized to provide compute on demand in the form of virtual machine instances.
Platform-as-a-Service solutions are the next step in the stack. They deliver scalable and elastic runtime environments on demand and host the execution of applications. These services are backed by a core middleware platform that is responsible for creating the abstract environment where applications are deployed and executed.
At the top of the stack, Software-as-a-Service solutions provide applications and services on demand. Most of the common functionalities of desktop applications. Each layer provides a different service to users. IaaS solutions are sought by users who want to leverage cloud computing from building dynamically scalable computing systems requiring a spe- cific software stack. IaaS services are therefore used to develop scalable Websites or for back- ground processing.

1.1.5 Characteristics and benefits

Cloud computing has some interesting characteristics that bring benefits to both cloud service consumers (CSCs) and cloud service providers (CSPs). These characteristics are:

- No up-front commitments
- On-demand access
- Nice pricing
- Simplified application acceleration and scalability
- Efficient resource allocation
- Energy efficiency
- Seamless creation and use of third-party services

The most evident benefit from the use of cloud computing systems and technologies is the increased economical return due to the reduced maintenance costs and operational costs related to IT software and infrastructure.
This is mainly because IT assets, namely software and infrastructure, are turned into utility costs, which are paid for as long as they are used, not paid for up front.
IT infrastructure and software generated capital costs, since they were paid up front so that business start-ups could afford a computing infrastructure, enabling the business activities of the organization. The revenue of the business is then utilized to compensate over time for these costs.
End users can benefit from cloud computing by having their data and the capability of operating on it always available, from anywhere, at any time, and through multiple devices. Information and services stored in the cloud are exposed to users by Web-based interfaces that make them accessible from portable devices as well as desktops at home.

1.1.6 Challenges ahead

New, interesting problems and challenges are regularly being posed to the cloud community, including IT practitioners, managers, governments, and regulators. Technical challenges also arise for cloud service providers for the management of large computing infrastructures and the use of virtualization technologies on top of them.
Security in terms of confidentiality, secrecy, and protection of data in a cloud environment is

another important challenge. Organizations do not own the infrastructure they use to process data and store information. This condition poses challenges for confidential data, which organizations cannot afford to reveal.
Legal issues may also arise. These are specifically tied to the ubiquitous nature of cloud computing, which spreads computing infrastructure across diverse geographical locations. Different legislation about privacy in different countries may potentially create disputes as to the rights that third parties (including government agencies) have to your data.

1.2 Historical developments

The idea of renting computing services by leveraging large distributed computing facilities has been around for long time. It dates back to the days of the mainframes in the early 1950s. Figure 1.6 provides an overview of the evolution of the distributed computing technologies that have influenced cloud computing. In tracking the historical evolution, we briefly review five core technologies that played an important role in the realization of cloud computing. These technologies are distributed systems, virtualization, Web 2.0, service orientation, and utility computing.

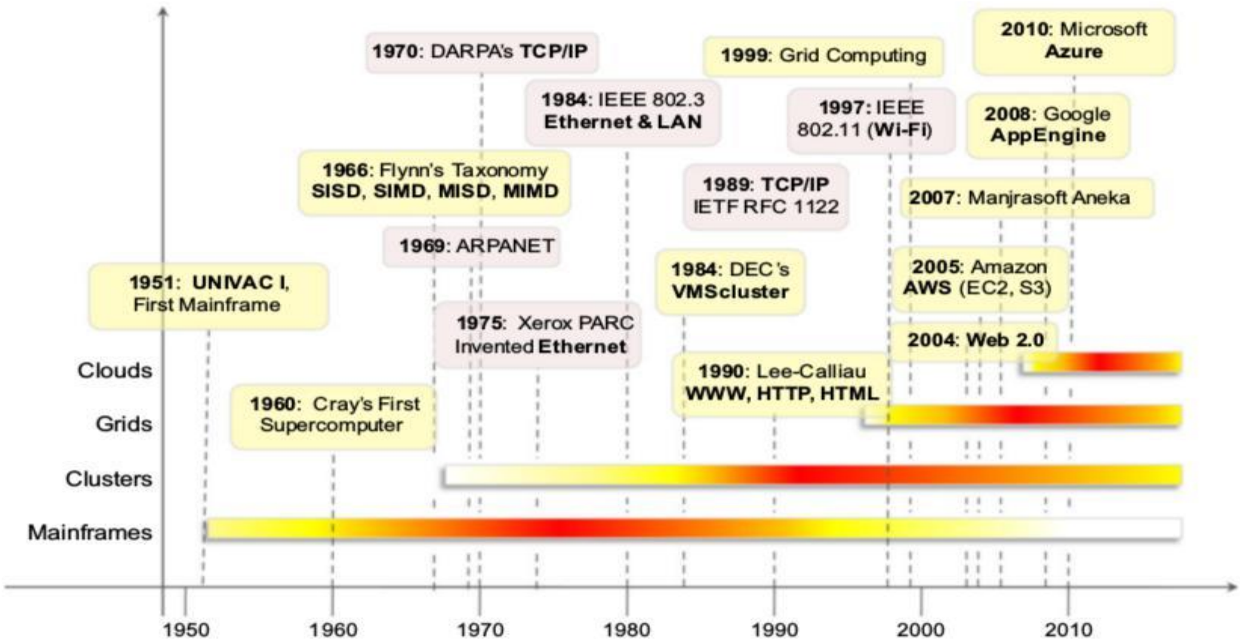

FIGURE 1.6

The evolution of distributed computing technologies, 1950s–2010s.

1.2.1 Distributed systems

Clouds are essentially large distributed computing facilities that make available their services to third parties on demand. As a reference, we consider the characterization of a distributed system proposed by Tanenbaum et al.:

“A distributed system is a collection of independent computers that appears to its users as a single coherent system.”

Three major milestones have led to cloud computing: mainframe computing, cluster computing, and grid computing.

Mainframes. These were the first examples of large computational facilities leveraging multiple processing units. Mainframes were powerful, highly reliable computers specialized for large data movement and massive input/output (I/O) operations. They were mostly used by large organizations for bulk data processing tasks such as online transactions, enterprise resource planning, and other operations involving the processing of significant amounts of data.

Clusters. Cluster computing started as a low-cost alternative to the use of mainframes and supercomputers. The technology advancement that created faster and more powerful mainframes and supercomputers eventually generated an increased availability of cheap commodity machines as a side effect. These machines could then be connected by a high-bandwidth network and controlled by specific software tools that manage them as a single system. Starting in the 1980s.

Cluster technology contributed considerably to the evolution of tools and frameworks for distributed computing, including Condor, Parallel Virtual Machine (PVM), and Message Passing Interface (MPI).

Grid computing appeared in the early 1990s as an evolution of cluster computing. In an analogy to the power grid, grid computing proposed a new approach to access large computational power, hugeBstorage facilities, and a variety of services.

A computing grid was a dynamic aggregation of heterogeneous computing nodes, and its scale was nationwide or even worldwide. Several developments made possible the diffusion of computing grids:

(a) clusters became quite common resources;
(b) they were often underutilized;
(c) new problems were requiring computational power that went beyond the capability of single clusters; and
(d) the improvements in networking and the diffusion of the Internet made possible long-distance, high-bandwidth connectivity.

