implementation would not require the cost of a host operating system, though a true cost comparison would be a more complicated discussion. Type 2 hypervisors allow a user to take advantage of virtualization without needing to dedicate a server to only that function. Developers who need to run multiple environments as part of their process, in addition to taking advantage of the personal productive workspace that a PC operating system provides, can do both with a Type 2 hypervisor installed as an application on their Linux or Windows desktop. The VMs that are created and used can be migrated or copied from one hypervisor environment to another, reducing deployment time and increasing the accuracy of what is deployed, reducing the time to market of a project.

Container Virtualization

A relatively recent approach to virtualization is known as **container virtualization**. In this approach, software, known as a virtualization **container**, runs on top of the host OS kernel and provides an execution environment for applications (Figure 7.4). Unlike hypervisor-based VMs, containers do not aim to emulate physical servers. Instead, all containerized applications on a host share a common OS kernel. This eliminates the resources needed to run a separate OS for each application and can greatly reduce overhead.

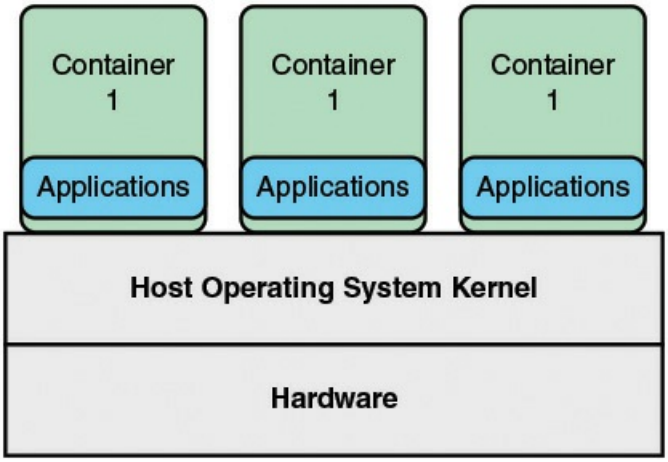


FIGURE 7.4 Container Virtualization

Because the containers execute on the same kernel, thus sharing most of the base OS, containers are much smaller and lighter weight compared to a hypervisor/guest OS VM arrangement. Accordingly, an OS can have many containers running on top of it, compared to the limited number of hypervisors and guest operating systems that can be supported.

7.3 NFV Concepts

Chapter 2, "Requirements and Technology," defined network functions virtualization (NFV) as the virtualization of network functions by implementing these functions in software and running them on VMs. NFV is a significant departure from traditional approaches to the design, deployment, and management of networking services. NFV decouples network functions, such

as Network Address Translation (NAT), firewalling, intrusion detection, Domain Name Service (DNS), and caching, from proprietary hardware appliances so that they can run in software on VMs. NFV builds on standard VM technologies, extending their use into the networking domain.

Virtual machine technology, as discussed in Section 7.2, enables migration of dedicated application and database servers to **commercial off-the-shelf (COTS)** x86 servers. The same technology can be applied to network-based devices, including the following:

- **Network function devices:** Such as switches, routers, network access points, customer premises equipment (CPE), and deep packet inspectors (for **deep packet inspection**).
- **Network-related compute devices:** Such as firewalls, intrusion detection systems, and network management systems.
- **Network-attached storage:** File and database servers attached to the network.

In traditional networks, all devices are deployed on proprietary/closed platforms. All network elements are enclosed boxes, and hardware cannot be shared. Each device requires additional hardware for increased capacity, but this hardware is idle when the system is running below capacity. With NFV, however, network elements are independent applications that are flexibly deployed on a unified platform comprising standard servers, storage devices, and switches. In this way, software and hardware are decoupled, and capacity for each application is increased or decreased by adding or reducing virtual resources (see Figure 7.5).

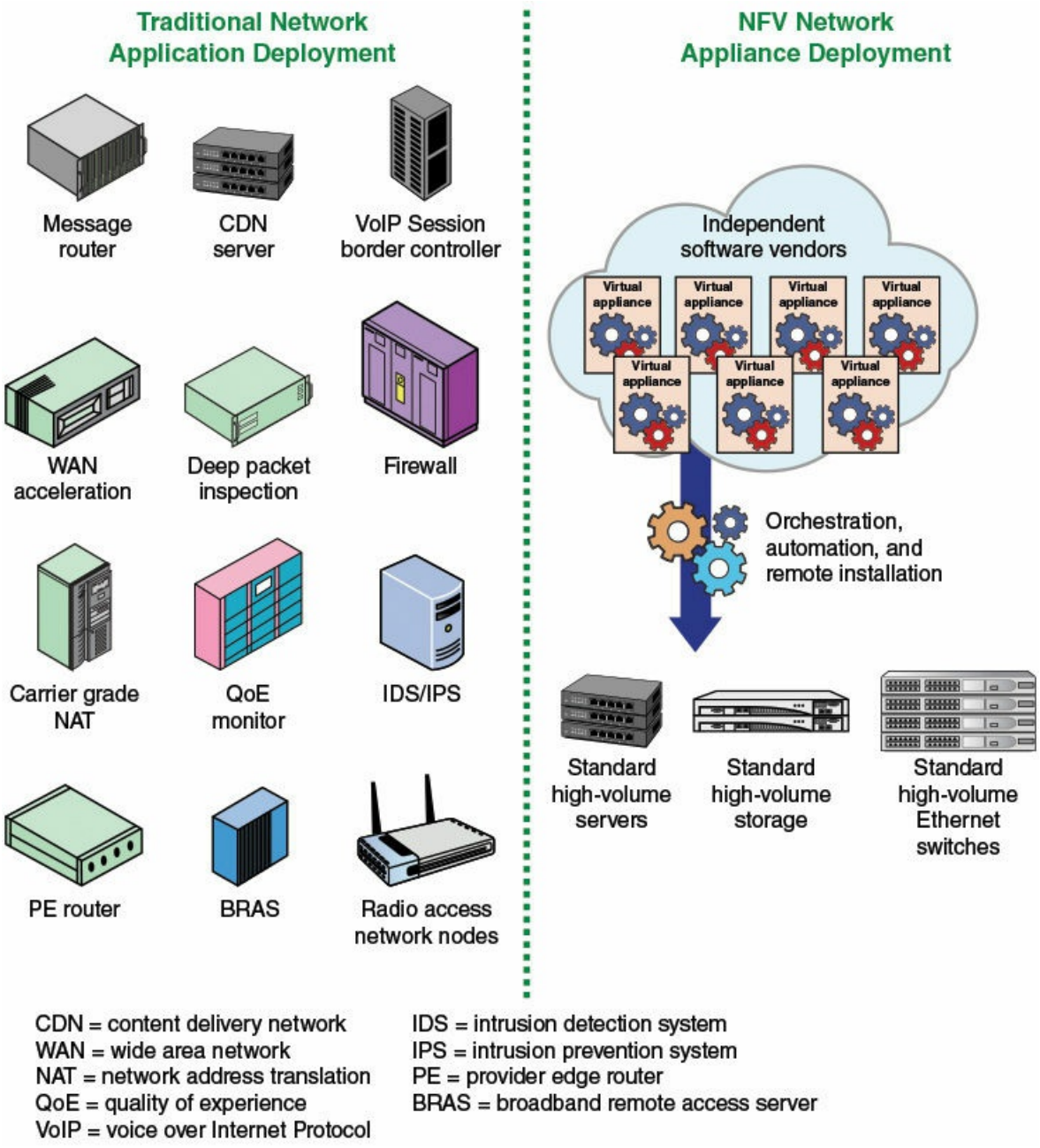


FIGURE 7.5 Vision for Network Functions Visualization

By broad consensus, the Network Functions Virtualization Industry Standards Group (ISG NFV), created as part of the European Telecommunications Standards Institute (ETSI), has the lead and indeed almost the sole role in creating NFV standards. ISG NFV was established in 2012 by seven major telecommunications network operators. Its membership has since grown to include network equipment vendors, network technology companies, other IT companies, and service providers such as cloud service providers.

NFV ISG

ISG NFV published the first batch of specifications in October 2013, and subsequently updated most of those in late 2014 and early 2015. Table 7.1 shows the complete list of specifications as of early 2015. Table 7.2 provides definitions for a number of terms that are used in the ISG NFV documents and the NFV literature in general.

Standard Number	Standard Title
GS NFV 002	Architectural Framework
GS NFV-INF 001	Infrastructure Overview
GS NFV-INF 003	Infrastructure; Computer Domain
GS NFV-INF 004	Infrastructure; Hypervisor Domain
GS NFV-INF 005	Infrastructure; Network Domain
GS NFV-INF 007	Infrastructure; Methodology to Describe Interfaces and Abstractions
GS NFV-MAN 001	Management and Orchestration
GS NFV-SEC 001	NFV Security; Problem Statement
GS NFV-SEC 003	NFV Security; Security and Trust Guidance
GS NFV-PER 001	NFV Performance & Portability Best Practices
GS NFV-PER 002	Proofs of Concept; Framework
GS NFV-REL 001	Resiliency Requirements
GS NFV-INF 010	Service Quality Metrics
GS NFV 003	Terminology for Main Concepts in NFV
GS NFV 001	Use Cases
GS NFV-SWA 001	Virtual Network Functions Architecture
GS NFV 004	Virtualization Requirements

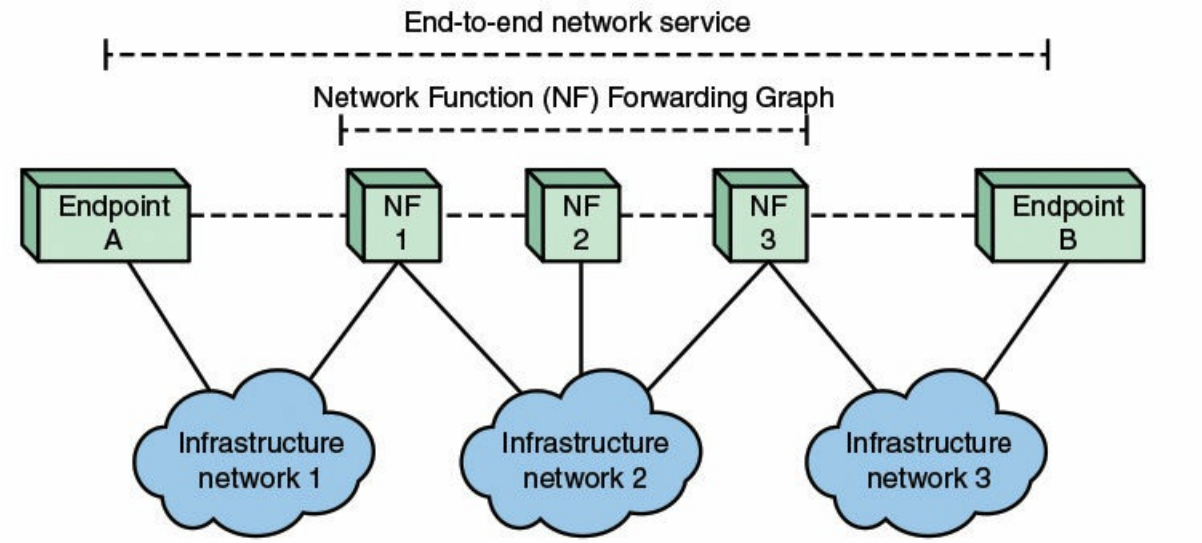
TABLE 7.1 ISG NFV Specifications

Term	Definition
Compute domain	Domain within the NFVI that includes servers and storage.
Infrastructure network domain (IND)	Domain within the NFVI that includes all networking that interconnects compute/storage infrastructure.
Network function (NF)	A functional block within a network infrastructure that has well-defined external interfaces and well-defined functional behavior. Typically, this is a physical network node or other physical appliance.
Network functions virtualization (NFV)	The principle of separating network functions from the hardware they run on by using virtual hardware abstraction.
Network functions virtualization infrastructure (NFVI)	The totality of all hardware and software components that build up the environment in which virtual network functions (VNFs) are deployed. The NFVI can span across several locations (that is, multiple points of presence [N-PoPs]). The network providing connectivity between these locations is considered to be part of the NFVI.
NFVI-Node	Physical devices deployed and managed as a single entity, providing the NFVI functions required to support the execution environment for VNFs.
NFVI-PoP	An N-PoP where a network function is or could be deployed as a VNF.
Network forwarding path	Ordered list of connection points forming a chain of NFs, along with policies associated with the list.
Network point of presence (N-PoP)	A location where a network function is implemented as either a physical network function (PNF) or a VNF.
Network service	A composition of network functions that is defined by its functional and behavioral specification.
Physical network function (PNF)	An implementation of a NF via a tightly coupled software and hardware system. This is typically a proprietary system.
Virtual machine (VM)	A virtualized computation environment that behaves very much like a physical computer/server.
Virtual network	A topological component used to affect routing of specific characteristic information. The virtual network is bounded by its set of permissible network interfaces. In the NFVI architecture, a virtual network routes information among the network interfaces of VM instances and physical network interfaces, providing the necessary connectivity.
Virtualized network function (VNF)	An implementation of an NF that can be deployed on an NFVI.
VNF forwarding graph (VNF FG)	Graph of logical links connecting VNF nodes for the purpose of describing traffic flow between these network functions.
VNF set	Collection of VNFs with unspecified connectivity between them.

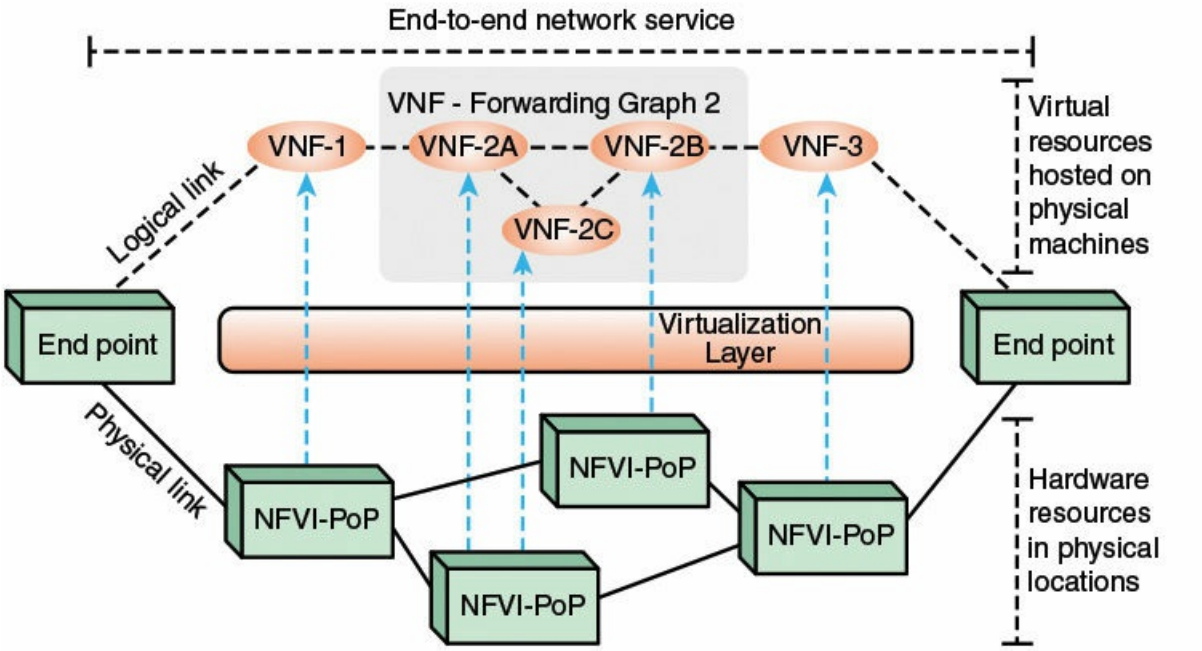
TABLE 7.2 NFV Terminology

Simple Example of the Use of NFV

This section considers a simple example from the NFV Architectural Framework document. Part a of Figure 7.6 shows a physical realization of a network service. At a top level, the network service consists of endpoints connected by a forwarding graph of network functional blocks, called network functions (NFs). Examples of NFs are firewalls, load balancers, and wireless network access points. In the Architectural Framework, NFs are viewed as distinct physical nodes. The endpoints are beyond the scope of the NFV specifications and include all customer-owned devices. So, in the figure, endpoint A could be a smartphone and endpoint B a content delivery network (CDN) server.



(a) Graph representation of an end-to-end network service



(b) Example of an end-to-end network service with VNFs and nested forwarding graphs

FIGURE 7.6 A Simple NFV Configuration Example

Part a of Figure 7.6 highlights the network functions that are relevant to the service provider and customer. The interconnections among the NFs and endpoints are depicted by dashed lines, representing logical links. These logical links are supported by physical paths through infrastructure networks (wired or wireless).

Part b of Figure 7.6 illustrates a virtualized network service configuration that could be implemented on the physical configuration of part a of Figure 7.6. VNF-1 provides network access for endpoint A, and VNF-2 provides network access for B. The figure also depicts the

case of a nested VNF forwarding graph (VNF-FG-2) constructed from other VNFs (that is, VNF-2A, VNF-2B and VNF-2C). All of these VNFs run as VMs on physical machines, called points of presence (PoPs). This configuration illustrates several important points. First, VNF-FG-2 consists of three VNFs even though ultimately all the traffic transiting VNF-FG-2 is between VNF-1 and VNF-3. The reason for this is that three separate and distinct network functions are being performed. For example, it may be that some traffic flows need to be subjected to a traffic policing or shaping function, which could be performed by VNF-2C. So, some flows would be routed through VNF-2C, while others would bypass this network function.