1.2.2 Virtualization

Virtualization is another core technology for cloud computing. It encompasses a collection of solutions allowing the abstraction of some of the fundamental elements for computing, such as hardware, runtime environments, storage, and networking. Virtualization has been around for more than 40 years, but its application has always been limited by technologies that did not allow an efficient use of virtualization solutions.

Virtualization is essentially a technology that allows creation of different computing environments. These environments are called virtual because they simulate the interface that is expected by a guest. The most common example of virtualization is hardware virtualization.

Virtualization technologies are also used to replicate runtime environments for programs. Applications in the case of process virtual machines (which include the foundation of technologies such as Java or .NET), instead of being executed by the operating system, are run by a specific program called a virtual machine. This technique allows isolating the execution of applications and providing a finer control on the resource they access.

1.2.3 Web 2.0

The Web is the primary interface through which cloud computing delivers its services. At present, the Web encompasses a set of technologies and services that facilitate interactive information sharing, collaboration, user-centered design, and application composition. This evolution has transformed the Web into a rich platform for application development and is known as Web 2.0. This term captures a new way in which developers architect applications and deliver services through the Internet and provides new experience for users of these applications and services.

Web 2.0 brings interactivity and flexibility into Web pages, providing enhanced user experience by gaining Web-based access to all the functions that are normally found in desktop applications. These capabilities are obtained by integrating a collection of standards and technologies such as XML, Asynchronous JavaScript and XML (AJAX), Web Services, and others. These technologies allow us to build applications leveraging the contribution of users, who now become providers of content.

Web 2.0 applications are extremely dynamic: they improve continuously, and new updates and features are integrated at a constant rate by following the usage trend of the community. There is no need to deploy new software releases on the installed base at the client side.

Web 2.0 applications aim to leverage the "long tail" of Internet users by making themselves available to everyone in terms of either media accessibility or affordability.

Examples of Web 2.0 applications are Google Documents, Google Maps, Flickr, Facebook, Twitter, YouTube, de.li.cious, Blogger, and Wikipedia. In particular, social networking Websites take the biggest advantage of Web 2.0. The level of interaction in Websites such as Facebook or Flickr would not have been possible without the support of AJAX, Really Simple Syndication (RSS), and other tools that make the user experience incredibly interactive.

This idea of the Web as a transport that enables and enhances interaction was introduced in 1999 by Darcy DiNucci 5 and started to become fully realized in 2004. Today it is a mature platform for supporting the needs of cloud computing, which strongly leverages Web 2.0. Applications and frameworks for delivering rich Internet applications (RIAs) are fundamental for making cloud services accessible to the wider public.

1.2.4 Service-oriented computing

Service orientation is the core reference model for cloud computing systems. This approach adopts the concept of services as the main building blocks of application and system development. Service-oriented computing (SOC) supports the development of rapid, low-cost, flexible, interoperable, and evolvable applications and systems.

A service is an abstraction representing a self-describing and platform-agnostic component that can perform any function—anything from a simple function to a complex business process.

A service is supposed to be loosely coupled, reusable, programming language independent, and location transparent. Loose coupling allows services to serve different scenarios more easily and makes them reusable. Independence from a specific platform increases services accessibility. Thus, a wider range of clients, which can look up services in global registries and consume them in a location-transparent manner, can be served.

Service-oriented computing introduces and diffuses two important concepts, which are also fundamental to cloud computing: quality of service (QoS) and Software-as-a-Service (SaaS).

- Quality of service (QoS) identifies a set of functional and nonfunctional attributes that can be used to evaluate the behavior of a service from different perspectives. These could be performance metrics such as response time, or security attributes, transactional integrity, reliability, scalability, and availability.
- The concept of Software-as-a-Service introduces a new delivery model for applications. The term has been inherited from the world of application service providers (ASPs), which deliver software services-based solutions across the wide area network from a central datacenter and make them available on a subscription or rental basis.

1.2.5 Utility-oriented computing

Utility computing is a vision of computing that defines a service-provisioning model for compute services in which resources such as storage, compute power, applications, and infrastructure are

packaged and offered on a pay-per-use basis. The idea of providing computing as a utility like natural gas, water, power, and telephone connection has a long history but has become a reality today with the advent of cloud computing.
The American scientist John McCarthy, who, in a speech for the Massachusetts Institute of Technology (MIT) centennial in 1961, observed:
"If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility, just as the telephone system is a public utility . . . The computer utility could become the basis of a new and important industry."

The first traces of this service-provisioning model can be found in the mainframe era. IBM and other mainframe providers offered mainframe power to organizations such as banks and government agencies throughout their datacenters.
From an application and system development perspective, service-oriented computing and service-oriented architectures (SOAs) introduced the idea of leveraging external services for performing a specific task within a software system.

1.3 Building cloud computing environments

The creation of cloud computing environments encompasses both the development of applications and systems that leverage cloud computing solutions and the creation of frameworks, platforms, and infrastructures delivering cloud computing services.

1.3.1 Application development

Applications that leverage cloud computing benefit from its capability to dynamically scale on demand. One class of applications that takes the biggest advantage of this feature is that of Web applications. Their performance is mostly influenced by the workload generated by varying user demands. With the diffusion of Web 2.0 technologies, the Web has become a platform for develop-ing rich and complex applications, including enterprise applications that now leverage the Internet as the preferred channel for service delivery and user interaction.
Another class of applications that can potentially gain considerable advantage by leveraging cloud computing is represented by resource-intensive applications. These can be either data-intensive or compute- intensive applications. In both cases, considerable amounts of resources are required to complete execution in a reasonable timeframe.
Cloud computing provides a solution for on-demand and dynamic scaling across the entire stack of computing.
This is achieved by
(a) providing methods for renting compute power, storage, and networking;
(b) offering runtime environments designed for scalability and dynamic sizing; and
(c) providing application services that mimic the behavior of desktop applications but that are completely hosted and managed on the provider side.

1.3.2 Infrastructure and system development

Distributed computing, virtualization, service orientation, and Web 2.0 form the core technologies enabling the provisioning of cloud services from anywhere on the globe. Developing applications and systems that leverage the cloud requires knowledge across all these technologies.
Distributed computing is a foundational model for cloud computing because cloud systems are distributed systems. Besides administrative tasks mostly connected to the accessibility of resources in the cloud, the extreme dynamism of cloud systems—where new nodes and services are provisioned on demand—constitutes the major challenge for engineers and developers.

Web 2.0 technologies constitute the interface through which cloud computing services are delivered, managed, and provisioned.
Cloud computing is often summarized with the acronym XaaS—Everything-as-a-Service—that clearly underlines the central role of service orientation.
Virtualization is another element that plays a fundamental role in cloud computing. This technology is a core feature of the infrastructure used by cloud providers.

1.3.3 Computing platforms and technologies

Development of a cloud computing application happens by leveraging platforms and frameworks that provide different types of services, from the bare-metal infrastructure to customizable applications serving specific purposes.

1 Amazon web services (AWS)
2 **Google AppEngine**
3 Microsoft Azure
4 Hadoop
5 Force.com and Salesforce.com
6 Manjrasoft Aneka

1 Amazon web services (AWS)

AWS offers comprehensive cloud IaaS services ranging from virtual compute, storage, and networking to complete computing stacks. AWS is mostly known for its compute and storage-on-demand services, namely Elastic Compute Cloud (EC2) and Simple Storage Service (S3). EC2 provides users with customizable virtual hardware that can be used as the base infrastructure for deploying computing systems on the cloud. It is possible to choose from a large variety of virtual hardware configurations, including GPU and cluster instances. EC2 also provides the capability to save a specific running instance as an image, thus allowing users to create their own templates for deploying systems. These templates are stored into S3 that delivers persistent storage on demand. S3 is organized into buckets; these are containers of objects that are stored in binary form and can be enriched with attributes. Users can store objects of any size, from simple files to entire disk images, and have them accessible from everywhere.

2 Google AppEngine

Google AppEngine is a scalable runtime environment mostly devoted to executing Web applications. These take advantage of the large computing infrastructure of Google to dynamically scale as the demand varies over time. AppEngine provides both a secure execution environment and a collection of services that simplify the development of scalable and high-performance Web applications. These services include in-memory caching, scalable data store, job queues, messaging, and cron tasks.
Developers can build and test applications on their own machines using the AppEngine software development kit (SDK). Once development is complete, developers can easily migrate their application to AppEngine, set quotas to contain the costs generated, and make the application available to the world. The languages currently supported are Python, Java, and Go.

3 Microsoft Azure

Microsoft Azure is a cloud operating system and a platform for developing applications in the cloud. Applications in Azure are organized around the concept of roles, which identify a distribution unit for applications and embody the application's logic. Currently, there are three types ofrole: Web role, worker role, and virtual machine role. The Web role is designed to host a Web application, the worker role is a more generic container of applications and can be used to perform workload processing, and the virtual machine role provides a virtual environment in which the computing stack can be fully customized, including the operating systems.

4 Hadoop

Apache Hadoop is an open-source framework that is suited for processing large data sets on commodity hardware. Hadoop is an implementation of MapReduce, an application programming model developed by Google, which provides two fundamental operations for data processing: map and reduce. The former transforms and synthesizes the input data provided by the user; the latter aggragates the output obtained by the map operations. Hadoop provides the runtime environment, and developers need only provide the input data and specify the map and reduce functions that need to be executed.

5 Force.com and Salesforce.com

Force.com is a cloud computing platform for developing social enterprise applications. The platform is the basis for SalesForce.com, a Software-as-a-Service solution for customer relationship management. Force.com allows developers to create applications by composing ready-to-use blocks; a complete set of components supporting all the activities of an enterprise are available. The platform provides complete support for developing applications, from the design of the data layout to the definition of business rules and workflows and the definition of the user interface.

6 Manjrasoft Aneka

Manjrasoft Aneka is a cloud application platform for rapid creation of scalable applications and their deployment on various types of clouds in a seamless and elastic manner. It supports a collection of programming abstractions for developing applications and a distributed runtime environment that can be deployed on heterogeneous hardware (clusters, networked desktop computers, and cloud resources).

Developers can choose different abstractions to design their application: tasks, distributed threads, and map-reduce. These applications are then executed on the distributed service-oriented runtime environment, which can dynamically integrate additional resource on demand.

CHAPTER - 3

3. Virtualization

Virtualization technology is one of the fundamental components of cloud computing, especially in regard to infrastructure-based services. Virtualization allows the creation of a secure, customizable, and isolated execution environment for running applications, even if they are untrusted, without affecting other users' applications. The basis of this technology is the ability of a computer pro-gram—or a combination of software and hardware—to emulate an executing environment separate from the one that hosts such programs.

3.1 Introduction

Virtualization is a large umbrella of technologies and concepts that are meant to provide an abstract environment—whether virtual hardware or an operating system—to run applications. The term virtualization is often synonymous with hardware virtualization, which plays a fundamental role in efficiently delivering Infrastructure-as-a-Service (IaaS) solutions for cloud computing.

Virtualization technologies have gained renewed interested recently due to the confluence of several phenomena:

- Increased performance and computing capacity.

The high-end side of the PC market, where supercomputers can provide immense compute power that can accommodate the execution of hundreds or thousands of virtual machines.

- Underutilized hardware and software resources.
  Hardware and software underutilization is occurring due to (1) increased performance and computing capacity, and (2) the effect of limited or sporadic use of resources. Computers today are so powerful that in most cases only a fraction of their capacity is used by an application or the system. Using these resources for other purposes after hours could improve the efficiency of the IT infrastructure.
- Lack of space.
  Companies such as Google and Microsoft expand their infrastructures by building data centers as large as football fields that are able to host thousands of nodes. Although this is viable for IT giants, in most cases enterprises cannot afford to build another data center to accommodate additional resource capacity. This condition, along with hardware underutilization, has led to the diffusion of a technique called server consolidation
- Greening initiatives.
  Maintaining a data center operation not only involves keeping servers on, but a great deal of energy is also consumed in keeping them cool. Infrastructures for cooling have a significant impact on the carbon footprint of a data center. Hence, reducing the number of servers through server consolidation will definitely reduce the impact of cooling and power consumption of a data center. Virtualization technologies can provide an efficient way of consolidating servers.
- Rise of administrative costs.
  The increased demand for additional capacity, which translates into more servers in a data center, is also responsible for a significant increment in administrative costs. Computers—in particular, servers—do not operate all on their own, but they require care and feeding from system administrators.
  These are labor-intensive operations, and the higher the number of servers that have to be managed, the higher the administrative costs. Virtualization can help reduce the number of required servers for a given workload, thus reducing the cost of the administrative personnel.

3.2 Characteristics of virtualized environments

Virtualization is a broad concept that refers to the creation of a virtual version of something, whether hardware, a software environment, storage, or a network. In a virtualized environment there are three major components: guest, host, and virtualization layer. The guest represents the system component that interacts with the virtualization layer rather than with the host, as would normally happen. The host represents the original environment where the guest is supposed to be managed. The virtualization layer is responsible for recreating the same or a different environment where the guest will operate (see Figure 3.1).

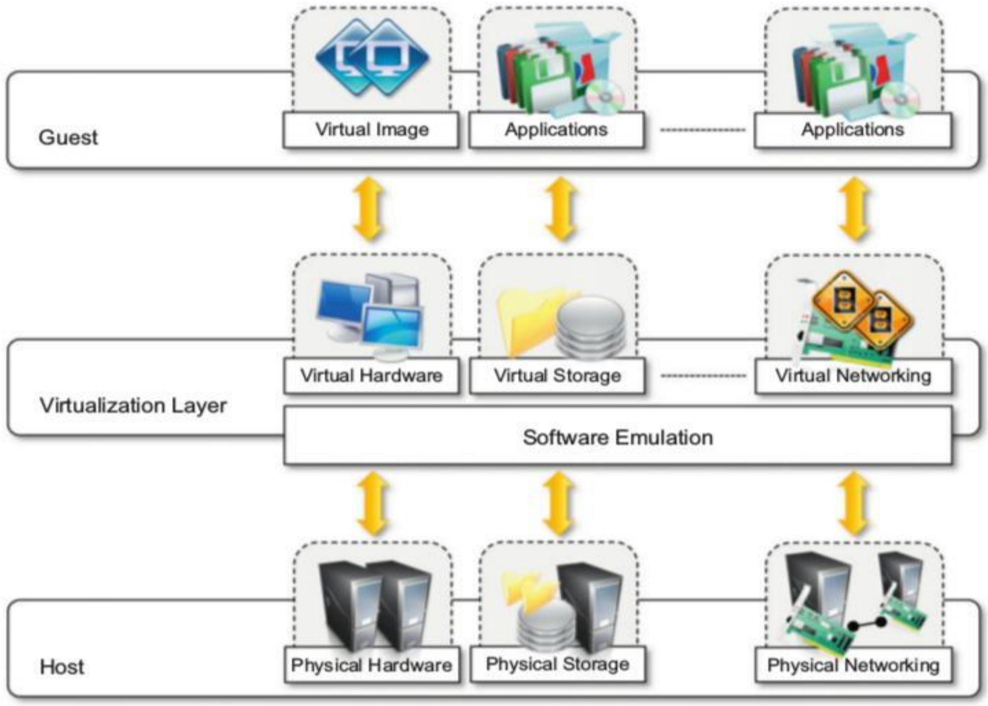

FIGURE 3.1

The virtualization reference model.