A second observation is that two of the VMs in VNF-FG-2 are hosted on the same physical machine. Because these two VMs perform different functions, they need to be distinct at the virtual resource level but can be supported by the same physical machine. But this is not required, and a network management function may at some point decide to migrate one of the VMs to another physical machine, for reasons of performance. This movement is transparent at the virtual resource level.

NFV Principles

As suggested by Figure 7.6, the VNFs are the building blocks used to create end-to-end network services. Three key NFV principles are involved in creating practical network services:

- **Service chaining:** VNFs are modular and each VNF provides limited functionality on its own. For a given traffic flow within a given application, the service provider steers the flow through multiple VNFs to achieve the desired network functionality. This is referred to as service chaining.
- **Management and orchestration (MANO):** This involves deploying and managing the lifecycle of VNF instances. Examples include VNF instance creation, VNF service chaining, monitoring, relocation, shutdown, and billing. MANO also manages the NFV infrastructure elements.
- **Distributed architecture:** A VNF may be made up of one or more VNF components (VNFC), each of which implements a subset of the VNF's functionality. Each VNFC may be deployed in one or multiple instances. These instances may be deployed on separate, distributed hosts to provide scalability and redundancy.

High-Level NFV Framework

Figure 7.7 shows a high-level view of the NFV framework defined by ISG NFV. This framework supports the implementation of network functions as software-only VNFs. We use this to provide an overview of the NFV architecture, which is examined in more detail in Chapter 8, "NFV Functionality."

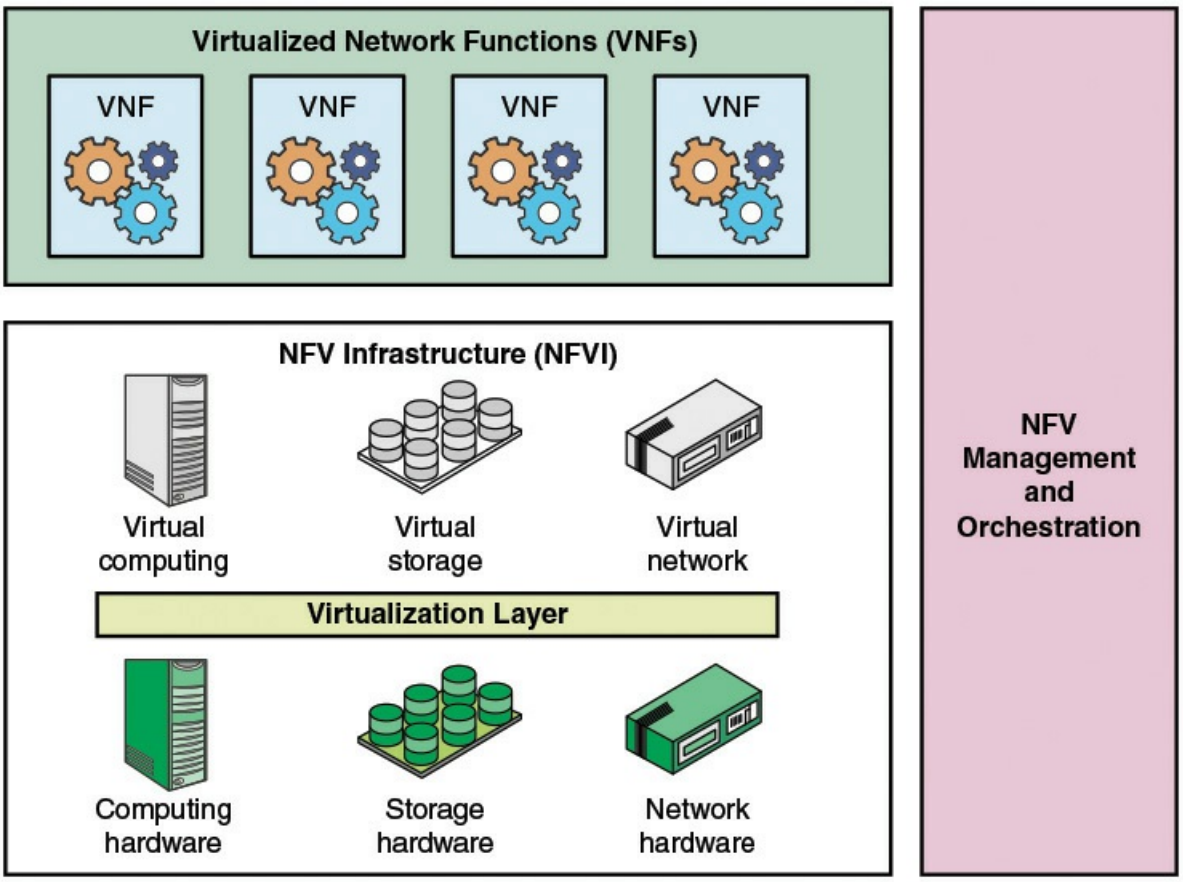


FIGURE 7.7 High-Level NFV Framework

The NFV framework consists of three domains of operation:

- **Virtualized network functions:** The collection of VNFs, implemented in software, that run over the NFVI.
- **NFV infrastructure (NFVI):** The NFVI performs a virtualization function on the three main categories of devices in the network service environment**:** computer devices, storage devices, and network devices.
- **NFV management and orchestration:** Encompasses the orchestration and lifecycle management of physical/software resources that support the infrastructure virtualization, and the lifecycle management of VNFs. NFV management and orchestration focuses on all virtualization-specific management tasks necessary in the NFV framework.

The ISG NFV Architectural Framework document specifies that in the deployment, operation, management and orchestration of VNFs, two types of relations between VNFs are supported:

- **VNF forwarding graph (VNF FG):** Covers the case where network connectivity between VNFs is specified, such as a chain of VNFs on the path to a web server tier (for example, firewall, network address translator, load balancer).
- **VNF set:** Covers the case where the connectivity between VNFs is not specified, such as a web server pool.

7.4 NFV Benefits and Requirements

Having considered on overview of NFV concepts, we can now summarize key benefits of NFV and requirements for successful implementation.

NFV Benefits

If NFV is implemented efficiently and effectively, it can provide a number of benefits compared to traditional networking approaches. The following are the most important potential benefits:

- Reduced **CapEx**, by using commodity servers and switches, consolidating equipment, exploiting economies of scale, and supporting pay-as-you grow models to eliminate wasteful overprovisioning. This is perhaps the main driver for NFV.
- Reduced **OpEx**, in terms of power consumption and space usage, by using commodity servers and switches, consolidating equipment, and exploiting economies of scale, and reduced network management and control expenses. Reduced CapeX and OpEx are perhaps the main drivers for NFV.
- The ability to innovate and roll out services quickly, reducing the time to deploy new networking services to support changing business requirements, seize new market opportunities, and improve return on investment of new services. Also lowers the risks associated with rolling out new services, allowing providers to easily trial and evolve services to determine what best meets the needs of customers.
- Ease of interoperability because of standardized and open interfaces.
- Use of a single platform for different applications, users and tenants. This allows network operators to share resources across services and across different customer bases.
- Provided agility and flexibility, by quickly scaling up or down services to address changing demands.
- Targeted service introduction based on geography or customer sets is possible. Services can be rapidly scaled up/down as required.
- A wide variety of ecosystems and encourages openness. It opens the virtual appliance market to pure software entrants, small players and academia, encouraging more innovation to bring new services and new revenue streams quickly at much lower risk.

NFV Requirements

To deliver these benefits, NFV must be designed and implemented to meet a number of requirements and technical challenges, including the following [ISGN12]:

- **Portability/interoperability:** The capability to load and execute VNFs provided by different vendors on a variety of standardized hardware platforms. The challenge is to define a unified interface that clearly decouples the software instances from the underlying hardware, as represented by VMs and their hypervisors.

- **Performance trade-off:** Because the NFV approach is based on industry standard hardware (that is, avoiding any proprietary hardware such as acceleration engines), a probable decrease in performance has to be taken into account. The challenge is how to keep the performance degradation as small as possible by using appropriate hypervisors and modern software technologies, so that the effects on latency, throughput, and processing overhead are minimized.
- **Migration and coexistence with respect to legacy equipment:** The NFV architecture must support a migration path from today's proprietary physical network appliance-based solutions to more open standards-based virtual network appliance solutions. In other words, NFV must work in a hybrid network composed of classical physical network appliances and virtual network appliances. Virtual appliances must therefore use existing northbound Interfaces (for management and control) and interwork with physical appliances implementing the same functions.
- **Management and orchestration:** A consistent management and orchestration architecture is required. NFV presents an opportunity, through the flexibility afforded by software network appliances operating in an open and standardized infrastructure, to rapidly align management and orchestration northbound interfaces to well defined standards and abstract specifications.
- **Automation:** NFV will scale only if all the functions can be automated. Automation of process is paramount to success.
- **Security and resilience:** The security, resilience, and availability of their networks should not be impaired when VNFs are introduced.
- **Network stability:** Ensuring stability of the network is not impacted when managing and orchestrating a large number of virtual appliances between different hardware vendors and hypervisors. This is particularly important when, for example, virtual functions are relocated, or during reconfiguration events (for example, because of hardware and software failures) or because of cyber-attack.
- **Simplicity:** Ensuring that virtualized network platforms will be simpler to operate than those that exist today. A significant focus for network operators is simplification of the plethora of complex network platforms and support systems that have evolved over decades of network technology evolution, while maintaining continuity to support important revenue generating services.
- **Integration:** Network operators need to be able to "mix and match" servers from different vendors, hypervisors from different vendors, and virtual appliances from different vendors without incurring significant integration costs and avoiding lock-in. The ecosystem must offer integration services and maintenance and third-party support; it must be possible to resolve integration issues between several parties. The ecosystem will require mechanisms to validate new NFV products.

7.5 NFV Reference Architecture

Figure 7.7 provided a high-level view of the NFV framework. Figure 7.8 shows a more detailed

look at the ISG NFV reference architectural framework. You can view this architecture as consisting of four major blocks:

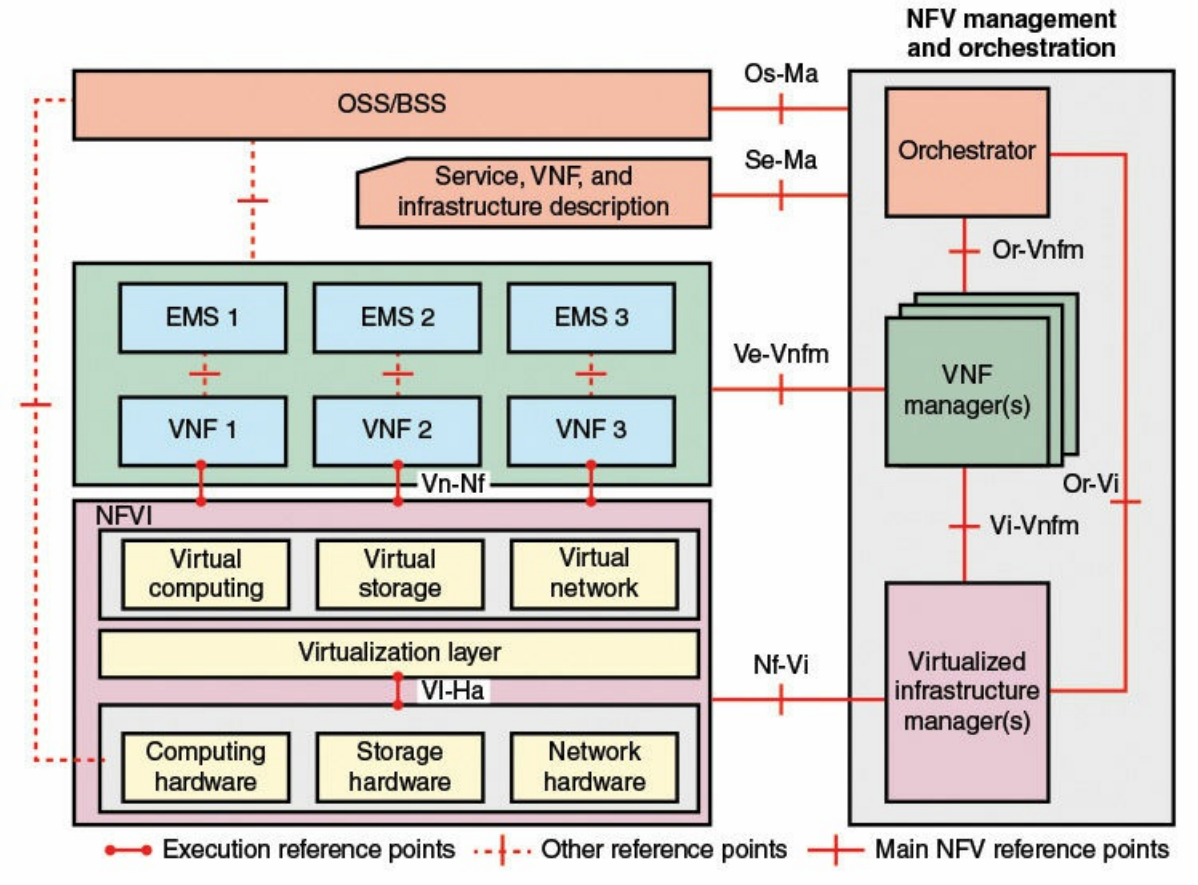


FIGURE 7.8 NFV Reference Architectural Framework

- **NFV infrastructure (NFVI):** Comprises the hardware and software resources that create the environment in which VNFs are deployed. NFVI virtualizes physical computing, storage, and networking and places them into resource pools.
- **VNF/EMS:** The collection of VNFs implemented in software to run on virtual computing, storage, and networking resources, together with a collection of element management systems (EMS) that manage the VNFs.
- **NFV management and orchestration (NFV-MANO):** Framework for the management and orchestration of all resources in the NFV environment. This includes computing, networking, storage, and VM resources.
- **OSS/BSS:** Operational and business support systems implemented by the VNF service provider.

It is also useful to view the architecture as consisting of three layers. The NFVI together with the virtualized infrastructure manager provide and manage the virtual resource environment and its underlying physical resources. The VNF layer provides the software implementation of network functions, together with element management systems and one or more VNF managers. Finally, there is a management, orchestration, and control layer consisting of OSS/BSS and the NFV

orchestrator.

NFV Management and Orchestration

The NFV management and orchestration facility includes the following functional blocks:

- **NFV orchestrator:** Responsible for installing and configuring new network services (NS) and virtual network function (VNF) packages, NS lifecycle management, global resource management, and validation and authorization of NFVI resource requests.
- **VNF manager:** Oversees lifecycle management of VNF instances.
- **Virtualized infrastructure manager:** Controls and manages the interaction of a VNF with computing, storage, and network resources under its authority, in addition to their virtualization.

Reference Points

Figure 7.8 also defines a number of reference points that constitute interfaces between functional blocks. The main (named) reference points and execution reference points are shown by solid lines and are in the scope of NFV. These are potential targets for standardization. The dashed line reference points are available in present deployments but might need extensions for handling network function virtualization. The dotted reference points are not a focus of NFV at present.

The main reference points include the following considerations:

- **Vi-Ha:** Marks interfaces to the physical hardware. A well-defined interface specification will facilitate for operators sharing physical resources for different purposes, reassigning resources for different purposes, evolving software and hardware independently, and obtaining software and hardware component from different vendors.
- **Vn-Nf:** These interfaces are APIs used by VNFs to execute on the virtual infrastructure. Application developers, whether migrating existing network functions or developing new VNFs, require a consistent interface the provides functionality and the ability to specify performance, reliability, and scalability requirements.
- **Nf-Vi:** Marks interfaces between the NFVI and the virtualized infrastructure manager (VIM). This interface can facilitate specification of the capabilities that the NFVI provides for the VIM. The VIM must be able to manage all the NFVI virtual resources, including allocation, monitoring of system utilization, and fault management.
- **Or-Vnfm:** This reference point is used for sending configuration information to the VNF manager and collecting state information of the VNFs necessary for network service lifecycle management.
- **Vi-Vnfm:** Used for resource allocation requests by the VNF manager and the exchange of resource configuration and state information.
- **Or-Vi:** Used for resource allocation requests by the NFV orchestrator and the exchange of resource configuration and state information.

- **Os-Ma:** Used for interaction between the orchestrator and the OSS/BSS systems.
- **Ve-Vnfm:** Used for requests for VNF lifecycle management and exchange of configuration and state information.
- **Se-Ma:** Interface between the orchestrator and a data set that provides information regarding the VNF deployment template, VNF forwarding graph, service-related information, and NFV infrastructure information models.

Implementation

As with SDN, success for NFV requires standards at appropriate interface reference points and open source software for commonly used functions. For several years, ISG NFV is working on standards for the various interfaces and components of NFV. In September of 2014, the Linux Foundation announced the Open Platform for NFV (OPNFV) project. OPNFV aims to be a carrier-grade, integrated platform that introduces new products and services to the industry more quickly. The key objectives of OPNFV are as follows:

Open Platform for NFV (OPNFV)

- Develop an integrated and tested open source platform that can be used to investigate and demonstrate core NFV functionality.
- Secure proactive participation of leading end users to validate that OPNFV releases address participating operators' needs.
- Influence and contribute to the relevant open source projects that will be adopted in the OPNFV reference platform.
- Establish an open ecosystem for NFV solutions based on open standards and open source software.
- Promote OPNFV as the preferred open reference platform to avoid unnecessary and costly duplication of effort.

OPNFV and ISG NFV are independent initiatives but it is likely that they will work closely together to assure that OPNFV implementations remain within the standardized environment defined by ISG NFV.