The characteristics of virtualized solutions are:
1 Increased security
2 Managed execution
3 Portability

1. Increased security

The virtual machine represents an emulated environment in which the guest is executed. All the operations of the guest are generally performed against the virtual machine, which then translates and applies them to the host. This level of indirection allows the virtual machine manager to control and filter the activity of the guest, thus preventing some harmful operations from being performed. For example, applets downloaded from the Internet run in a sandboxed 3 version of the Java Virtual Machine (JVM), which provides them with limited access to the hosting operating system resources. Both the JVM and the .NET runtime provide extensive security policies for customizing the execution environment of applications.

2 Managed execution.

Virtualization of the execution environment not only allows increased security, but a wider range of features also can be implemented. In particular, sharing, aggregation, emulation, and isolation are the most relevant features (see Figure 3.2).

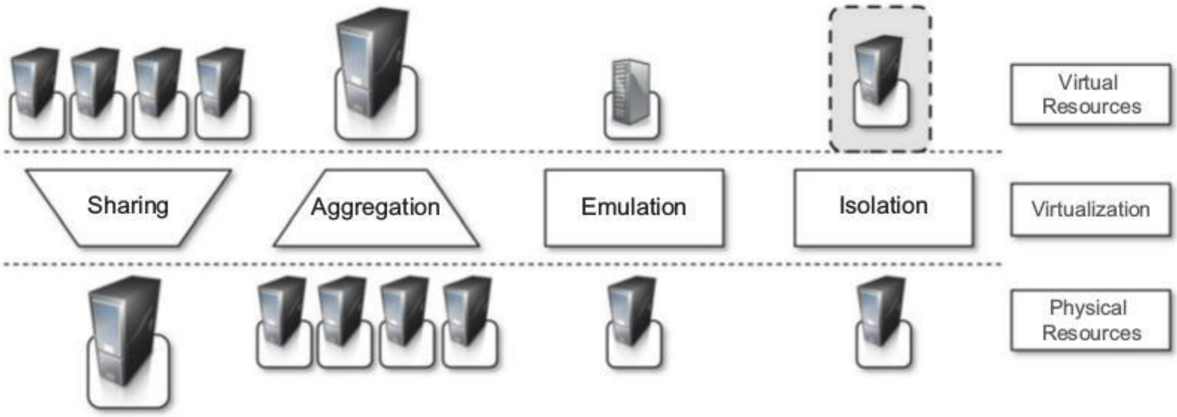

FIGURE 3.2

Functions enabled by managed execution.

Sharing. Virtualization allows the creation of a separate computing environments within the same host. In this way it is possible to fully exploit the capabilities of a powerful guest, which would otherwise be underutilized.

Aggregation. Not only is it possible to share physical resource among several guests, but virtualization also allows aggregation, which is the opposite process. A group of separate hosts can be tied together and represented to guests as a single virtual host.

Emulation. Guest programs are executed within an environment that is controlled by the virtualization layer, which ultimately is a program. This allows for controlling and tuning the environment that is exposed to guests. For instance, a completely different environment with respect to the host can be emulated, thus allowing the execution of guest programs requiring specific characteristics that are not present in the physical host.

Isolation. Virtualization allows providing guests—whether they are operating systems, applications, or other entities—with a completely separate environment, in which they are executed. The guest program performs its activity by interacting with an abstraction layer, which provides access to the underlying resources.

3 Portability

The concept of portability applies in different ways according to the specific type of virtualization considered. In the case of a hardware virtualization solution, the guest is packaged into a virtual image that, in most cases, can be safely moved and executed on top of different virtual machines.

In the case of programming-level virtualization, as implemented by the JVM or the .NET runtime, the binary code representing application components (jars or assemblies) can be run without any recompilation on any implementation of the corresponding virtual machine. This makes the application development cycle more flexible and application deployment very straightforward: One version of the application, in most cases, is able to run on different platforms with no changes.

3.3 Taxonomy of virtualization techniques

Virtualization covers a wide range of emulation techniques that are applied to different areas of computing. A classification of these techniques helps us better understand their characteristics and use (see Figure 3.3).

The first classification discriminates against the service or entity that is being emulated.

Virtualization is mainly used to emulate execution environments, storage, and networks. Among these categories, execution virtualization constitutes the oldest, most popular, and most developed area. Therefore, it deserves major investigation and a further categorization.