The initial scope of OPNFV will be on building NFVI, VIM, and including application programmable interfaces (APIs) to other NFV elements, which together form the basic infrastructure required for VNFs and MANO components. This scope is highlighted in Figure 7.9 as consisting of NFVI and VMI. With this platform as a common base, vendors can add value by developing VNF software packages and associated VNF manager and orchestrator software.

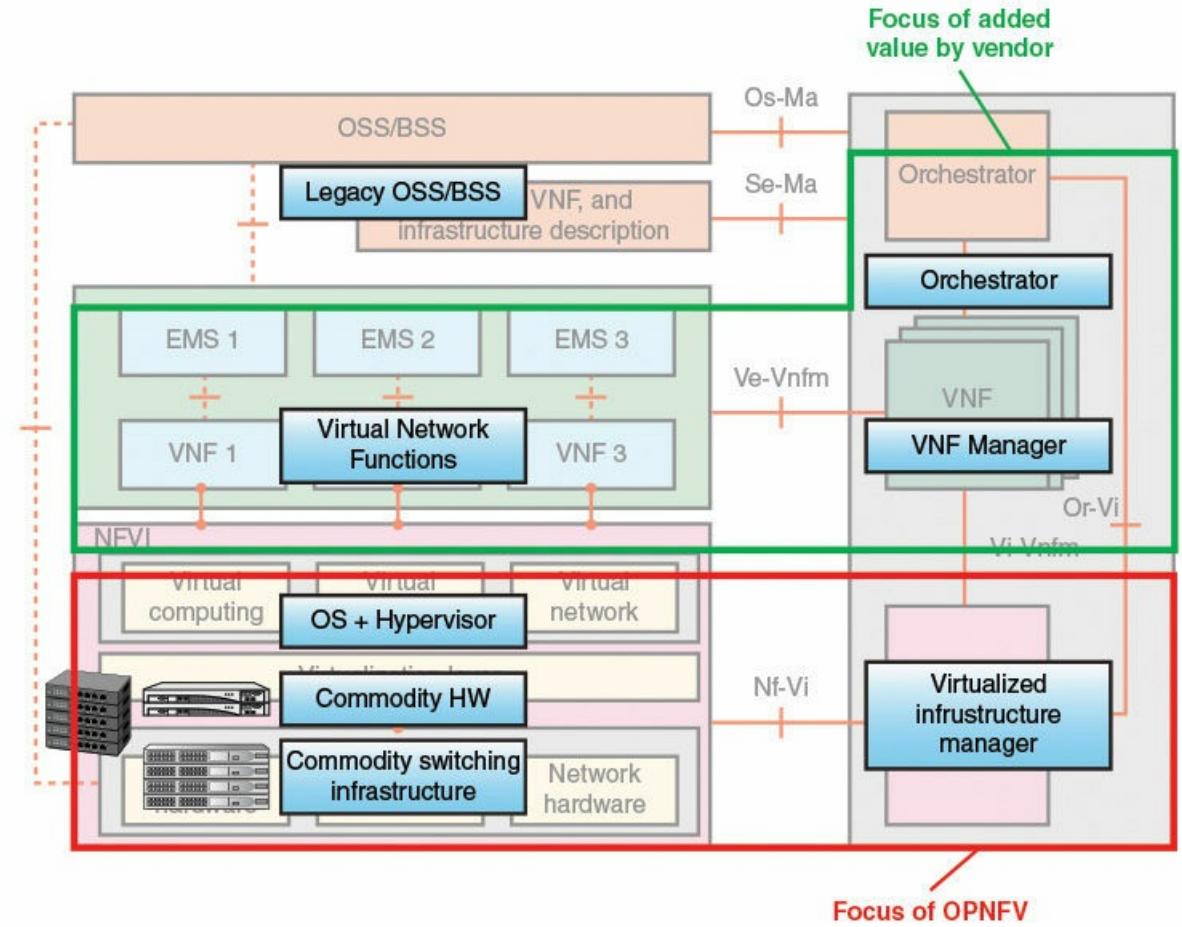


FIGURE 7.9 NFV Implementation

7.6 Key Terms

After completing this chapter, you should be able to define the following terms.

business support system (BSS)

capital expenditure (CapEx)

commercial off-the-shelf (COTS)

consolidation ratio

hardware virtualization

hypervisor

hypervisor domain

infrastructure-based virtual network

L2 virtual network

network functions virtualization (NFV)

Open Platform for NFV (OPNFV)

operational expenditure (OpEx)

point of presence (PoP)

scale down

scale in

Type 1 hypervisor

Type 2 hypervisor

virtual machine (VM)

virtual machine monitor (VMM)

7.7 References

ISGN12: ISG NFV. Network Functions Virtualization: An Introduction, Benefits, Enablers, Challenges & Call for Action. ISG NFV White Paper, October 2012.

Chapter 8. NFV Functionality

> The world has arrived at an age of cheap, complex devices of great reliability; and something is bound to come of it.
>
> —"As We May Think," Vannevar Bush, *The Atlantic*, July 1945

Chapter Objectives: **After studying this chapter, you should be able to**

- Explain the elements of the NFV infrastructure and their interrelationships.
- Understand key design issues related to virtualized network functions.
- Explain the purpose of and operation of NFV management and orchestration.
- Present an overview of important NFV use cases.
- Discuss the relationship between SDN and NFV.

This chapter concludes our discussion of network functions virtualization (NFV).

8.1 NFV Infrastructure

The heart of the NFV architecture is a collection of resources and functions known as the NFV infrastructure (NFVI). The NFVI encompasses three domains, as illustrated in Figure 8.1 and described in the list that follows.

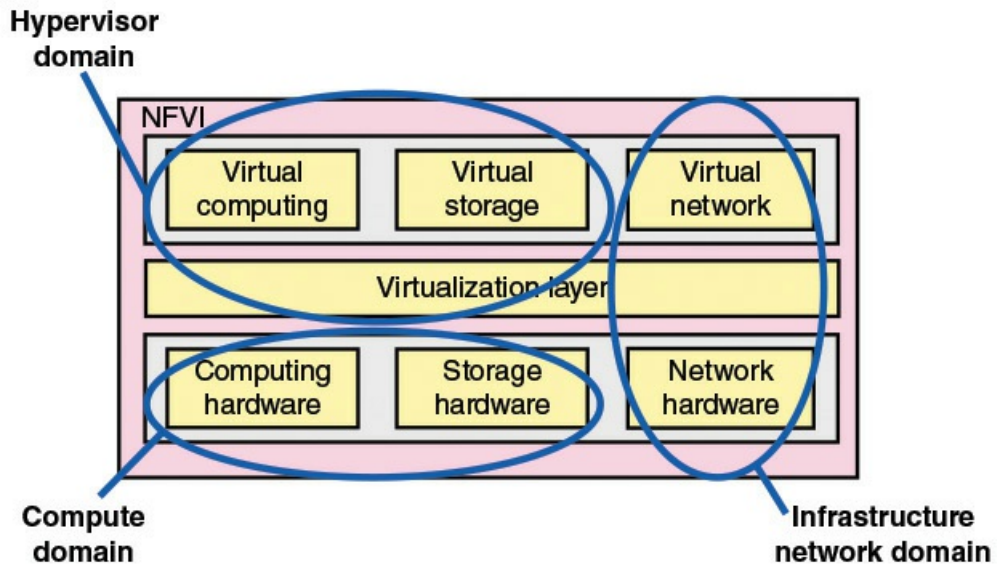


FIGURE 8.1 NFV Domains

- **Compute domain:** Provides commercial off-the-shelf (COTS) high-volume servers and storage.

- **Hypervisor domain:** Mediates the resources of the compute domain to the VMs of the software appliances, providing an abstraction of the hardware.
- **Infrastructure network domain (IND):** Comprises all the generic high volume switches interconnected into a network that can be configured to supply infrastructure network services.

Container Interface

Before proceeding with a discussion of NFVI, we need to clarify the concept of **container interface** as used in the Network Functions Virtualization Industry Standards Group (ISG NFV) documents. Unfortunately, the European Telecommunications Standards Institute (ETSI) documents use the term *container* in a different sense than that of container virtualization. The NFV Infrastructure document states that *container interface* should not be confused with *container* as used in the context of container virtualization as an alternative to full VMs. Further, the Infrastructure document states that some virtual network functions (VNFs) may be designed for hypervisor virtualization and other VNFs may be designed for container virtualization. With this clarification, the following examines the container interface concept.

The ETSI documents make a distinction between a functional block interface and a container interface, as follows:

- **Functional block interface:** An interface between two blocks of software that perform separate (perhaps identical) functions. The interface allows communication between the two blocks. The two functional blocks may or may not be on the same physical host.
- **Container interface:** An execution environment on a host system within which a functional block executes. The functional block is on the same physical host as the container that provides the container interface.

The concept of container interface is important because, in discussing VMs and VNFs within the NFV architecture, and how these functional blocks interact, it is easy to lose sight of the fact that all of these virtualized functions must execute on actual physical hosts.

Figure 8.2 relates container and functional block interfaces to the domain structure of NFVI.

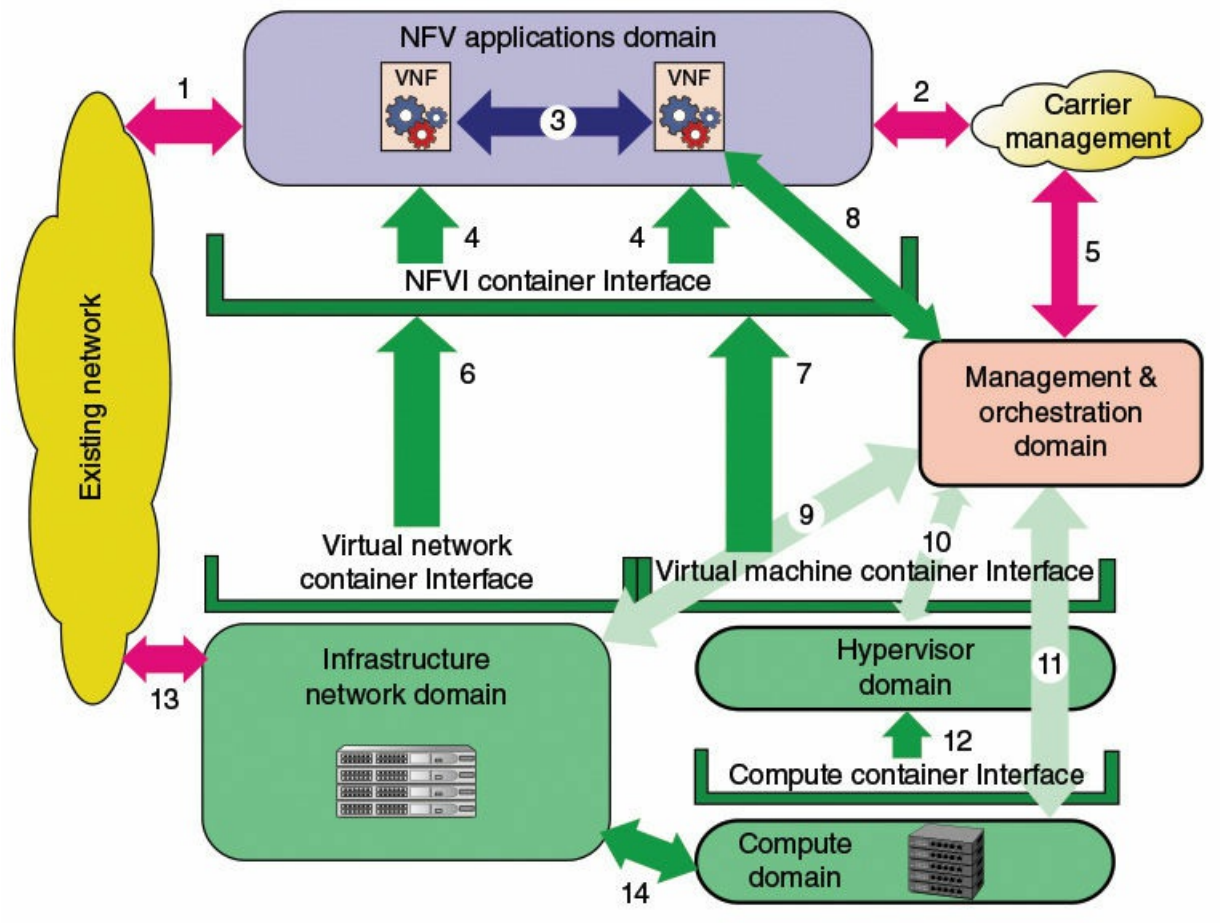


FIGURE 8.2 General Domain Architecture and Associated Interfaces

The ETSI NFVI Architecture Overview document makes the following points concerning this figure:

- The architecture of the VNFs is separated from the architecture hosting the VNFs (that is, the NFVI).
- The architecture of the VNFs may be divided into a number of domains with consequences for the NFVI and vice versa.
- Given the current technology and industrial structure, compute (including storage), hypervisors, and infrastructure networking are already largely separate domains and are maintained as separate domains within the NFVI.
- Management and orchestration tends to be sufficiently distinct from the NFVI as to warrant being defined as its own domain; however, the boundary between the two is often only loosely defined with functions such as element management functions in an area of overlap.
- The interface between the VNF domains and the NFVI is a container interface and not a functional block interface.
- The management and orchestration functions are also likely to be hosted in the NFVI (as VMs) and therefore also likely to sit on a container interface.

Figure 8.2 gives insight into the deployment of NFV. The user view of a network of interconnected VNFs is of a virtualized network in which the physical and lower level logical details are transparent. But the VNFs and logical links between VNFs are hosted on an NFVI container, which in turn is hosted on VM and VM containers running on physical hosts. Therefore, if we view the VNF architecture as having three layers (physical resource, virtualization, application), all three layers are present on a single physical host. Of course, functionality may be distributed across multiple computer and switch hosts, but all application software ultimately runs on the same physical host as the virtualization software. This is in contrast to software-defined networking (SDN), where by design the data plane functions and the control plane functions are on separate physical hosts. Application plane SDN functions may execute on the same host as the control plane functions but may also execute remotely on another host.

Table 8.1 describes the interfaces labeled in Figure 8.2; the numbers in the second column of the table correspond to the numbered arrows in the figure. Interfaces 4, 6, 7, and 12 are container interfaces, so that components on both side of the interface are executing on the same host. Interfaces 3, 8, 9, 10, 11, and 14 are functional block interfaces and, in most cases, the functional blocks on the two sides of the interface execute on different hosts. However, in some cases, some of the management and orchestration software may be hosted on a system that also hosts other NFVI components. Figure 8.2 also shows interfaces 1, 2, 5, and 13 to existing networks that have not implemented NFV. The NFV documents anticipate that typically NFV will be introduced over time into an enterprise facility, so that interaction with non-NFV network is necessary.

Interface Type	#	Description
NFVI container interfaces	4	Primary interface provided by the infrastructure to host VNFs. The applications may be distributed and the infrastructure provides virtual connectivity that interconnects the distributed components of an application.
VNF interconnect interfaces	3	Interfaces between VNFs. The specification of these interfaces does not include, and is transparent to, the way the infrastructure provides the connectivity service between the hosted functional blocks, however distributed.
VNF management and orchestration interface	8	Interface that allows the VNFs to request different resources of the infrastructure (for example, request new infrastructure connectivity services, allocate more compute resources, or activate/deactivate other VM components of the application).
Infrastructure container interfaces	6	Virtual network container interface: Interface to the connectivity services provided by the infrastructure. This container interface makes the infrastructure appear to NFV applications as instances of these connectivity services.
	7	Virtual machine container interface: Primary hosting interface on which VNF VMs run.
	12	Compute container interface: Primary hosting interface on which the hypervisor runs.
Infrastructure interconnect interfaces	9	Management and orchestration interface with the infrastructure network domain.
	10	Management and orchestration interface with the hypervisor domain.
	11	Management and orchestration interface with the compute domain.
	14	Network interconnect between the compute equipment and the infrastructure network equipment.
Legacy interconnect interfaces	1	Interface between the VNF and the existing network. This is likely to be higher layers of protocol only as all protocols provided by the infrastructure are transparent to the VNFs.
	2	Management of VNFs by existing management systems.
	5	Management of NFV infrastructure by existing management systems.
	13	Interface between the infrastructure network and the existing network. This is likely to be lower layers of protocol only because all protocols provided by VNFs are transparent to the infrastructure.

TABLE 8.1 Inter-Domain Interfaces Arising from Domain Architecture

Deployment of NFVI Containers

A single compute or network host can host multiple virtual machines (VMs), each of which can

host a single VNF. The single VNF hosted on a VM is referred to as a VNF component (VNFC). A network function may be virtualized by a single VNFC, or multiple VNFCs may be combined to form a single VNF. Part a of Figure 8.3 shows the organization of VNFCs on a single compute node. The compute container interface hosts a hypervisor, which in turn can host multiple VMs, each hosting a VNFC.