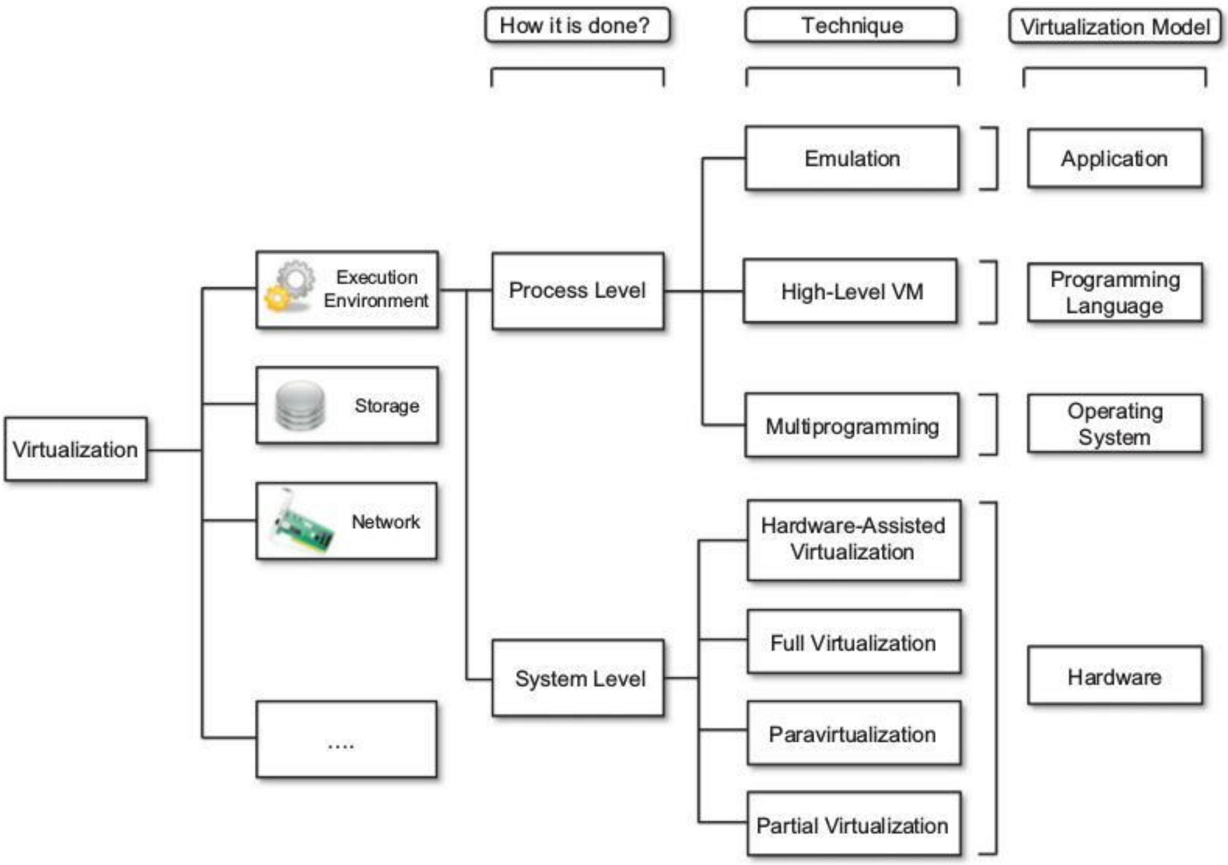

FIGURE 3.3

A taxonomy of virtualization techniques.

3.3.1 Execution virtualization

Execution virtualization includes all techniques that aim to emulate an execution environment that is separate from the one hosting the virtualization layer. All these techniques concentrate their interest on providing support for the execution of programs, whether these are the operating system, a binary specification of a program compiled against an abstract machine model, or an application.

1 Machine reference model
2 Hardware-level virtualization
- **a. Hypervisors**
- **b. Hardware virtualization techniques**
- **c. Operating system-level virtualization**

3. Programming language-level virtualization
4. Application-level virtualization

1 Machine reference model

Modern computing systems can be expressed in terms of the reference model described in Figure 3.4. At the bottom layer, the model for the hardware is expressed in terms of the Instruction Set

Architecture (ISA), which defines the instruction set for the processor, registers, memory, and interrupt management. ISA is the interface between hardware and software, and it is important to the operating system (OS) developer (System ISA) and developers of applications that directly manage the underlying hardware (User ISA). The application binary interface (ABI) separates the operating system layer from the applications and libraries, which are managed by the OS. ABI covers details such as low-level data types, alignment, and call conventions and defines a format for executable programs.
The highest level of abstraction is represented by the application programming interface (API), which interfaces applications to libraries and/or the underlying operating system.

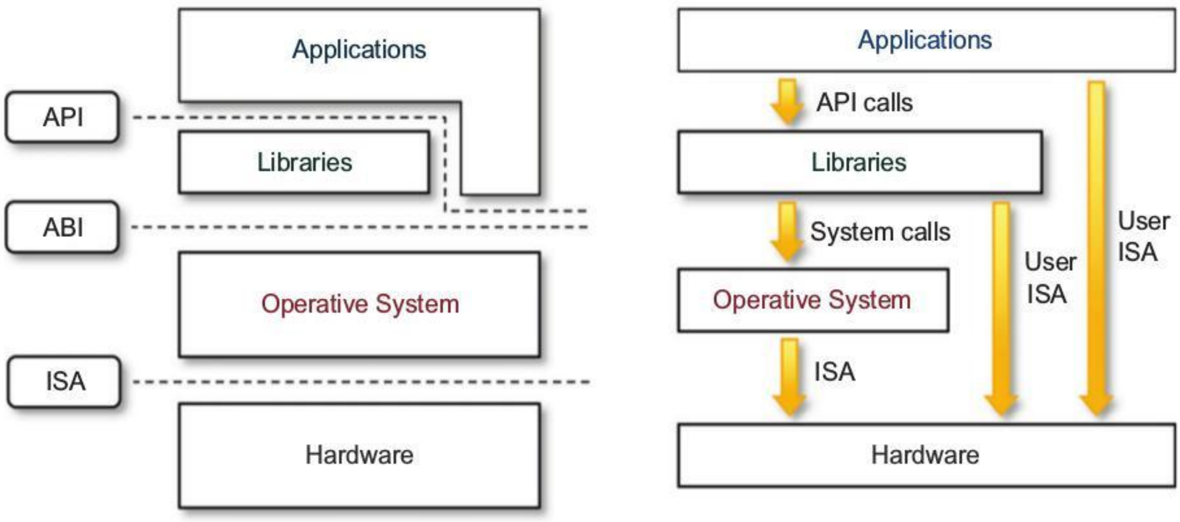

FIGURE 3.4

A machine reference model.

For this purpose, the instruction set exposed by the hardware has been divided into different security classes that define who can operate with them. The first distinction can be made between privileged and nonprivileged instructions. Nonprivileged instructions are those instructions that can be used without interfering with other tasks because they do not access shared resources. This category contains, for example, all the floating, fixed-point, and arithmetic instructions. Privileged instructions are those that are executed under specific restrictions and are mostly used for sensitive operations, which expose (behavior-sensitive) or modify (control-sensitive) the privileged state.
For instance, a possible implementation features a hierarchy of privileges (see Figure 3.5) in the form of ring-based security: Ring 0, Ring 1, Ring 2, and Ring 3; Ring 0 is in the most privileged level and Ring 3 in the least privileged level. Ring 0 is used by the kernel of the OS, rings 1 and 2 are used by the OS -level services, and Ring 3 is used by the user. Recent systems support only two levels, with Ring 0 for supervisor mode and Ring 3 for user mode.

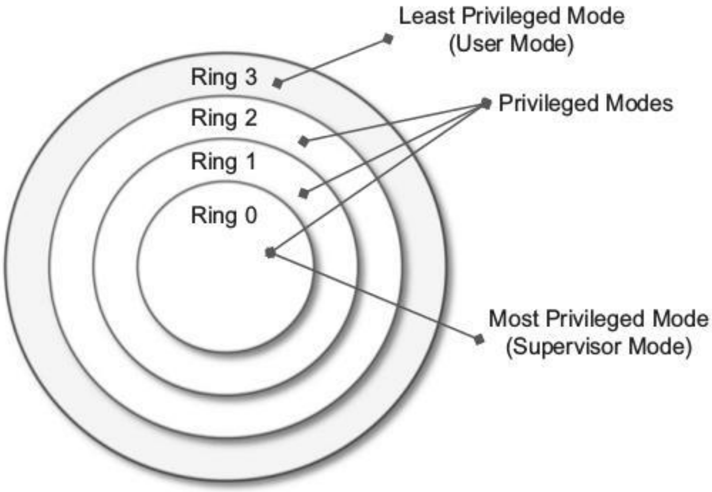

FIGURE 3.5

Security rings and privilege modes.

The distinction between user and supervisor mode allows us to understand the role of the hypervisor and why it is called that. Conceptually, the hypervisor runs above the supervisor mode, and from here the prefix hyper- is used. In reality, hypervisors are run in supervisor mode.