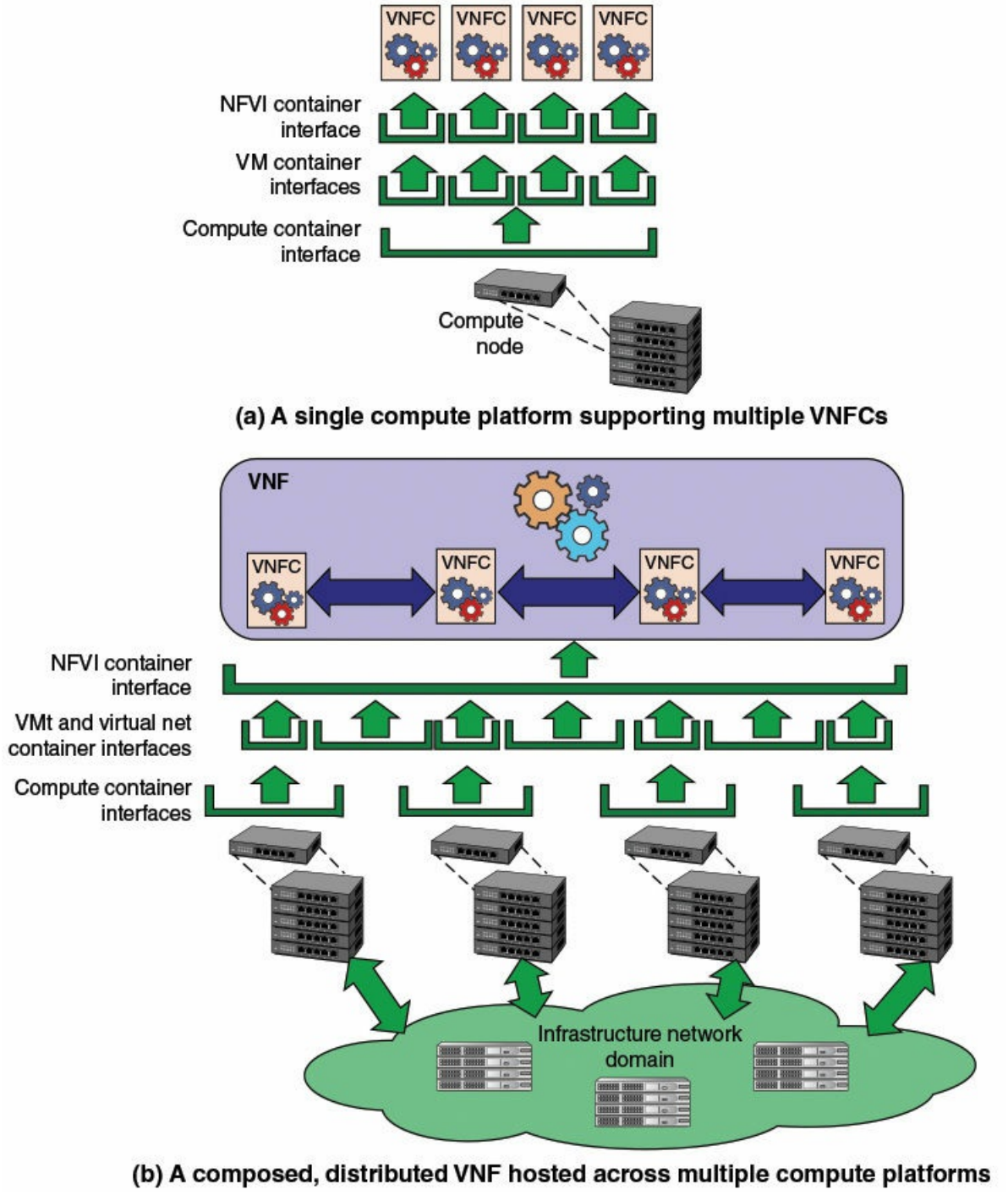


FIGURE 8.3 Deployment of NVFI Containers

When a VNF is composed of multiple VNFCs, it is not necessary that all the VNFCs execute in

the same host. As shown in part b of Figure 8.3, the VNFCs can be distributed across multiple compute nodes interconnected by network hosts forming the infrastructure network domain.

Logical Structure of NFVI Domains

The ISG NFV standards documents lay out the logical structure of the NFVI domains and their interconnections. The specifics of the actual implementation of the elements of this architecture will evolve in both open source and proprietary implementation efforts. The NFVI domain logical structure provides a framework for such development and identifies the interfaces between the main components, as shown in Figure 8.4.

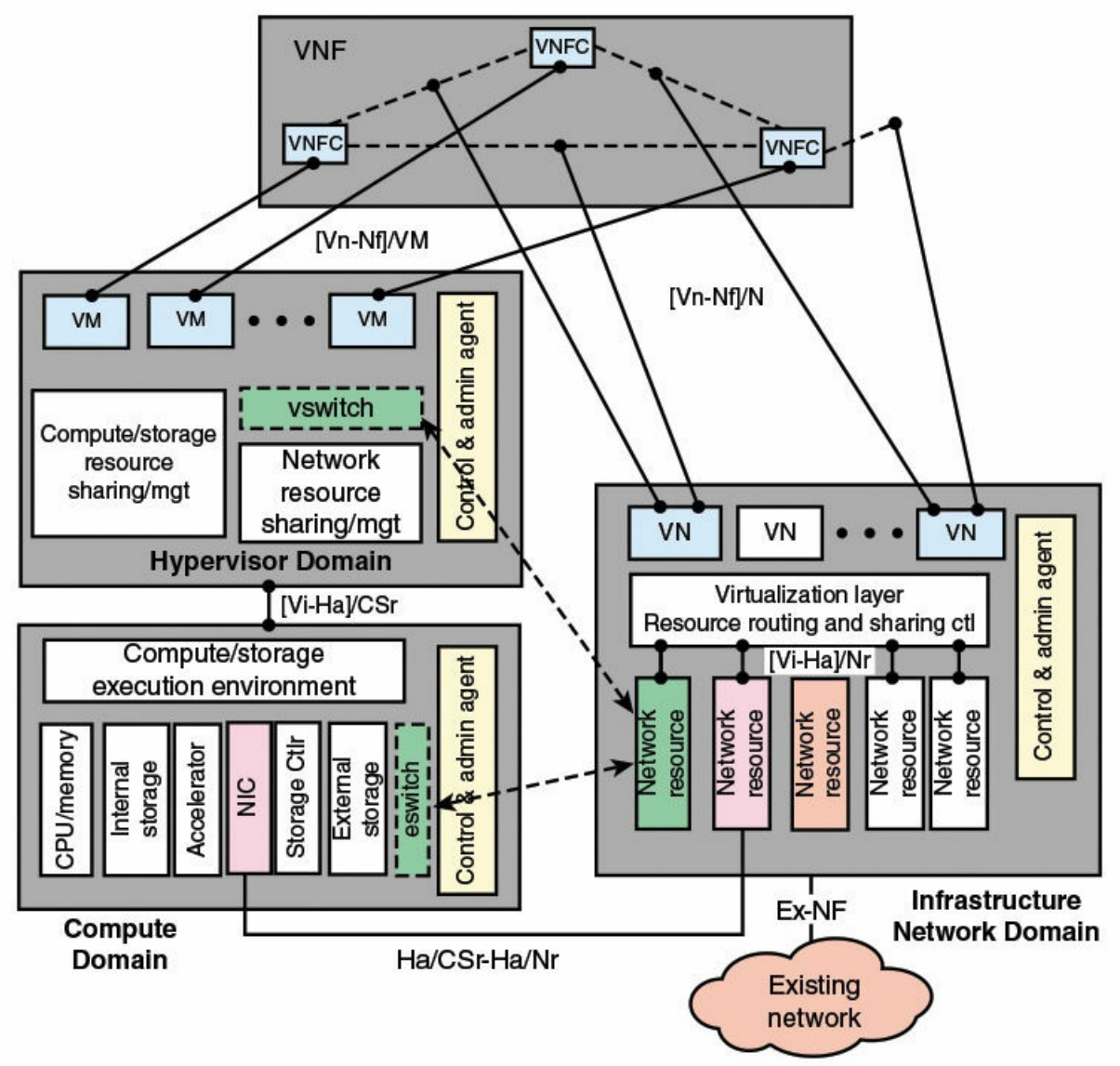


FIGURE 8.4 Logical Structure of NFVI Domains

Compute Domain

The principal elements in a typical compute domain may include the following:

- **CPU/memory:** A COTS processor, with main memory, that executes the code of the VNFC.
- **Internal storage:** Nonvolatile storage housed in the same physical structure as the processor, such as flash memory.
- **Accelerator:** Accelerator functions for security, networking, and packet processing may also be included.
- **External storage with storage controller:** Access to secondary memory devices.
- **Network interface card (NIC):** Provides the physical interconnection with the infrastructure network domain, which is labeled Ha/CSr-Ha/Nr and corresponds to interface 14 of Figure 8.2.
- **Control and admin agent:** Connects to the virtualized infrastructure manager (VIM); see Figure 7.8 in Chapter 7, "Network Functions Virtualization: Concepts and Architecture."
- **Eswitch:** Server embedded switch. The eswitch function, described in the following section, is implemented in the compute domain. However, functionally it forms an integral part of the infrastructure network domain.
- **Compute/storage execution environment:** This is the execution environment presented to the hypervisor software by the server or storage device ([VI-Ha]/CSr, interface 12 of Figure 8.2).

Eswitch

To understand the functionality of the eswitch, first note that, broadly speaking, VNFs deal with two different kinds of workloads:

- **Control plane workloads:** Concerned with signaling and control plane protocols such as BGP. Typically, these workloads are more processor rather than I/O intensive and do not place a significant burden on the I/O system.
- **Data plane workloads:** Concerned with the routing, switching, relaying or processing of network traffic payloads. Such workloads can require high I/O throughput.

In a virtualized environment such as NFV, all VNF network traffic would go through a virtual switch in the hypervisor domain, which invokes a layer of software between virtualized VNF software and host networking hardware. This can create a significant performance penalty. The purpose of the eswitch is to bypass the virtualization software and provide the VNF with a direct memory access (DMA) path to the NIC. The eswitch approach accelerates packet processing without any processor overhead.

NFVI Implementation Using Compute Domain Nodes

As suggested by Figure 8.3, a VNF consists of one or more logically connected VNFCs. The VNFCs run as software on hypervisor domain containers that in turn run on hardware in the compute domain. Although virtual links and networks are defined through the infrastructure

network domain, the actual implementation of network functions at the VNF level consists of software on compute domain nodes. The IND interfaces with the compute domain and not directly with the hypervisor domain or the VNFs. Again, this latter point is illustrated in Figure 8.3.

Before proceeding, we need to define the term *node*, which is used often in the ISG NFV documents. The documents define an **NFVI-Node** as collection of physical devices deployed and managed as a single entity, providing the NFVI functions required to support the execution environment for VNFs. NFVI nodes are in the compute domain and encompass the following types of compute domain nodes:

- **Compute node:** A functional entity which is capable of executing a generic computational instruction set (each instruction be being fully atomic and deterministic) in such a way that the execution cycle time is of the order of units to tens of nanoseconds irrespective of what specific state is required for cycle execution. In practical terms, this defines a compute node in terms of memory access time. A distributed system cannot meet this requirement as the time taken to access state stored in remote memory cannot meet this requirement.
- **Gateway node:** A single identifiable, addressable, and manageable element within an NFVI-Node that implements gateway functions. Gateway functions provide the interconnection between NFVI-PoPs and the transport networks. They also connect virtual networks to existing network components. A gateway may process packets going between different networks, such as removing headers and adding headers. A gateway may operate at the transport level, dealing with IP and data-link packets, or at the application level.
- **Storage node:** A single identifiable, addressable, and manageable element within an NFVI-Node that provides storage resource using compute, storage, and networking functions. Storage may be physically implemented in a variety of ways. It could, for example be implemented as a component within a compute node. An alternative approach is to implement storage nodes independent of the compute nodes as physical nodes within the NFVI-Node. An example of such a storage node may be a physical device accessible via a remote storage technology, such as Network File System (NFS) and Fibre Channel.
- **Network node:** A single identifiable, addressable, and manageable element within an NFVI-Node that provides networking (switching/routing) resources using compute, storage, and network forwarding functions.

A compute domain within an NFVI node will often be deployed as a number of interconnected physical devices. Physical compute domain nodes may include a number of physical resources, such as a multicore processor, memory subsystems, and NICs. An interconnected set of these nodes comprise one NFVI-Node and constitutes one NFVI point of presence (NFVI-PoP). An NFV provider might maintain a number of NFVI-PoPs at distributed locations, providing service to a variety of users, each of whom could implement their VNF software on compute domain nodes at various NFVI-PoP locations.

Table 8.2 lists some deployment scenarios suggested in the ISG NFV Compute Domain document. The scenarios include the following:

Deployment Scenario	Building	Host Hardware	Hypervisor	Guest VNF
Monolithic operator	N	N	N	N
Network operator hosting virtual network operators	N	N	N	N, N1, N2
Hosted network operator	H	H	H	N
Hosted communications providers	H	H	H	N1, N2, N3
Hosted communications and application providers	H	H	H	N1, N2, N3, P
Managed network service on customer premises	C	N	N	N
Managed network service on customer equipment	C	C	N	N

TABLE 8.2 Some Realistic Deployment Scenarios

- **Monolithic operator:** One organization owns and houses the hardware equipment and deploys and operates the VNFs and the hypervisors they run on. A private cloud or a data center are examples of this deployment model.
- **Network operator hosting virtual network operators:** Based on the monolithic operator scenario, with the addition that the monolithic operator host other virtual network operators within the same facility. A hybrid cloud is an example of this deployment model.
- **Hosted network operator:** An IT services organization (for example, HP, Fujitsu) operates the compute hardware, infrastructure network, and hypervisors on which a separate network operator (for example, BT, Verizon) runs VNFs. These are physically secured by the IT services organization.
- **Hosted communications providers:** Similar to the hosted network operator scenario, but in this case multiple communications providers are hosted. A community cloud is an example of this deployment model.
- **Hosted communications and application providers:** Similar to the previous scenario. In addition to host network and communications providers, servers in a data center facility are offered to the public for deploying virtualized applications. A public cloud is an example of this deployment model.
- **Managed network service on customer premises:** Similar to the monolithic operator scenario. In this case, the NFV provider's equipment is housed on the customer's premises. One example of this model is a remotely managed gateway in a residential or enterprise location. Another example is remotely managed networking equipment such as firewalls or virtual private network gateways.
- **Managed network service on customer equipment:** Similar to the monolithic operator scenario. In this case, the equipment is housed on the customer's premises on customer equipment. This scenario could be used for managing an enterprise network. A private cloud could also be deployed in this fashion.

Note

The different letters represent different companies or organizations, and are chosen to represent different roles (for example, H = hosting provider, N = network operator, P = public, C = customer). The numbered network operators (N1, N2, and so on) represent multiple individual hosted network operators.

See the discussion on the National Institute of Standards and Technology (NIST) cloud computing models for a definition of the four cloud types referenced in the preceding list.

➔ *See Chapter 13, "Cloud Computing"*

Hypervisor Domain

The hypervisor domain is a software environment that abstracts hardware and implements services, such as starting a VM, terminating a VM, acting on policies, scaling, live migration, and high availability. The principal elements in the hypervisor domain are the following:

- **Compute/storage resource sharing/management:** Manages these resources and provides virtualized resource access for VMs.
- **Network resource sharing/management:** Manages these resources and provides virtualized resource access for VMs.
- **Virtual machine management and API:** This provides the execution environment of a single VNFC instance ([Vn-Nf]/VM, interface 7 of Figure 8.2).
- **Control and admin agent:** Connects to the virtualized infrastructure manager (VIM); see Figure 7.8.
- **Vswitch:** The vswitch function, described in the next paragraph, is implemented in the hypervisor domain. However, functionally it forms an integral part of the infrastructure network domain.

The vswitch is an Ethernet switch implemented by the hypervisor that interconnects virtual NICs of VMs with each other and with the NIC of the compute node. If two VNFs are on the same physical server, they would be connected through the same vswitch. If two VNFs are on different servers, the connection passes through the first vswitch to the NIC and then to an external switch. This switch forwards the connection to the NIC of the desired server. Finally, this NIC forwards it to its internal vswitch and then to the destination VNF.

Infrastructure Network Domain

The infrastructure network domain (IND) performs a number of roles. It provides

- The communication channel between the VNFCs of a distributed VNF
- The communications channel between different VNFs
- The communication channel between VNFs and their orchestration and management

- The communication channel between components of the NFVI and their orchestration and management
- The means of remote deployment of VNFCs
- The means of interconnection with the existing carrier network

Figure 8.2 illustrates key reference points defined for the IND. As already mentioned, Ha/CSr-Ha/Nr defines the interface between the IND and the servers/storage of the compute domain, connecting the NIC in the compute domain to a network resource in the infrastructure network domain. Ex-Nf is the reference point between any existing/nonvirtualized network (interface 13 of Figure 8.2). Reference point [VI-HA]/Nr is the interface between the hardware network resources of the IND and the virtualization layer. The virtualization layer provides container interfaces for virtual network entities. The [Vn-Nf]/N reference point (interface 7 of Figure 8.2) is the virtual network (VN) container interface (for example, a link or a LAN) for carrying communication between VNFC instances. Note that a single VN can support communication between more than a single pairing of VNFC instances (for example, a LAN).

There is an important distinction to be made between the virtualization function provided by the hypervisor domain and that provided by the infrastructure network domain. Virtualization in the hypervisor domain uses VM technology to create an execution environment for individual VNFCs. Virtualization in IND creates virtual networks for interconnecting VNFCs with each other and with network nodes outside the NFV ecosystem. These latter types of nodes are called physical network functions (PNFs).

Virtual Networks

Before proceeding, we need to clarify how the term *virtual network* is used in the ISG NFV documents. In general terms, a virtual network is an abstraction of physical network resources as seen by some upper software layer. Virtual network technology enables a network provider to support multiple virtual networks that are isolated from one another. Users of a single virtual network are not aware of the details of the underlying physical network or of the other virtual network traffic sharing the physical network resources. Two common approaches for creating virtual networks are (1) protocol-based methods that define virtual networks based on fields in protocol headers, and (2) virtual-machine-based methods, in which networks are created among a set of VMs by the hypervisor. The NFVI network virtualization combines both these forms.