2 Hardware-level virtualization

Hardware-level virtualization is a virtualization technique that provides an abstract execution envi-ronment in terms of computer hardware on top of which a guest operating system can be run. In this model, the guest is represented by the operating system, the host by the physical computer hardware, the virtual machine by its emulation, and the virtual machine manager by the hypervisor (see Figure 3.6). The hypervisor is generally a program or a combination of software and hardware that allows the abstraction of the underlying physical hardware. Hardware-level virtualization is also called system virtualization, since it provides ISA to virtual machines, which is the representation of the hardware interface of a system. This is to differentiate it from process virtual machines, which expose ABI to virtual machines.

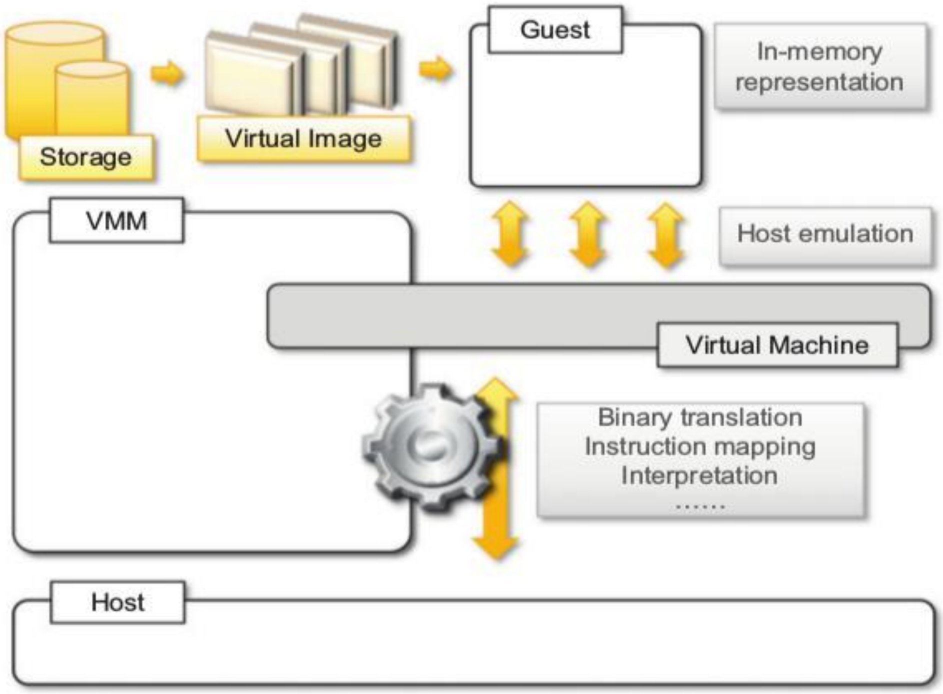

FIGURE 3.6

A hardware virtualization reference model.

a. Hypervisors

A fundamental element of hardware virtualization is the hypervisor, or virtual machine manager (VMM). It recreates a hardware environment in which guest operating systems are installed. There are two major types of hypervisor: Type I and Type II (see Figure 3.7).

Type I hypervisors run directly on top of the hardware. Therefore, they take the place of the operating systems and interact directly with the ISA interface exposed by the underlying hardware, and they emulate this interface in order to allow the management of guest operating systems. This type of hypervisor is also called a native virtual machine since it runs natively on hardware.

Type II hypervisors require the support of an operating system to provide virtualization services. This means that they are programs managed by the operating system, which interact with it through the ABI and emulate the ISA of virtual hardware for guest operating systems. This type of hypervisor is also called a hosted virtual machine since it is hosted within an operating system.

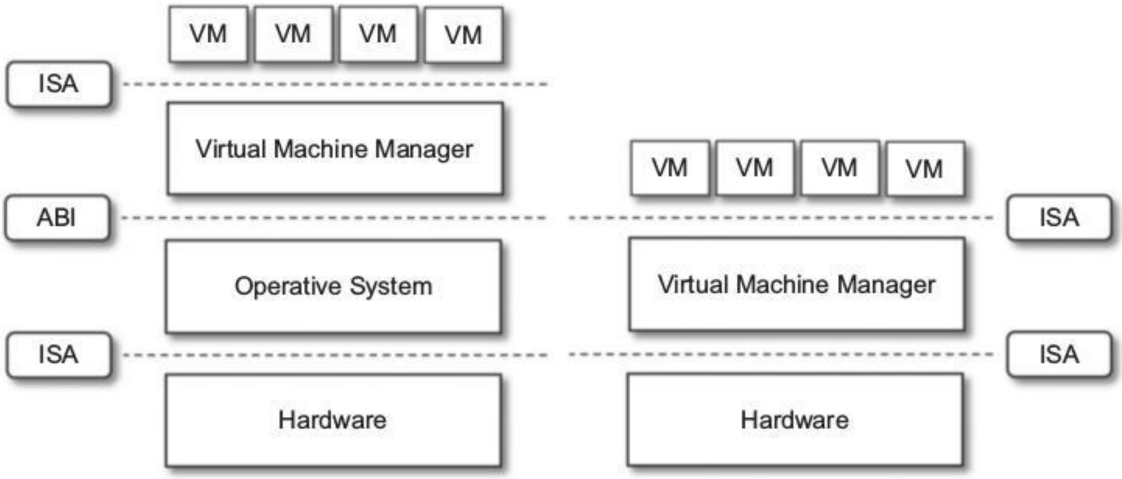

FIGURE 3.7

Hosted (left) and native (right) virtual machines. This figure provides a graphical representation of the two types of hypervisors.

Conceptually, a virtual machine manager is internally organized as described in Figure 3.8. Three main modules, dispatcher, allocator, and interpreter, coordinate their activity in order to emulate the underlying hardware. The dispatcher constitutes the entry point of the monitor and reroutes the instructions issued by the virtual machine instance to one of the two other modules. The allocator is responsible for deciding the system resources to be provided to the VM: whenever a virtual machine tries to execute an instruction that results in changing the machine resources associated with that VM, the allocator is invoked by the dispatcher. The interpreter module consists of interpreter routines. These are executed whenever a virtual machine executes a privileged instruction: a trap is triggered and the corresponding routine is executed.

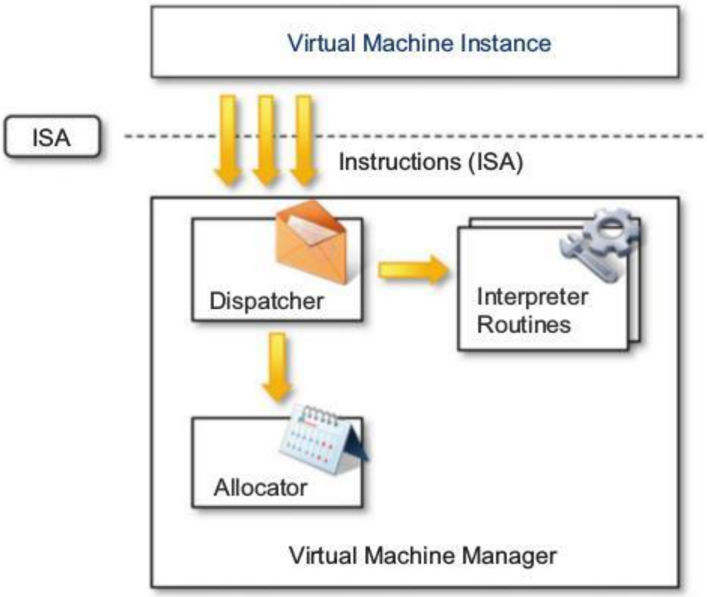

FIGURE 3.8

A hypervisor reference architecture.

The design and architecture of a virtual machine manager, together with the underlying hardware design of the host machine, determine the full realization of hardware virtualization, where a guest operating system can be transparently executed on top of a VMM as though it were run on the underlying hardware. The criteria that need to be met by a virtual machine manager to efficiently support virtualization were established by Goldberg and Popek in 1974 [23]. Three properties have to be satisfied:

1. Equivalence. A guest running under the control of a virtual machine manager should exhibit the same behavior as when it is executed directly on the physical host.
2. Resource control. The virtual machine manager should be in complete control of virtualized resources.
3. Efficiency. A statistically dominant fraction of the machine instructions should be executed without intervention from the virtual machine manager.

Popek and Goldberg provided a classification of the instruction set and proposed three theorems that define the properties that hardware instructions need to satisfy in order to efficiently support virtualization.

THEOREM 3.1

For any conventional third-generation computer, a VMM may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions.

This theorem establishes that all the instructions that change the configuration of the system resources should generate a trap in user mode and be executed under the control of the virtual machine manager.

THEOREM 3.2

A conventional third-generation computer is recursively virtualizable if:

- It is virtualizable and
- A VMM without any timing dependencies can be constructed for it.

Recursive virtualization is the ability to run a virtual machine manager on top of another virtual machine manager. This allows nesting hypervisors as long as the capacity of the underlying resources can accommodate that. Virtualizable hardware is a prerequisite to recursive virtualization.

THEOREM 3.3

A hybrid VMM may be constructed for any conventional third-generation machine in which the set of user-sensitive instructions is a subset of the set of privileged instructions.

There is another term, hybrid virtual machine (HVM), which is less efficient than the virtual machine system. In the case of an HVM, more instructions are interpreted rather than being executed directly. All instructions in virtual supervisor mode are interpreted. Whenever there is an attempt to execute a behavior-sensitive or control-sensitive instruction, HVM controls the execution directly or gains the control via a trap. Here all sensitive instructions are caught by HVM that are simulated.

b. Hardware virtualization techniques

Hardware-assisted virtualization. This term refers to a scenario in which the hardware provides architectural support for building a virtual machine manager able to run a guest operating system in complete isolation. This technique was originally introduced in the IBM System/370. At present, examples of hardware-assisted virtualization are the extensions to the x86-64 bit architecture introduced with Intel VT (formerly known as Vanderpool) and AMD V (formerly known as Pacifica). Intel and AMD introduced processor extensions, and a wide range of virtualization solutions took advantage of them: Kernel-based Virtual Machine (KVM), VirtualBox, Xen, VMware, Hyper-V, Sun xVM, Parallels, and others.

Full virtualization. Full virtualization refers to the ability to run a program, most likely an operating system, directly on top of a virtual machine and without any modification, as though it were run on the raw hardware. To make this possible, virtual machine managers are required to provide a complete emulation of the entire underlying hardware. The principal advantage of full virtualization is complete isolation, which leads to enhanced security, ease of emulation of different architectures, and coexistence of different systems on the same platform.

Paravirtualization. This is a not-transparent virtualization solution that allows implementing thin virtual machine managers. Paravirtualization techniques expose a software interface to the virtual machine that is slightly modified from the host and, as a consequence, guests need to be modified. The aim of paravirtualization is to provide the capability to demand the execution of performance-critical operations directly on the host, thus preventing performance losses that would otherwise be experienced in managed execution.

Partial virtualization. Partial virtualization provides a partial emulation of the underlying hard-ware, thus not allowing the complete execution of the guest operating system in complete isolation. Partial virtualization allows many applications to run transparently, but not all the features of the operating system can be supported.

c. Operating system-level virtualization

Operating system-level virtualization offers the opportunity to create different and separated execu-tion environments for applications that are managed concurrently. Differently from hardware virtua-lization, there is no virtual machine manager or hypervisor, and the virtualization is done within a single operating system, where the OS kernel allows for multiple isolated user space instances. The kernel is also responsible for sharing the system resources among instances and for limiting the impact of instances on each other.

This virtualization technique can be considered an evolution of the chroot mechanism in Unix systems. The chroot operation changes the file system root directory for a process and its children to a specific directory. As a result, the process and its children cannot have access to other portions of the file system than those accessible under the new root directory.

Examples of operating system-level virtualizations are FreeBSD Jails, IBM Logical Partition (LPAR), SolarisZones and Containers, Parallels Virtuozzo Containers, OpenVZ, iCore Virtual Accounts, Free Virtual Private Server (FreeVPS).

3 Programming language-level virtualization

Programming language-level virtualization is mostly used to achieve ease of deployment of applications, managed execution, and portability across different platforms and operating sys - tems. It consists of a virtual machine executing the byte code of a program, which is the result of the compilation process. Compilers implemented and used this technology to produce a binary format representing the machine code for an abstract architecture.

Programming language-level virtualization has a long trail in computer science history and originally was used in 1966 for the implementation of Basic Combined Programming Language (BCPL), a language for writing compilers and one of the ancestors of the C programming language.

The ability to support multiple programming languages has been one of the key elements of the Common Language Infrastructure (CLI), which is the specification behind .NET Framework. Currently, the Java platform and .NET Framework represent the most popular technologies for enterprise application development. Both Java and the CLI are stack-based virtual machines.
The main advantage of programming-level virtual machines, also called process virtual machines, is the ability to provide a uniform execution environment across different platforms.
The process virtual machines allow for more control over the execution of programs since they do not provide direct access to the memory. Security is another advantage of managed programming languages; by filtering the I/O operations, the process virtual machine can easily support sandboxing of applications.

4 Application-level virtualization
Application-level virtualization is a technique allowing applications to be run in runtime environments that do not natively support all the features required by such applications. In this scenario, applications are not installed in the expected runtime environment but are run as though they were.
Emulation can also be used to execute program binaries compiled for different hardware architectures. In this case, one of the following strategies can be implemented:
a. Interpretation. In this technique every source instruction is interpreted by an emulator for executing native ISA instructions, leading to poor performance. Interpretation has a minimal startup cost but a huge overhead, since each instruction is emulated.
b. Binary translation. In this technique every source instruction is converted to native instructions with equivalent functions. After a block of instructions is translated, it is cached and reused. Binary translation has a large initial overhead cost, but over time it is subject to better performance, since previously translated instruction blocks are directly executed.

3.3.2 Other types of virtualization
Other than execution virtualization, other types of virtualization provide an abstract environment to interact with. These mainly cover storage, networking, and client/server interaction.

1 Storage virtualization
Storage virtualization is a system administration practice that allows decoupling the physical organization of the hardware from its logical representation. Using this technique, users do not have to be worried about the specific location of their data, which can be identified using a logical path.
Storage virtualization allows us to harness a wide range of storage facilities and represent them under a single logical file system. There are different techniques for storage virtualization, one of the most popular being network-based virtualization by means of storage area networks (SANs).