L2 Versus L3 Virtual Networks

Protocol-based virtual networks can be classified by whether they are defined at protocol Layer 2 (L2), which is typically the LAN media access control (MAC) layer, or Layer 3 (L3), which is typically the Internet Protocol (IP). With an L2 VN, a virtual LAN is identified by a field in the MAC header, such as the MAC address or a virtual LAN ID field inserted into the header. So, for example, within a data center, all the servers and end systems connected to a single Ethernet switch could support virtual LANs among the connected devices. Now suppose there are IP routers connecting segments of the data center, as illustrated in Figure 8.5. Normally, an IP router will strip off the MAC header of incoming Ethernet frames and insert a new MAC header when forwarding the packet to the next network. The L2 VN could be extended across this router

only if the router had additional capability to support the L2 VN, such as being able to reinsert the virtual LAN ID field in the outgoing MAC frame. Similarly, if an enterprise had two data centers connected by a router and a dedicated line, that router would need the L2 VN capability to extend a VN.

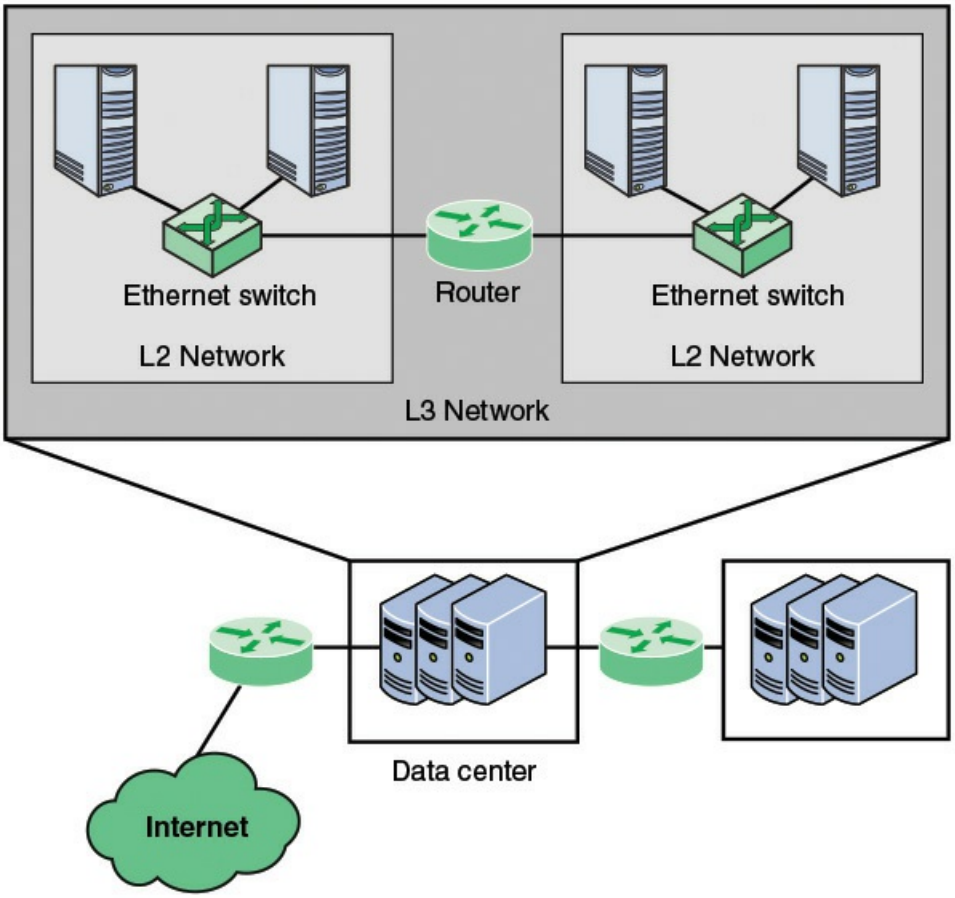


FIGURE 8.5 Levels of Network Virtualization

An L3 VN makes use of one or more fields in the IP header. A good example of this is the virtual private network (VPN) defined using IPsec. Packets traveling on a VPN are encapsulated in a new outer IP header and the data are encrypted so that VPN traffic is isolated and protected as it transits third-party network such as the Internet.

Chapter 9, "Network Virtualization," covers virtual LANs and VPNs in more detail.

NFVI Virtual Network Alternatives

ISG NFV defines a virtual network as the network construct that provides network connectivity to one or more VNFs that are hosted on the NFVI. Therefore, the concept of a virtual network that extends beyond the NFV infrastructure is not currently addressed. In NFV, a virtual network is a network among VNFs.

The Network Domain document indicates that three approaches are envisioned for providing a virtual network service:

- Infrastructure-based VNs
- Layered VNs using virtual overlays
- Layered VNs using virtual partitioning

A facility can use one or a combination of these approaches.

The **infrastructure-Based VN** uses the native networking functions of the NFVI compute and networking components. The address space is partitioned so that VNF membership in a VN is defined by IP address. The IND document gives the following example of an L3 infrastructure-based VN:

- Each VNF is assigned its own unique IP address that does not overlap with any other address of elements within the NFVI.
- Logical partitioning of the VNFs into their VNs is achieved by managing access control lists in the L3 forwarding function in each compute node.
- The L3 forwarding between VNFs and the physical fabric can then be handled by the L3 forwarding information base running on the hosting compute node.
- Control plane solutions, such as Border Gateway Protocol (BGP), can be used to advertise reachability of the VNFs to other compute hosts.

The other two approaches are referred to as *layered virtual network approaches*. These approaches allow overlapping address spaces. That is, a VNF may participate in more than one VN using the same IP address. The virtualization layer of the IND essentially creates private topologies on the underlying NFVI network fabric, using either virtual overlays or virtual partitioning.

The **virtual overlay VN** uses the concept of an overlay network. In essence, an overlay network is a logical network that is built on the top of another network. Nodes in the overlay network can be thought of as being connected by virtual or logical links, each of which corresponds to a path, perhaps through many physical links in the underlying network. However, overlay networks do not have the ability to control the routing between two overlay network nodes. In the NFV context, the overlay networks are the VNs used by the VNFs and the underlay network consists of the infrastructure network resources. These overlay networks are normally created by edge nodes which have a dual personality, participating in both the creation of the virtual networks and also acting as infrastructure network resources. In contrast, the core nodes of the infrastructure network only participate in the infrastructure network and have no overlay awareness. The L2 and L3 virtual networks discussed earlier fit into this category.

The virtual partitioning VN approach directly integrates VNs, called virtual network partitions in this context, into the infrastructure network on an end-to-end basis. Discrete virtual topologies are built in both the edge and core nodes of the infrastructure network for each virtual network. This can consist of per virtual network forwarding tables, logical links and even control planes on an end-to-end basis across the infrastructure network.

8.2 Virtualized Network Functions

A VNF is a virtualized implementation of a traditional network function. Table 8.3 contains

examples of functions that could be virtualized.

Network Element	Function
Switching elements	Broadband network gateways, carrier grade Network Address Translation (NAT), routers
Mobile network nodes	Home location register/home subscriber server, gateway, General Packet Radio Service (GPRS) Protocol support node, radio network controller, various node B functions
Customer premises equipment	Home routers, set-top boxes
Tunneling gateway elements	IPsec / Secure Sockets Layer (SSL) virtual private network gateways
Traffic analysis	Deep packet inspection (DPI), quality of experience (QoE) measurement
Assurance	Service assurance, service level agreement (SLA) monitoring, testing and diagnostics
Signaling	Session border controllers, IP multimedia subsystem (IMS) components
Control plane / access functions	AAA (authentication, authorization, and accounting) servers, policy control and charging platforms, Dynamic Host Configuration Protocol (DHCP) servers
Application optimization	Content delivery networks, cache servers, load balancers, accelerators
Security	Firewalls, virus scanners, intrusion detection systems, spam protection

TABLE 8.3 Potential Network Functions to Be Virtualized

VNF Interfaces

As discussed earlier, a VNF consists of one or more VNF components (VNFCs). The VNFCs of a single VNF are connected internal to the VNF. This internal structure is not visible to other VNFs or to the VNF user.

Figure 8.6 shows the interfaces relevant to a discussion of VNFs as described in the list that follows.

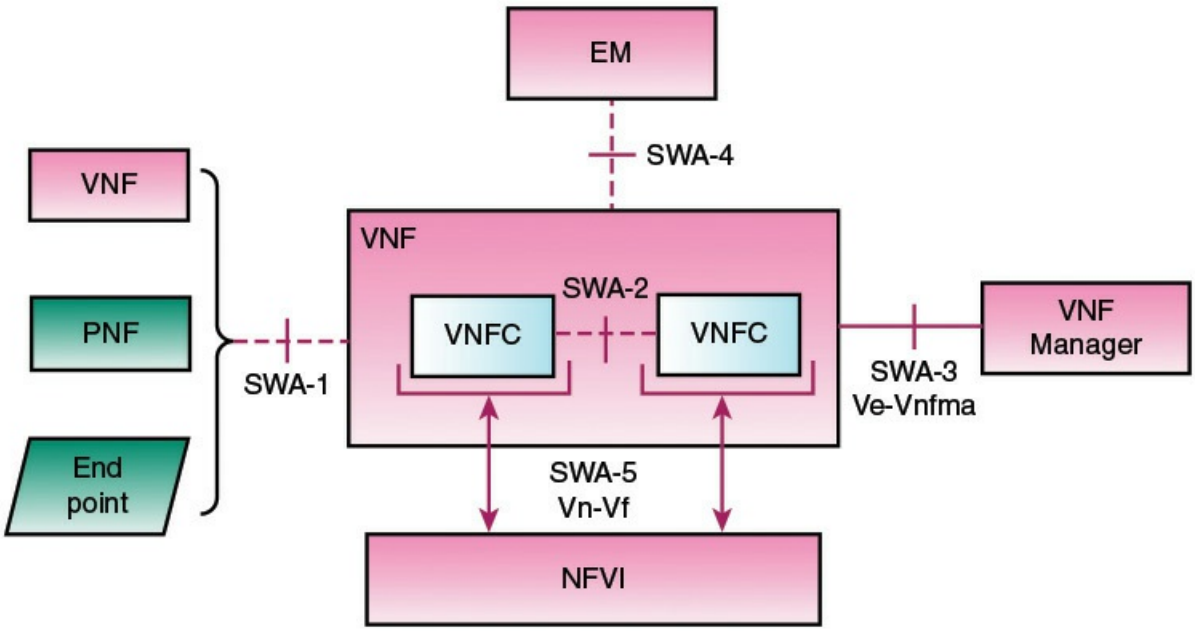


FIGURE 8.6 VNF Functional View

- **SWA-1:** This interface enables communication between a VNF and other VNFs, PNFs, and endpoints. Note that the interface is to the VNF as a whole and not to individual VNFCs. SWA-1 interfaces are logical interfaces that primarily make use of the network connectivity services available at the SWA-5 interface.
- **SWA-2:** This interface enables communications between VNFCs within a VNF. This interface is vendor specific and therefore not a subject for standardization. This interface may also make use of the network connectivity services available at the SWA-5 interface. However, if two VNFCs within a VNF are deployed on the same host, other technologies may be used to minimize latency and enhance throughput, as described below.
- **SWA-3:** This is the interface to the VNF manager within the NFV management and orchestration module. The VNF manager is responsible for lifecycle management (creation, scaling, termination, and so on). The interface typically is implemented as a network connection using IP.
- **SWA-4:** This is the interface for runtime management of the VNF by the element manager.
- **SWA-5:** This interface describes the execution environment for a deployable instance of a VNF. Each VNFC maps to a virtualized container interface to a VM.

VNFC to VNFC Communication

As mentioned earlier, the internal structure of a VNF, in terms of multiple VNFCs, is not exposed externally. The VNF appears as a single functional system in the network it supports. However, internal connectivity between VNFCs within the same VNF or across co-located VNFs needs to be specified by the VNF provider, supported by the NFVI, and managed by the

VNF manager. The VNF Architecture document describes a number of architecture design models that are intended to provide desired performance and quality of service (QoS), such as access to storage or compute resources. One of the most important of these design models relates to communication between VNFCs.

Figure 8.7, from the ETSI VNF Architecture document, illustrates six scenarios using different network technologies to support communication between VNFCs:

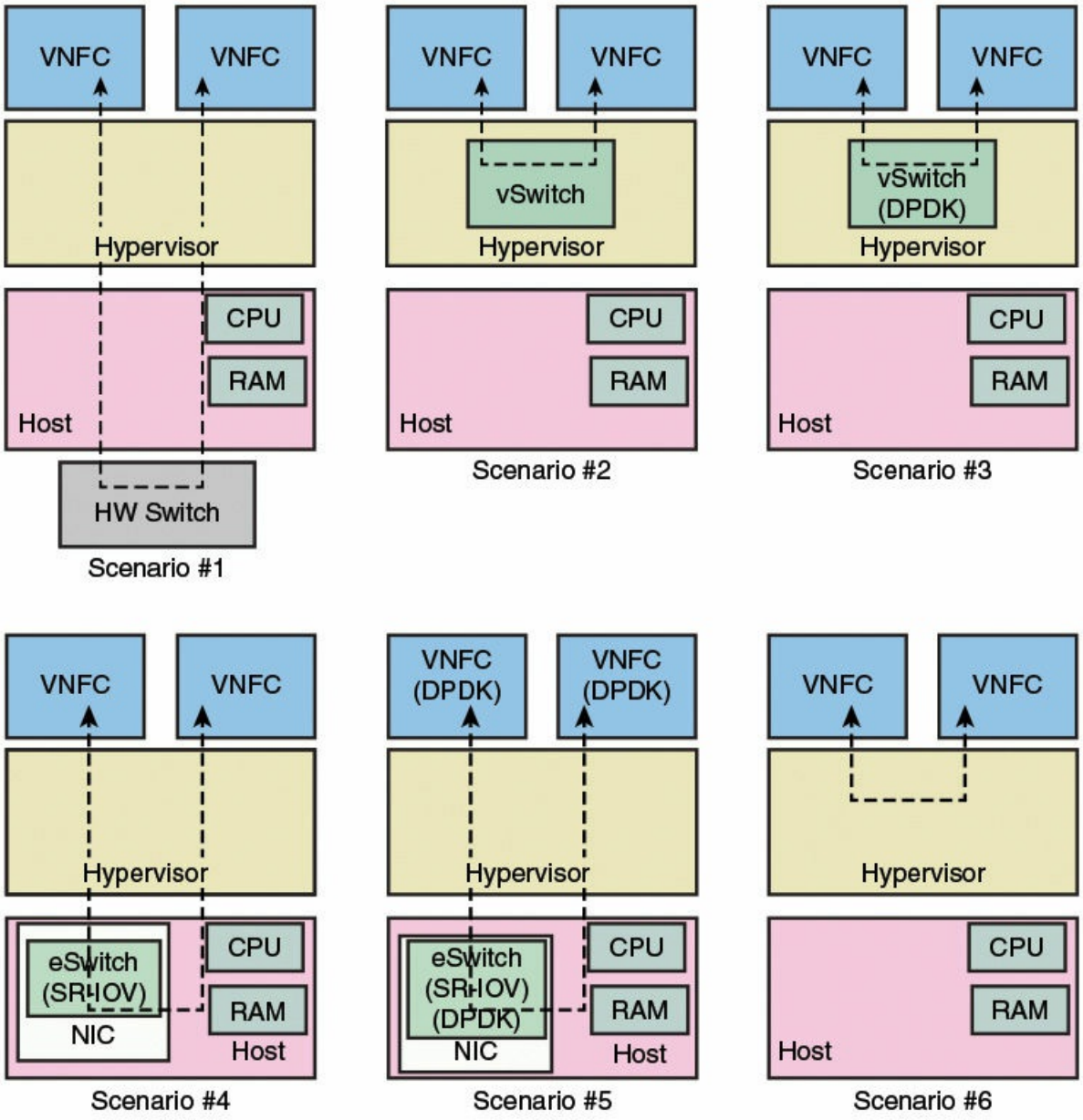


FIGURE 8.7 VNFC to VNFC Communication

1. Communication through a hardware switch. In this case, the VMs supporting the VNFCs bypass the hypervisor to directly access the physical NIC. This provides enhanced performance for VNFCs on different physical hosts.
2. Communication through the vswitch in the hypervisor. This is the basic method of

communication between co-located VNFCs but does not provide the QoS or performance that may be required for some VNFs.

3. Greater performance can be achieved by using appropriate data processing acceleration libraries and drivers compatible with the CPU being used. The library is called from the vswitch. An example of a suitable commercial product is the Data Plane Development Kit (DPDK), which is a set of data plane libraries and network interface controller drivers for fast packet processing on Intel architecture platforms. Scenario 3 assumes a Type 1 hypervisor (see Figure 7.3).
4. Communication through an embedded switch (eswitch) deployed in the NIC with Single Root I/O Virtualization (SR-IOV). SR-IOV is a PCI-SIG specification that defines a method to split a device into multiple PCI express requester IDs (virtual functions) in a fashion that allows an I/O memory management unit (MMU) to distinguish different traffic streams and apply memory and interrupt translations so that these traffic streams can be delivered directly to the appropriate VM, and in a way that prevents nonprivileged traffic flows from impacting other VMs.
5. Embedded switch deployed in the NIC hardware with SR-IOV, and with data plane acceleration software deployed in the VNFC.
6. A serial bus connects directly two VNFCs that have extreme workloads or very low-latency requirements. This is essentially an I/O channel means of communication rather than a NIC means.

VNF Scaling

An important property of VNFs is referred to as elasticity, which simply means the ability to **scale up**/down or **scale out**/in. Every VNF has associated with it an elasticity parameter of no elasticity, scale up/down only, scale out/in only, or both scale up/down and scale out/in.

A VNF is scaled by scaling one or more of its constituent VNFCs. Scale out/in is implemented by adding/removing VNFC instances that belong to the VNF being scaled. Scale up/down is implemented by adding/removing resources from existing VNFC instances that belong to the VNF being scaled.

8.3 NFV Management and Orchestration[1]

1 Some of the material in this section is based on [KHAN15].

The NFV management and orchestration (MANO) component of NFV has as its primary function the management and orchestration of an NFV environment. This task, by itself, is complex. Further complicating MANO functionality is its need to interoperate with and cooperate with existing operations support systems (OSS) and business support systems (BSS) in providing management functionality for customers whose networking environment consists of a mixture of physical and virtual elements.

Figure 8.8, from the ETSI MANO document, shows the basic structure of NFV-MANO and its key interfaces. As can be seen, there are five management blocks: three within NFV-MANO,

EMS associated with VNFs, and OSS/BSS. These two latter blocks are not part of MANO but do exchange information with MANO for the purpose of the overall management of a customer's networking environment.

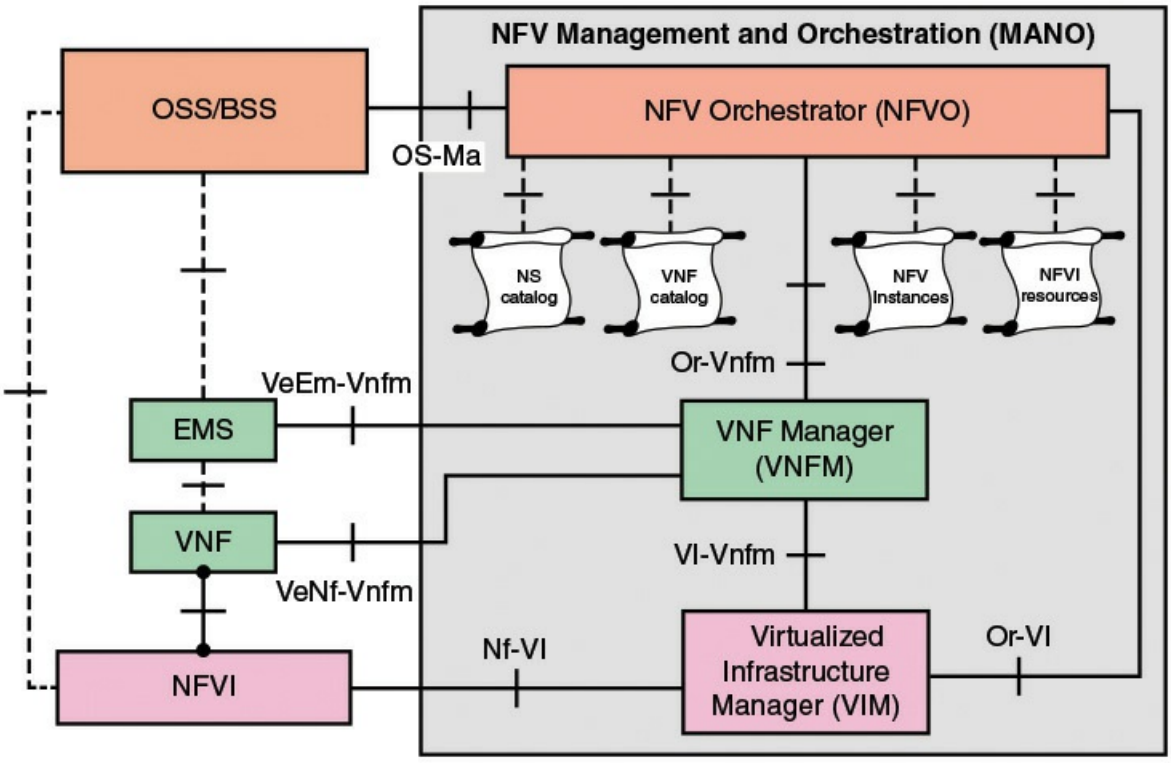


FIGURE 8.8 The NFV-MANO Architectural Framework with Reference Points

Virtualized Infrastructure Manager

Virtualized infrastructure management (VIM) comprises the functions that are used to control and manage the interaction of a VNF with computing, storage, and network resources under its authority, as well as their virtualization. A single instance of a VIM is responsible for controlling and managing the NFVI compute, storage, and network resources, usually within one operator's infrastructure domain. This domain could consist of all resources within an NFVI-PoP, resources across multiple NFVI-PoPs, or a subset of resources within an NFVI-PoP. To deal with the overall networking environment, multiple VIMs within a single MANO may be needed.

A VIM performs the following:

- **Resource management, in charge of the**
- Inventory of software (for example, hypervisors), computing, storage and network resources dedicated to NFV infrastructure.
- Allocation of virtualization enablers, for example, VMs onto hypervisors, compute resources, storage, and relevant network connectivity
- Management of infrastructure resource and allocation, for example, increase resources to

VMs, improve energy efficiency, and resource reclamation

- **Operations, for**
- Visibility into and management of the NFV infrastructure
- Root cause analysis of performance issues from the NFV infrastructure perspective
- Collection of infrastructure fault information
- Collection of information for capacity planning, monitoring, and optimization

Virtual Network Function Manager

A VNF manager (VNFM) is responsible for VNFs. Multiple VNFMs may be deployed; a VNFM may be deployed for each VNF, or a VNFM may serve multiple VNFs. Among the functions that a VNFM performs are the following:

- VNF instantiation, including VNF configuration if required by the VNF deployment template (for example, VNF initial configuration with IP addresses before completion of the VNF instantiation operation)
- VNF instantiation feasibility checking, if required
- VNF instance software update/upgrade
- VNF instance modification
- VNF instance scaling out/in and up/down
- VNF instance-related collection of NFVI performance measurement results and faults/events information, and correlation to VNF instance-related events/faults
- VNF instance assisted or automated healing
- VNF instance termination
- VNF lifecycle management change notification
- Management of the integrity of the VNF instance through its lifecycle
- Overall coordination and adaptation role for configuration and event reporting between the VIM and the EM

NFV Orchestrator

The NFV orchestrator (NFVO) is responsible for resource orchestration and network service orchestration.

Resource orchestration manages and coordinates the resources under the management of different VIMs. NFVO coordinates, authorizes, releases and engages NFVI resources among different PoPs or within one PoP. This does so by engaging with the VIMs directly through their northbound APIs instead of engaging with the NFVI resources directly.

Network services orchestration manages/coordinates the creation of an end-to-end service that

involves VNFs from different VNFMs domains. Service orchestration does this in the following way:

- It creates end-to-end service between different VNFs. It achieves this by coordinating with the respective VNFMs so that it does not need to talk to VNFs directly. An example is creating a service between the base station VNFs of one vendor and core node VNFs of another vendor.
- It can instantiate VNFMs, where applicable.
- It does the topology management of the network services instances (also called VNF forwarding graphs).

Repositories

Associated with NFVO are four repositories of information needed for the management and orchestration functions:

- **Network services catalog:** List of the usable network services. A deployment template for a network service in terms of VNFs and description of their connectivity through virtual links is stored in NS catalog for future use.
- **VNF catalog:** Database of all usable VNF descriptors. A VNF descriptor (VNFD) describes a VNF in terms of its deployment and operational behavior requirements. It is primarily used by VNFM in the process of VNF instantiation and lifecycle management of a VNF instance. The information provided in the VNFD is also used by the NFVO to manage and orchestrate network services and virtualized resources on NFVI.
- **NFV instances:** List containing details about network services instances and related VNF instances.
- **NFVI resources:** List of NFVI resources utilized for the purpose of establishing NFV services.

Element Management

The element management is responsible for fault, configuration, accounting, performance, and security (FCAPS) management functionality for a VNF. These management functions are also the responsibility of the VNFM. But EM can do it through a proprietary interface with the VNF in contrast to VNFM. However, EM needs to make sure that it exchanges information with VNFM through open reference point (VeEm-Vnfm). The EM may be aware of virtualization and collaborate with VNFM to perform those functions that require exchange of information regarding the NFVI resources associated with VNF. EM functions include the following:

- Configuration for the network functions provided by the VNF
- Fault management for the network functions provided by the VNF
- Accounting for the usage of VNF functions
- Collecting performance measurement results for the functions provided by the VNF

- Security management for the VNF functions

OSS/BSS

The OSS/BSS are the combination of the operator's other operations and business support functions that are not otherwise explicitly captured in the present architectural framework, but are expected to have information exchanges with functional blocks in the NFV-MANO architectural framework. OSS/BSS functions may provide management and orchestration of legacy systems and may have full end-to-end visibility of services provided by legacy network functions in an operator's network.

In principle, it would be possible to extend the functionalities of existing OSS/BSS to manage VNFs and NFVI directly, but that may be a proprietary implementation of a vendor. Because NFV is an open platform, managing NFV entities through open interfaces (as that in MANO) makes more sense. The existing OSS/BBS, however, can add value to the NFV MANO by offering additional functions if they are not supported by a certain implementation of NFV MANO. This is done through an open reference point (Os-Ma) between NFV MANO and existing OSS/BSS.

8.4 NFV Use Cases

ISG NFV has developed a representative set of service models and high-level use cases that may be addressed by NFV. These use cases are intended to drive further development of standards and products for network-wide implementation. The Use Cases document identifies and describes a first set of service models and high-level use cases that represent, in the view of NFV ISG member companies, important service models and initial fields of application for NFV, and that span the scope of technical challenges being addressed by the NFV ISG.

There are currently nine use cases, which can be divided into the categories of architectural use cases and service-oriented use cases, as described in Table 8.4.

Use Case	Description
Architectural Use Cases	
Network Functions Virtualization Infrastructure as a Service (NFVIaaS)	Provides an approach to mapping the cloud computing service models Infrastructure as a Service (IaaS) and Network as a Service (NaaS) as elements with the NFVI when it is provided as a service
Virtual Network Function as a Service (VNFaaS)	Application of virtualization to the enterprise to enable a lower-cost model in which the operator provides services and the enterprise consumes the resources it requires
Virtual Network Platform as a Service (VNPaaS)	Similar to VNFaaS, but in this use case the enterprise has the opportunity to host and introduce VNF instances on their own
VNF Forwarding Graphs	Building end-to-end services by composition
Service-Oriented Use Cases	
Virtualization of Mobile Core Network and IMS	Encompasses virtualization of the mobile packet core and IMS
Virtualization of the Mobile Base Station	Encompasses virtualization of the mobile RAN onto standard servers
Virtualization of the Home Environment	Encompasses virtualization of CPE, such as set-top boxes residential gateways
Virtualization of CDNs (vCDN)	Encompasses virtualization of content delivery networks (CDNs) to enable a more scalable and lower cost off-peak operational model
Fixed Access NFV	Encompasses virtualization of fixed network access infrastructure to optimize deployment costs and enable co-location with wireless access nodes

TABLE 8.4 ETSI NFV Use Cases

Architectural Use Cases

The four architectural use cases focus on providing general-purpose services and applications based on the NFVI architecture.

NFVI as a Service

NFVIaaS is a scenario in which a service provider implements and deploys an NFVI that may be used to support VNFs both by the NFVIaaS provider and by other network service providers. For the NFVIaaS provider, this service provides for economies of scale. The infrastructure is sized to support the provider's own needs for deploying VNFs and extra capacity that can be sold to other providers. The NFVIaaS customer can offer services using the NFVI of another service provider. The NFVIaaS customer has flexibility in rapidly deploying VNFs, either for new services or to scale out existing services. Cloud computing providers may find this service particularly attractive.

Figure 8.9 provides an example [ONF14]. Service provider X offers a virtualized load balancing service. Some of carrier X's customers need load balancing services at locations where X does not maintain NFVI, but where service provider Z does. NFVIaaS offers a means for carrier Z to lease NFV infrastructure (computer, network, hypervisors, and so on) to service provider X, which gives the latter access to infrastructure that would otherwise be prohibitively expensive to obtain. Through leasing, such capacity is available on demand, and can be scaled as needed.

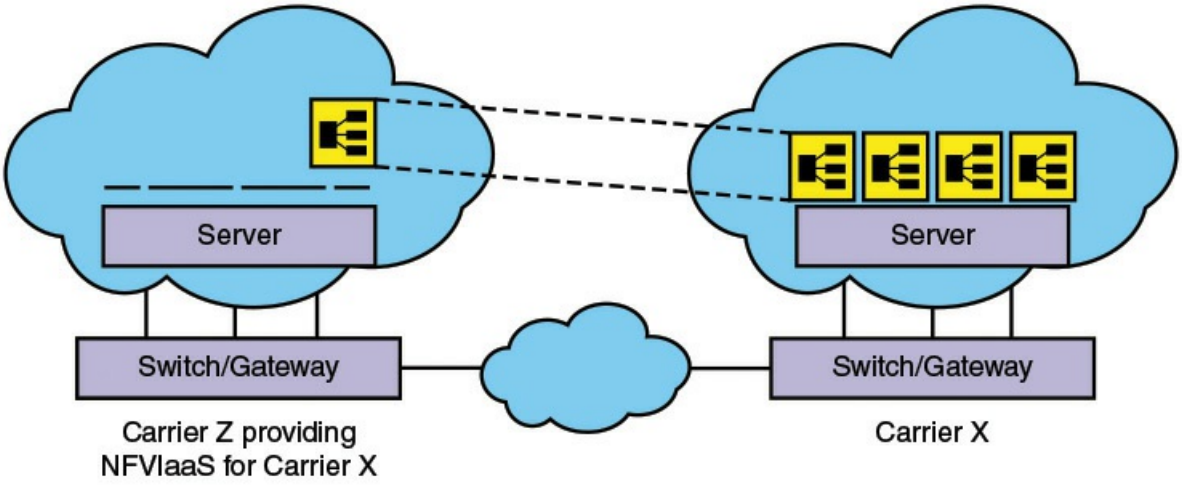


FIGURE 8.9 NFVIaaS Example

VNF as a Service

Whereas NFVIaaS is similar to the cloud model of Infrastructure as a Service (IaaS), VNFaaS corresponds to the cloud model of Software as a Service (SaaS). NFVIaaS provides the virtualization infrastructure to enable a network service provider to develop and deploy VNFs with reduced cost and time compared to implementing the NFVI and the VNFs. With VNFaaS, a provider develops VNFs that are then available off the shelf to customers. This model is well suited to virtualizing customer premises equipment such as routers and firewalls.

Virtual Network Platform as a Service

VNPaaS is similar to an NFVIaaS that includes VNFs as components of the virtual network infrastructure. The primary differences are the programmability and development tools of the VNPaaS that allow the subscriber to create and configure custom ETSI NFV-compliant VNFs to augment the catalog of VNFs offered by the service provider. This allows all the third-party and custom VNFs to be orchestrated via the VNF FG.

VNF Forwarding Graphs

VNF FG allows virtual appliances to be chained together in a flexible manner. This technique is called **service chaining**. For example, a flow may pass through a network monitoring VNF, a load-balancing VNF, and finally a firewall VNF in passing from one endpoint to another. The VNF FG use case is based on an information model that describes the VNFs and physical entities

to the appropriate management/orchestration systems used by the service provider. The model describes the characteristics of the entities including the NFV infrastructure requirements of each VNF and all the required connections among VNFs and between VNFs and the physical network included in the IaaS service. To ensure the required performance and resiliency of the end-to-end service, the information model must be able to specify the capacity, performance and resiliency requirements of each VNF in the graph. To meet SLAs, the management and orchestration system will need to monitor the nodes and linkages included in the service graph. In theory, a VNF FG can span the facilities of multiple network service providers.

Service-Oriented Use Cases

These use cases focus on the provision of services to end customers, in which the underlying infrastructure is transparent.

Virtualization of Mobile Core Network and IP Multimedia Subsystem

Mobile cellular networks have evolved to contain a variety of interconnected network function elements, typically involving a large variety of proprietary hardware appliances. NFV aims at reducing the network complexity and related operational issues by leveraging standard IT virtualization technologies to consolidate different types of network equipment onto industry standard high-volume servers, switches, and storage, located in NFVI-PoPs.

Virtualization of Mobile Base Station

The focus of this use case is radio access network (RAN) equipment in mobile networks. RAN is the part of a telecommunications system that implements a wireless technology to access the core network of the mobile network service provider. At minimum, it involves hardware on the customer premises or in the mobile device and equipment forming a base station for access to the mobile network. There is the possibility that a number of RAN functions can be virtualized as VNFs running on industry standard infrastructure.

Virtualization of the Home Environment

This use case deals with network provider equipment located as customer premises equipment (CPE) in a residential location. These CPE devices mark the operator/service provider presence at the customer premises and usually include a residential gateway (RGW) for Internet and Voice over IP (VoIP) services (for example, a modem/router for digital subscriber line [DSL] or cable), and a set-top box (STB) for media services normally supporting local storage for personal video recording (PVR) services. NFV technologies become ideal candidates to support this concentration of computation workload from formerly dispersed functions with minimal cost and improved time to market, while new services can be introduced as required on a grow-as-you-need basis. Further, the VNFs can reside on services in the network service provider’s PoP. This greatly simplifies the electronics environment of the home, reducing end user and operator capital expenditure (CapEx).

Virtualization of CDNs

Delivery of content, especially of video, is one of the major challenges of all operator networks because of the massive growing amount of traffic to be delivered to end customers of the network. The growth of video traffic is driven by the shift from broadcast to unicast delivery via IP, by the variety of devices used for video consumption and by increasing quality of video delivered via IP networks in resolution and frame rate.