2 Network virtualization
Network virtualization combines hardware appliances and specific software for the creation and management of a virtual network. Network virtualization can aggregate different physical networks into a single logical network (external network virtualization) or provide network-like functionality to an operating system partition (internal network virtualization). The result of external network virtualization is generally a virtual LAN (VLAN).

3 Desktop virtualization
Desktop virtualization abstracts the desktop environment available on a personal computer in order to provide access to it using a client/server approach. Desktop virtualization provides the same out-

come of hardware virtualization but serves a different purpose. Similarly to hardware virtualization, desktop virtualization makes accessible a different system as though it were natively installed on the host, but this system is remotely stored on a different host and accessed through a network connection. Moreover, desktop virtualization addresses the problem of making the same desktop environment accessible from everywhere.

4 Application server virtualization

Application server virtualization abstracts a collection of application servers that provide the same services as a single virtual application server by using load-balancing strategies and providing a high-availability infrastructure for the services hosted in the application server. This is a particular form of virtualization and serves the same purpose of storage virtualization: providing a better qual-ity of service rather than emulating a different environment.

3.4 Virtualization and cloud computing

Virtualization plays an important role in cloud computing since it allows for the appropriate degree of customization, security, isolation, and manageability that are fundamental for delivering IT ser-vices on demand. Particularly important is the role of virtual computing environment and execution virtualization techniques. Among these, hardware and programming language virtualization are the techniques adopted in cloud computing systems.
Besides being an enabler for computation on demand, virtualization also gives the opportunity to design more efficient computing systems by means of consolidation, which is performed transparently to cloud computing service users.

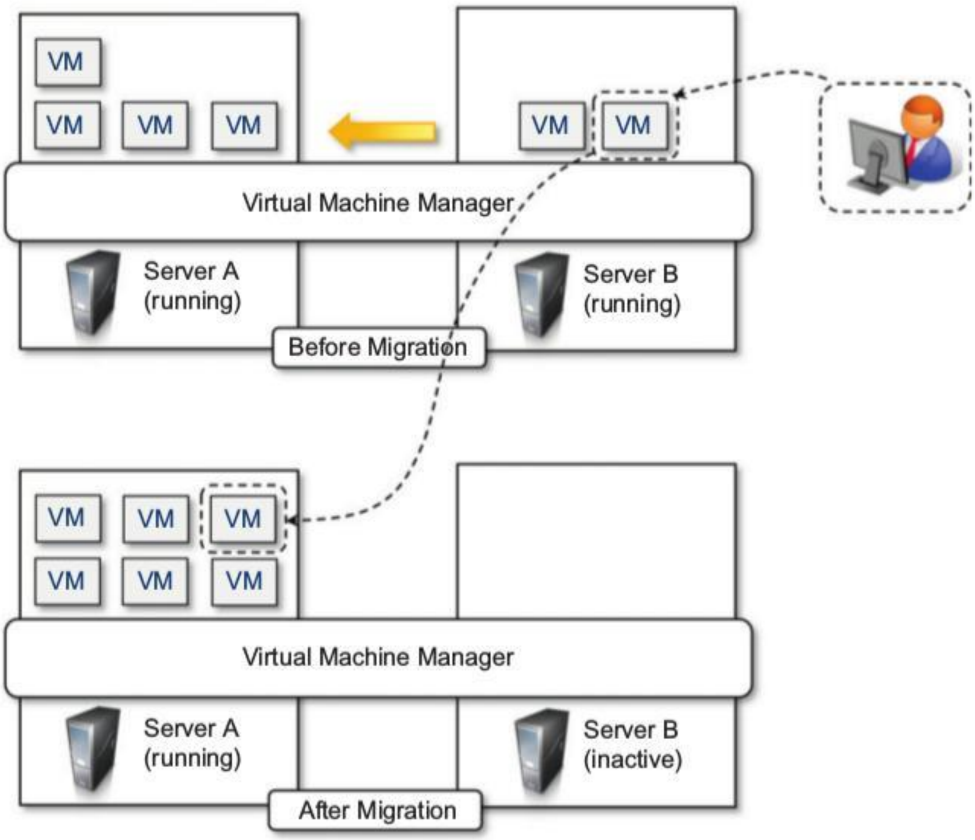

FIGURE 3.10

Live migration and server consolidation.

Since virtualization allows us to create isolated and controllable environments, it is possible to serve these environments with the same resource without them interfering with each other. This opportunity is particularly attractive when resources are underutilized, because it allows reducing the number of active resources by aggregating virtual machines over a smaller number of resources that become fully utilized. This practice is also known as server consolidation, while the movement of virtual machine instances is called virtual machine migration (see Figure 3.10). Because virtual machine instances are controllable environments, consolidation can be applied with a minimum impact, either by temporarily stopping its execution and moving its data to the new resources or by performing a finer control and moving the instance while it is running. This second techniques is known as live migration and in general is more complex to implement but more efficient since there is no disruption of the activity of the virtual machine instance.

3.5 Pros and cons of virtualization

Virtualization has now become extremely popular and widely used, especially in cloud computing. Today, the capillary diffusion of the Internet connection and the advancements in computing technology have made virtualization an interesting opportunity to deliver on-demand IT infrastructure and services.

3.5.1 Advantages of virtualization

1. Managed execution and isolation are perhaps the most important advantages of virtualization. In the case of techniques supporting the creation of virtualized execution environments, these two characteristics allow building secure and controllable computing environments.
2. Portability is another advantage of virtualization, especially for execution virtualization techni-ques. Virtual machine instances are normally represented by one or more files that can be easily transported with respect to physical systems.
3. Portability and self-containment also contribute to reducing the costs of maintenance, since the number of hosts is expected to be lower than the number of virtual machine instances. Since the guest program is executed in a virtual environment, there is very limited opportunity for the guest program to damage the underlying hardware.
4. Finally, by means of virtualization it is possible to achieve a more efficient use of resources. Multiple systems can securely coexist and share the resources of the underlying host, without inter-fering with each other.

3.5.2 The other side of the coin: disadvantages

1 Performance degradation

Performance is definitely one of the major concerns in using virtualization technology. Since virtualization interposes an abstraction layer between the guest and the host, the guest can experience increased latencies (delays).

For instance, in the case of hardware virtualization, where the intermediate emulates a bare machine on top of which an entire system can be installed, the causes of performance degradation can be traced back to the overhead introduced by the following activities:

- Maintaining the status of virtual processors
- Support of privileged instructions (trap and simulate privileged instructions)
- Support of paging within VM
- Console functions

2 Inefficiency and degraded user experience

Virtualization can sometime lead to an inefficient use of the host. In particular, some of the specific features of the host cannot be exposed by the abstraction layer and then become inaccessible. In the case of hardware virtualization, this could happen for device drivers: The virtual machine can sometime simply provide a default graphic card that maps only a subset of the features available in the host. In the case of programming-level virtual machines, some of the features of the underlying operating systems may become inaccessible unless specific libraries are used.

3 Security holes and new threats

Virtualization opens the door to a new and unexpected form of phishing. The capability of emulating a host in a completely transparent manner led the way to malicious programs that are designed to extract sensitive information from the guest.

The same considerations can be made for programming-level virtual machines: Modified ver- sions of the runtime environment can access sensitive information or monitor the memory locations utilized by guest applications while these are executed.