Complementary to the growth of today's video traffic, the requirements on quality are also evolving: Internet actors are more and more in position to provide both live and on-demand content services to Internet end users, with similar quality constraints as for traditional TV service of network operators.

Some Internet service providers (ISPs) are deploying proprietary Content Delivery Network (CDN) cache nodes in their networks to improve delivery of video and other high-bandwidth services to their customers. Cache nodes typically run on dedicated appliances running on custom or industry standard server platforms. Both CDN cache nodes and CDN control nodes can potentially be virtualized. The benefits of CDN virtualization are similar to those gained in other NFV use cases, such as VNFaaS.

Fixed Access Network Functions Virtualization

NFV offers the potential to virtualize remote functions in the hybrid fiber/copper access network and passive optical network (PON) fiber to the home and hybrid fiber/wireless access networks. This use case has the potential for cost savings by moving complex processing closer to the network. An additional benefit is that virtualization supports multiple tenancy, in which more than one organizational entity can either be allocated, or given direct control of, a dedicated partition of a virtual access node. Finally, virtualizing broadband access nodes can enable synergies to be exploited by the co-location of wireless access nodes in a common NFV platform framework (that is, common NFVI-PoPs), thereby improving the deployment economics and reducing the overall energy consumption of the combined solution.

An indication of the relative importance of the various use cases is found in a survey of 176 network professionals from a range of industries, reported in *2015 Guide to SDN and NFV* [METZ14] and conducted in late 2014. The survey respondents were asked to indicate the two use cases that they think will gain the most traction in the market over the next two years. Table 8.5 shows their responses. The data in Table 8.5 indicates that although IT organizations have interest in a number of the ETSI-defined use cases, by a wide margin they are most interested in the NFVIaaS use case.

Use Case	Percentage of Respondents
Network Functions Virtualization Infrastructure as a Service	51 percent
Virtual Network Function as a Service (VNFaaS)	37 percent
Virtualization of Mobile Core Networks and IMS	32 percent
Virtual Network Platform as a Service (VNPaaS)	22 percent
Fixed Access Network Functions Virtualization	13 percent
Virtualization of CDNs (vCDN)	12 percent
Virtualization of Mobile base station	11 percent
Virtualization of the Home Environment	4 percent
VNF Forwarding Graphs	1 percent

TABLE 8.5 Interest in ETSI NFV Use Cases

8.5 SDN and NFV

Over the past few years, the hottest topics in networking have been SDN and NFV. Separate standards bodies are pursuing the two technologies, and a large, growing number of providers have announced or are working on products in the two fields. Each technology can be implemented and deployed separately, but there is clearly a potential for added value by the coordinated use of both technologies. It is likely that over time, SDN and NFV will tightly interoperate to provide a broad, unified software-based networking approach to abstract and programmatically control network equipment and network-based resources.

The relationship between SDN and NFV is perhaps viewed as SDN functioning as an enabler of NFV. A major challenge with NFV is to best enable the user to configure a network so that VNFs running on servers are connected to the network at the appropriate place, with the appropriate connectivity to other VNFs, and with desired QoS. With SDN, users and orchestration software can dynamically configure the network and the distribution and connectivity of VNFs. Without SDN, NFV requires much more manual intervention, especially when resources beyond the scope of NFVI are part of the environment.

The *Kemp Technologies Blog* [MCMU14] gives the example of load balancing where load balancer services are implemented as VNF entities. If demand for load-balancing capacity increases, a network orchestration layer can rapidly spin up new load-balancing instances and also adjust the network switching infrastructure to accommodate the changed traffic patterns. In turn, the load-balancing VNF entity can interact with the SDN controller to assess network performance and capacity and use this additional information to balance traffic better, or even to request provisioning of additional VNF resources.

Some of the ways that ETSI believes that NFV and SDN complement each other include the following:

- The SDN controller fits well into the broader concept of a network controller in an NFVI network domain.
- SDN can play a significant role in the orchestration of the NFVI resources, both physical

and virtual, enabling functionality such as provisioning, configuration of network connectivity, bandwidth allocation, automation of operations, monitoring, security, and policy control.

- SDN can provide the network virtualization required to support multitenant NFVIs.
- Forwarding graphs can be implemented using the SDN controller to provide automated provisioning of service chains, while ensuring strong and consistent implementation of security and other policies.
- The SDN controller can be run as a VNF, possibly as part of a service chain including other VNFs. For example, applications and services originally developed to run on the SDN controller could also be implemented as separate VNFs.

Figure 8.10, from the ETSI VNF Architecture document, indicates the potential relationship between SDN and NFV. The arrows can be described as follows:

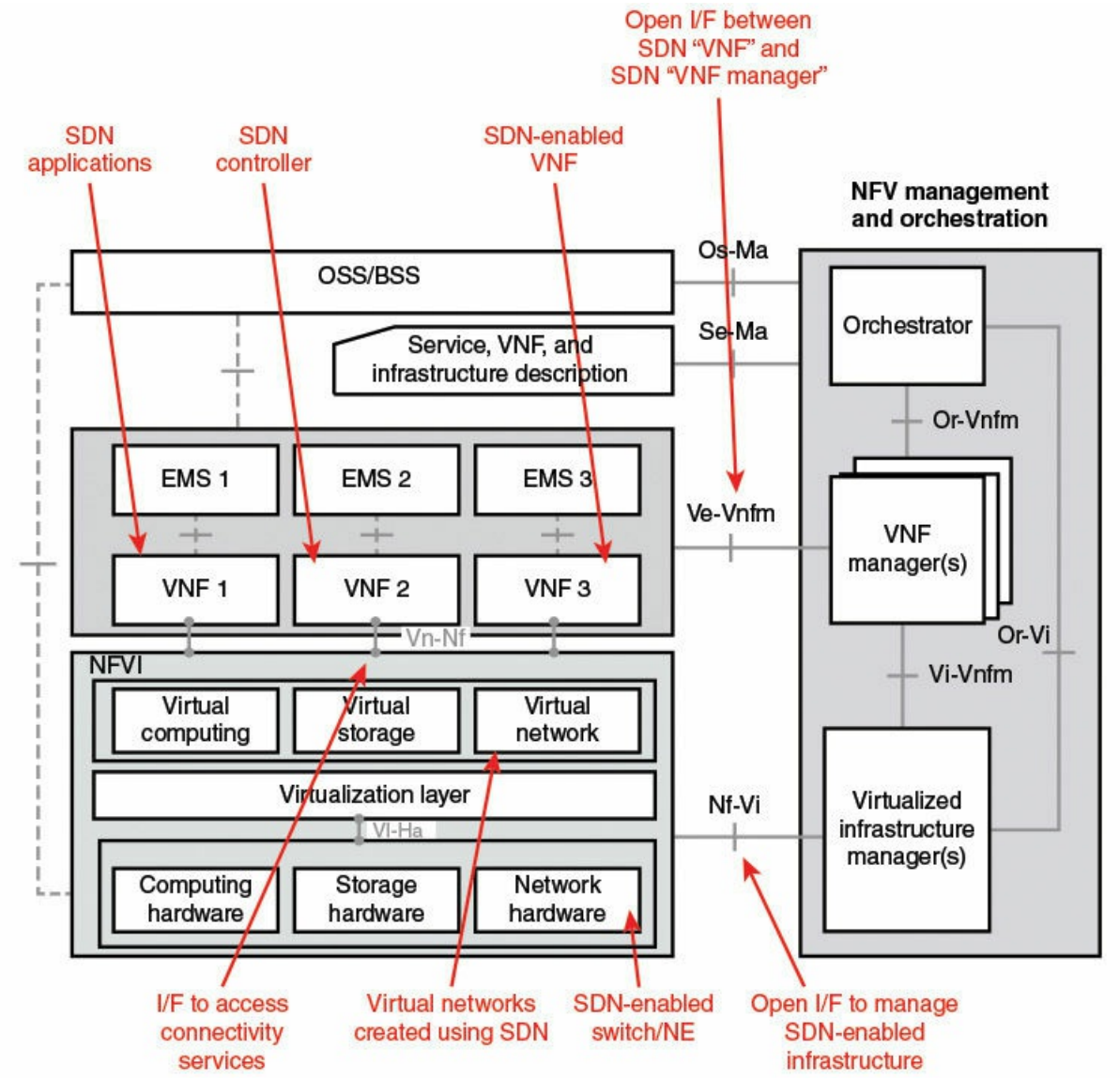


FIGURE 8.10 Mapping of SDN Components with NFV Architecture

- SDN enabled switch/NEs include physical switches, hypervisor virtual switches, and embedded switches on the NICs.
- Virtual networks created using an infrastructure network SDN controller provide connectivity services between VNFC instances.
- SDN controller can be virtualized, running as a VNF with its EM and VNF manager. Note that there may be SDN controllers for the physical infrastructure, the virtual infrastructure, and the virtual and physical network functions. As such, some of these SDN controllers may reside in the NFVI or management and orchestration (MANO) functional blocks (not shown in figure).
- SDN enabled VNF includes any VNF that may be under the control of an SDN controller (for example, virtual router, virtual firewall).
- SDN applications, for example service chaining applications, can be VNF themselves.
- Nf-Vi interface allows management of the SDN enabled infrastructure.
- Ve-Vnfm interface is used between the SDN VNF (SDN controller VNF, SDN network functions VNF, SDN applications VNF) and their respective VNF Manager for lifecycle management.
- Vn-Nf allows SDN VNFs to access connectivity services between VNFC interfaces.

8.6 Key Terms

After completing this chapter, you should be able to define the following terms.

compute domain

compute node

container

container interface

content delivery network (CDN)

deep packet inspection

element management

element management system (EMS)

forwarding graph (FG)

functional block interface

gateway node

hypervisor

hypervisor domain

infrastructure network domain (IND)

L3 virtual network

layered virtual network

network interface card

network node

NFV management and orchestration (MANO)

NFV infrastructure (NFVI)

NFV orchestrator

NFVI domain

operations support system

reference points

scale out

scale up

service chaining

storage node

virtual network

virtual overlay

virtual partition

virtualized infrastructure manager

virtualization

virtualization container

virtualized network function (VNF)

VNF manager

vswitch

8.7 References

KHAN15: Khan, F. *A Beginner's Guide to NFV Management & Orchestration (MANO)*. Telecom Lighthouse. April 9, 2015. http://www.telecomlighthouse.com.

MCMU14: McMullin, M. "SDN is from Mars, NFV is from Venus." *Kemp Technologies Blog*, November 20, 2014. http://kemptechnologies.com/blog/sdn-mars-nfv-venus.

METZ14a: Metzler, J. *The 2015 Guide to SDN and NFV.* Webtorials, December 2014.

ONF14: Open Networking Foundation. *OpenFlow-Enabled SDN and Network Functions Virtualization.* ONF white paper, February 17, 2014.

Chapter 9. Network Virtualization

> In recent years a strong and significant partnership has grown up between computers and communication systems. On the one hand, computers are being used to effect far-reaching improvements in communication systems, while on the other, communication systems are being used to increase and extend the utility of computers.
>
> —*What Can Be Automated?*
> The Computer Science and Engineering Research Study, National Science Foundation, 1980

Chapter Objectives: **After studying this chapter, you should be able to**

- Understand the concept of a virtual LAN and the three ways of defining a VLAN.
- Present an overview of the IEEE 802.1Q standards.
- Explain how OpenFlow supports VLANs.
- Understand the concept of a virtual private network.
- Define network virtualization.
- Understand the operation of OpenDaylight's Virtual Tenant Network.
- Summarize the concepts of software-defined infrastructure.
- Discuss software-defined storage.

Mechanisms for defining virtual networks have been in use for many years. Virtual networks have two important benefits:

- They enable the user to construct and manage networks independent of the underlying physical network and with assurance of isolation from other virtual networks using the same physical network.
- They enable network providers to efficiently use network resources to support a wide range of user requirements.

The chapter begins with a discussion of two well-established and widely used virtual network techniques: virtual LANs (VLANs) and virtual private networks (VPNs). The chapter then introduces the more general and broader concept of network virtualization. After exploring a simple example, you will learn about the network virtualization architecture and the benefits of this approach. The chapter also looks at OpenDaylight's Virtual Tenant Network, which is a VLAN-based capability, but which exhibits many of the features of network virtualization. Finally, the chapter introduces the concept of software-defined infrastructure, which encompasses many of the concepts of software-defined network (SDN), network functions virtualization (NFV), and network virtualization.

9.1 Virtual LANs

Figure 9.1 shows a relatively common type of hierarchical LAN configuration. In this example, the devices on the LAN are organized into four segments, each served by a LAN switch. The **LAN switch** is a store-and-forward packet-forwarding device used to interconnect a number of end systems to form a LAN segment. The switch can forward a **media access control (MAC) frame** from a source-attached device to a destination-attached device. It can also broadcast a frame from a source-attached device to all other attached devices. Multiples switches can be interconnected so that multiple LAN segments form a larger LAN. A LAN switch can also connect to a transmission link or a router or other network device to provide connectivity to the Internet or other WANs.

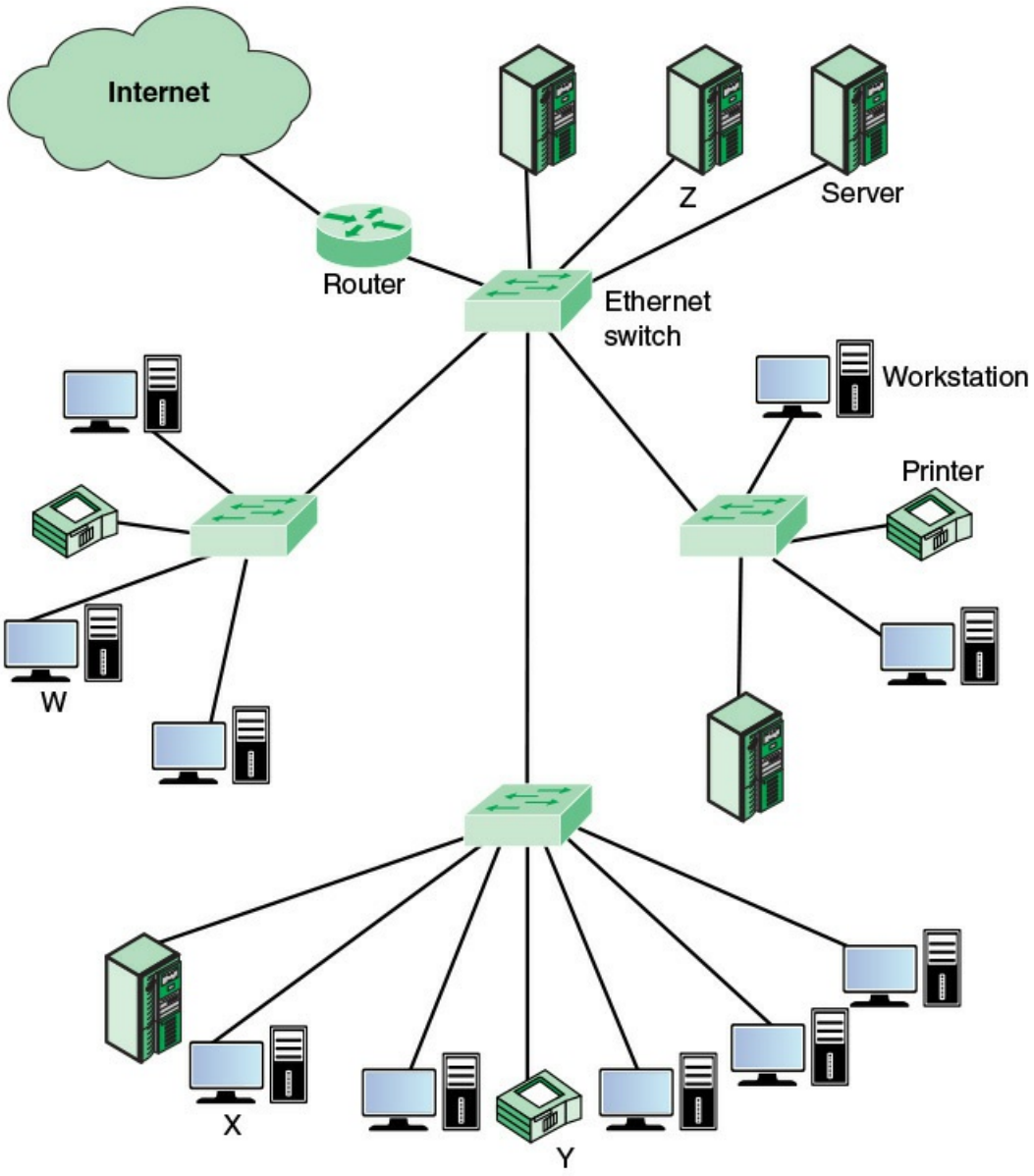


FIGURE 9.1 A LAN Configuration

Traditionally, a LAN switch operated exclusively at the MAC level. Contemporary LAN switches generally provide greater functionality, including multilayer awareness (Layers 3, 4,

application), quality of service (QoS) support, and trunking for wide-area networking.

The three lower groups in Figure 9.1 might correspond to different departments, which are physically separated, and the upper group could correspond to a centralized server farm that is used by all the departments.

Consider the transmission of a single MAC frame from workstation X. Suppose the destination MAC address in the frame is workstation Y. This frame is transmitted from X to the local switch, which then directs the frame along the link to Y. If X transmits a frame addressed to Z or W, its local switch forwards the MAC frame through the appropriate switches to the intended destination. All these are examples of **unicast addressing**, in which the destination address in the MAC frame designates a unique destination. A MAC frame may also contain a **broadcast address**, in which case the destination MAC address indicates that all devices on the LAN should receive a copy of the frame. Thus, if X transmits a frame with a broadcast destination address, all the devices on all the switches in Figure 9.1 receive a copy of the frame. The total collection of devices that receive broadcast frames from each other is referred to as a **broadcast domain**.

In many situations, a broadcast frame is used for a purpose, such as network management or the transmission of some type of alert, with a relatively local significance. Thus, in Figure 9.1, if a broadcast frame has information that is useful only to a particular department, transmission capacity is wasted on the other portions of the LAN and on the other switches.

One simple approach to improving efficiency is to physically partition the LAN into separate broadcast domains, as shown in Figure 9.2. We now have four separate LANs connected by a router. In this case, a broadcast frame from X is transmitted only to the other devices directly connected to the same switch as X. An IP packet from X intended for Z is handled as follows. The IP layer at X determines that the next hop to the destination is via router V. This information is handed down to X's MAC layer, which prepares a MAC frame with a destination MAC address of router V. When V receives the frame, it strips off the MAC header, determines the destination, and encapsulates the IP packet in a MAC frame with a destination MAC address of Z. This frame is then sent to the appropriate Ethernet switch for delivery.

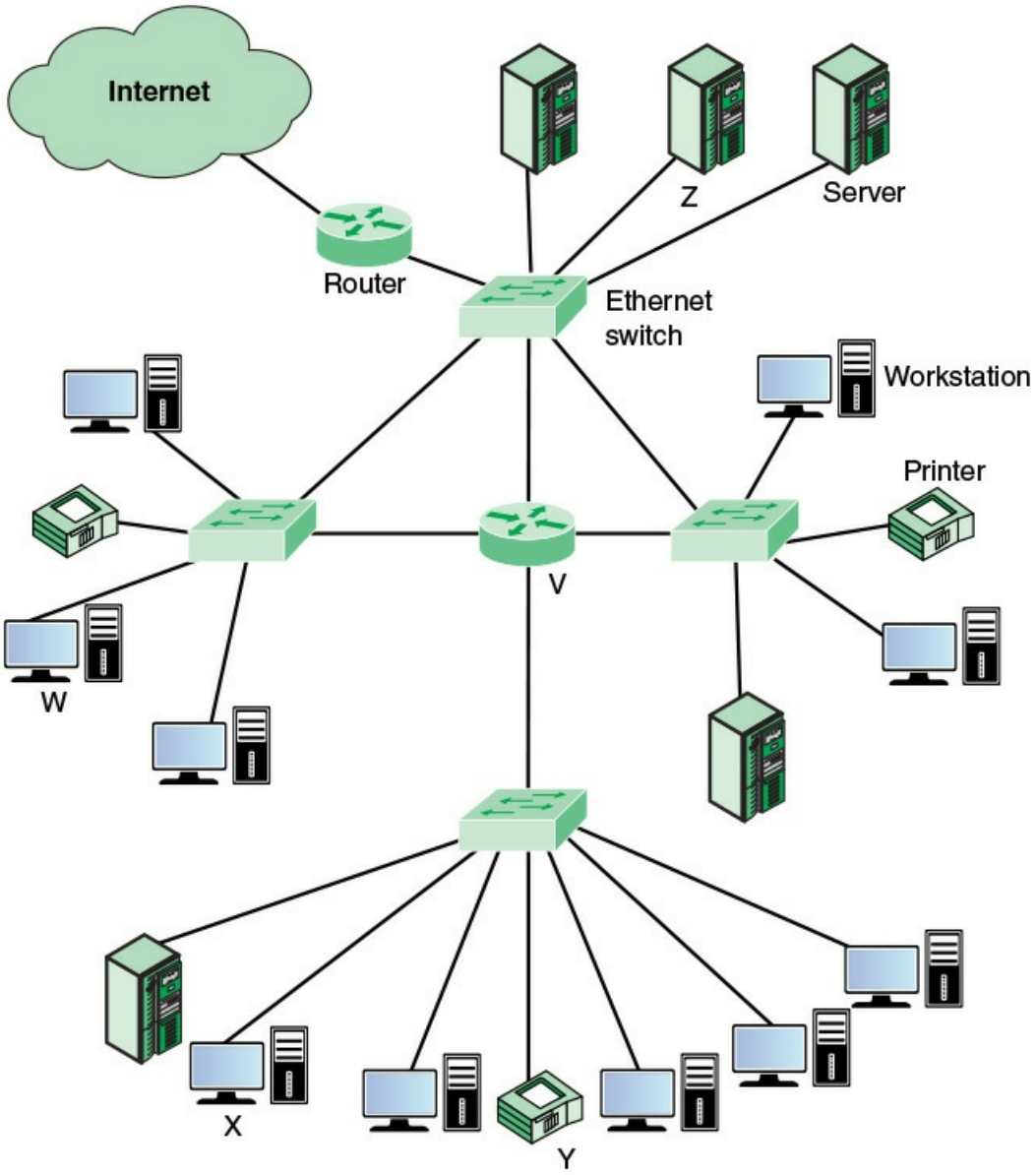


FIGURE 9.2 A Partitioned LAN

The drawback to this approach is that the traffic pattern may not correspond to the physical distribution of devices. For example, some departmental workstations may generate a lot of traffic with one of the central servers. Further, as the networks expand, more routers are needed to separate users into broadcast domains and provide connectivity among broadcast domains. Routers introduce more latency than switches because the router must process more of the packet to determine destinations and route the data to the appropriate end node.

The Use of Virtual LANs

A more effective alternative is the creation of VLANs. In essence, a **virtual local-area network (VLAN)** is a logical subgroup within a LAN that is created by software rather than by physically

moving and separating devices. It combines user stations and network devices into a single broadcast domain regardless of the physical LAN segment they are attached to and allows traffic to flow more efficiently within populations of mutual interest. The VLAN logic is implemented in LAN switches and functions at the MAC layer. Because the objective is to isolate traffic within the VLAN, a router is required to link from one VLAN to another. Routers can be implemented as separate devices, so that traffic from one VLAN to another is directed to a router, or the router logic can be implemented as part of the LAN switch, as shown in Figure 9.3.

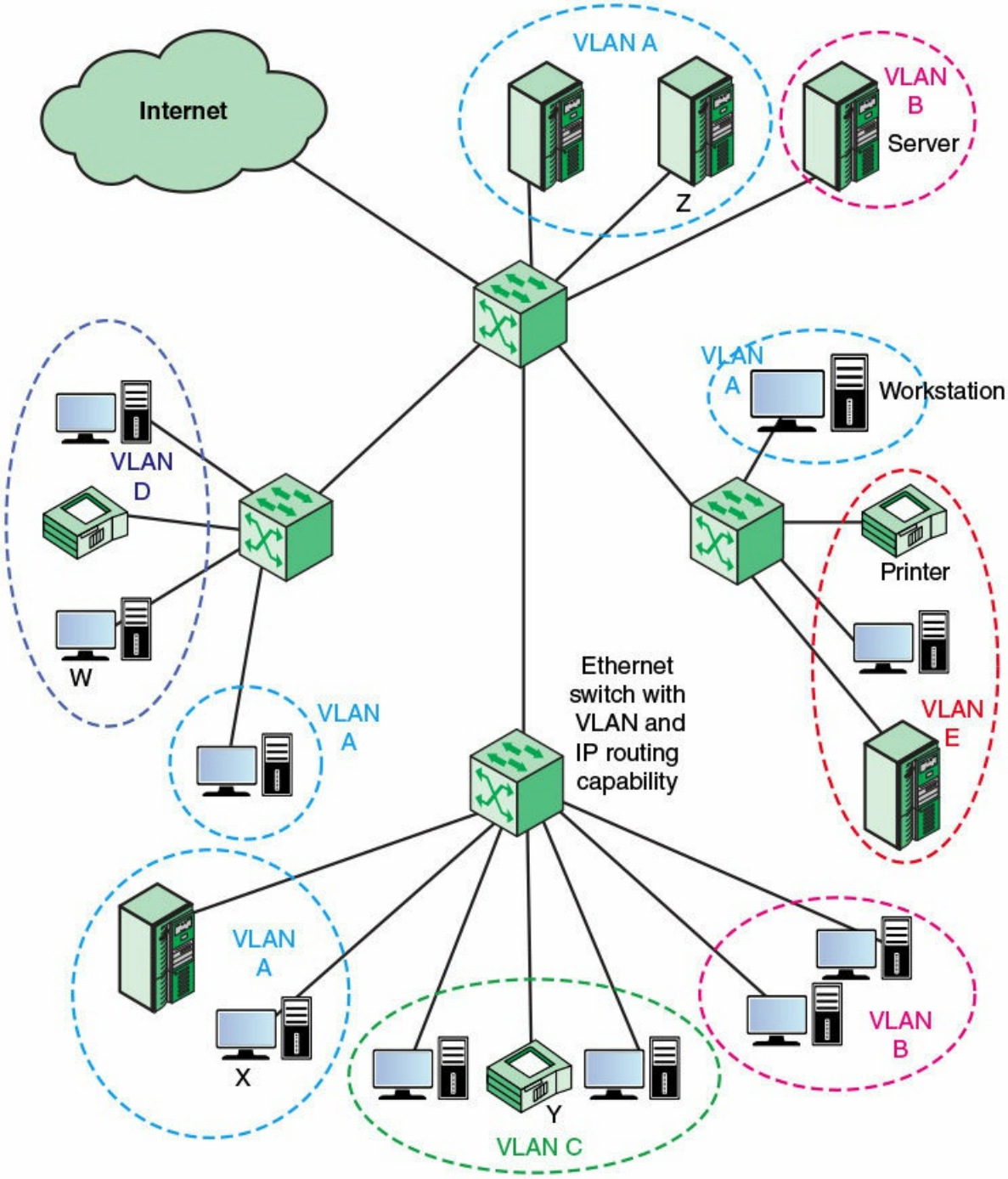


FIGURE 9.3 A VLAN Configuration

VLANs enable any organization to be physically dispersed throughout the company while maintaining its group identity. For example, accounting personnel can be located on the shop floor, in the research and development center, in the cash disbursement office, and in the corporate offices, while all members reside on the same virtual network, sharing traffic only with each other.

Figure 9.3 shows five defined VLANs. A transmission from workstation X to server Z is within the same VLAN, so it is efficiently switched at the MAC level. A broadcast MAC frame from X is transmitted to all devices in all portions of the same VLAN. But a transmission from X to printer Y goes from one VLAN to another. Accordingly, router logic at the IP level is required to move the IP packet from X to Y. Figure 9.3 shows that logic integrated into the switch, so that the switch determines whether the incoming MAC frame is destined for another device on the same VLAN. If not, the switch routes the enclosed IP packet at the IP level.

Defining VLANs

A VLAN is a broadcast domain consisting of a group of end stations, perhaps on multiple physical LAN segments, that are not constrained by their physical location and can communicate as if they were on a common LAN. Some means is therefore needed for defining VLAN membership. A number of different approaches have been used for defining membership, including the following:

- **Membership by port group**: Each switch in the LAN configuration contains two types of ports: a trunk port, which connects two switches; and an end port, which connects the switch to an end system. A VLAN can be defined by assigning each end port to a specific VLAN. This approach has the advantage that it is relatively easy to configure. The principle disadvantage is that the network manager must reconfigure VLAN membership when an end system moves from one port to another.
- **Membership by MAC address**: Because MAC layer addresses are hardwired into the workstation's network interface card (NIC), VLANs based on MAC addresses enable network managers to move a workstation to a different physical location on the network and have that workstation automatically retain its VLAN membership. The main problem with this method is that VLAN membership must be assigned initially. In networks with thousands of users, this is no easy task. Also, in environments where notebook PCs are used, the MAC address is associated with the docking station and not with the notebook PC. Consequently, when a notebook PC is moved to a different docking station, its VLAN membership must be reconfigured.
- **Membership based on protocol information**: VLAN membership can be assigned based on IP address, transport protocol information, or even higher-layer protocol information. This is a quite flexible approach, but it does require switches to examine portions of the MAC frame above the MAC layer, which may have a performance impact.

Communicating VLAN Membership

Switches must have a way of understanding VLAN membership (that is, which stations belong to which VLAN) when network traffic arrives from other switches; otherwise, VLANs would be limited to a single switch. One possibility is to configure the information manually or with some type of network management signaling protocol, so that switches can associate incoming frames with the appropriate VLAN.

A more common approach is frame tagging, in which a header is typically inserted into each frame on interswitch trunks to uniquely identify to which VLAN a particular MAC-layer frame belongs. The **IEEE 802** committee has developed a standard for frame tagging, **IEEE 802.1Q**, which we examine next.

IEEE 802.1Q VLAN Standard

The IEEE 802.1Q standard, last updated in 2014, defines the operation of VLAN bridges and switches that permits the definition, operation, and administration of VLAN topologies within a bridged/switched LAN infrastructure. In this section, we concentrate on the application of this standard to **802.3** LANs.

Recall that a VLAN is an administratively configured broadcast domain, consisting of a subset of end stations attached to a LAN. A VLAN is not limited to one switch but can span multiple interconnected switches. In that case, traffic between switches must indicate VLAN membership. This is accomplished in 802.1Q by inserting a tag with a VLAN identifier (VID) with a value in the range from 1 to 4094. Each VLAN in a LAN configuration is assigned a globally unique VID. By assigning the same VID to end systems on many switches, one or more VLAN broadcast domains can be extended across a large network.

Figure 9.4 shows the position and content of the 802.1 tag, referred to as Tag Control Information (TCI). The presence of the two-octet TCI field is indicated by inserting a Length/Type field in the 802.3 MAC frame with a value of 8100 hex. The TCI consists of three subfields, as described in the list that follows.

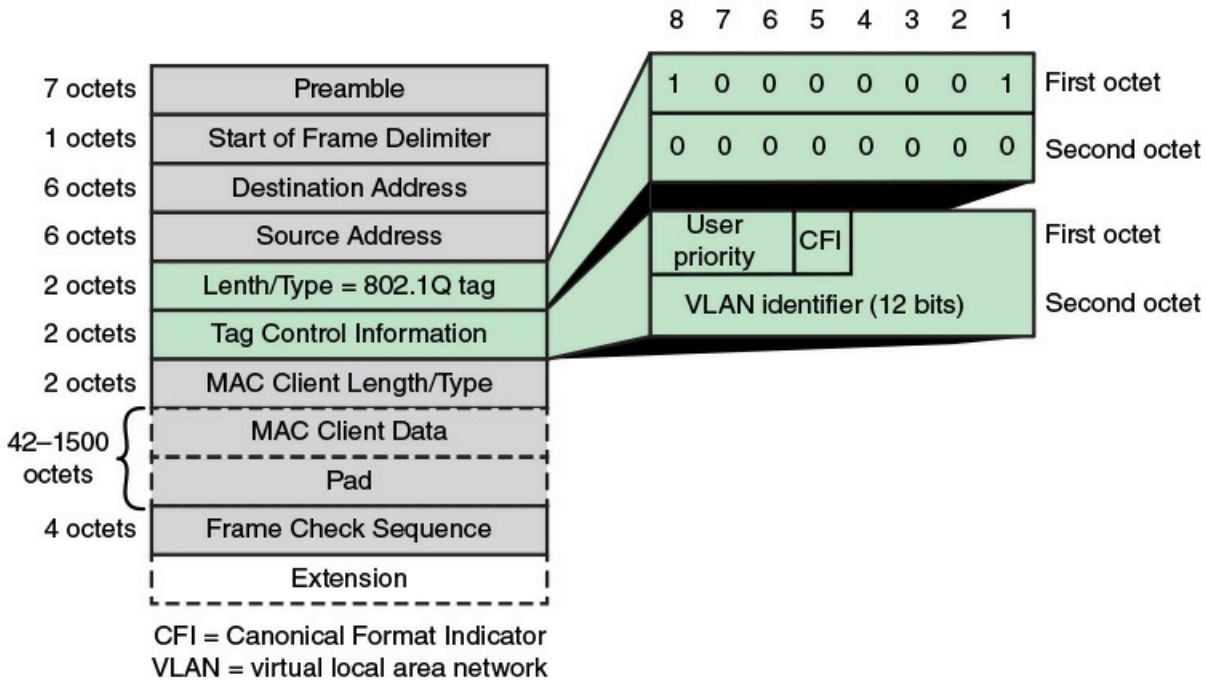


FIGURE 9.4 Tagged IEEE 802.3 MAC Frame Format

- **User priority (3 bits):** The priority level for this frame.
- **Canonical format indicator (1 bit):** Is always set to 0 for Ethernet switches. CFI is used for compatibility between Ethernet type networks and Token Ring type networks. If a frame received at an Ethernet port has a CFI set to 1, that frame should not be forwarded as it is to an untagged port.
- **VLAN identifier (12 bits):** The identification of the VLAN. Of the 4096 possible VIDs, a VID of 0 is used to identify that the TCI contains only a priority value, and 4095 (0xFFF) is reserved, so the maximum possible number of VLAN configurations is 4094.

Figure 9.5 illustrates a LAN configuration that includes three switches that implement 802.1Q and one "legacy" switch that does not. In this case, all the end systems of the legacy device must belong to the same VLAN. The MAC frames that traverse trunks between VLAN-aware switches include the 802.1Q TCI tag. This tag is stripped off before a frame is forwarded to a legacy switch. For end systems connected to a VLAN-aware switch, the MAC frame may or may not include the TCI tag, depending on the implementation. The important point is that the TCI tag is used between VLAN-aware switches so that appropriate routing and frame handling can be performed.

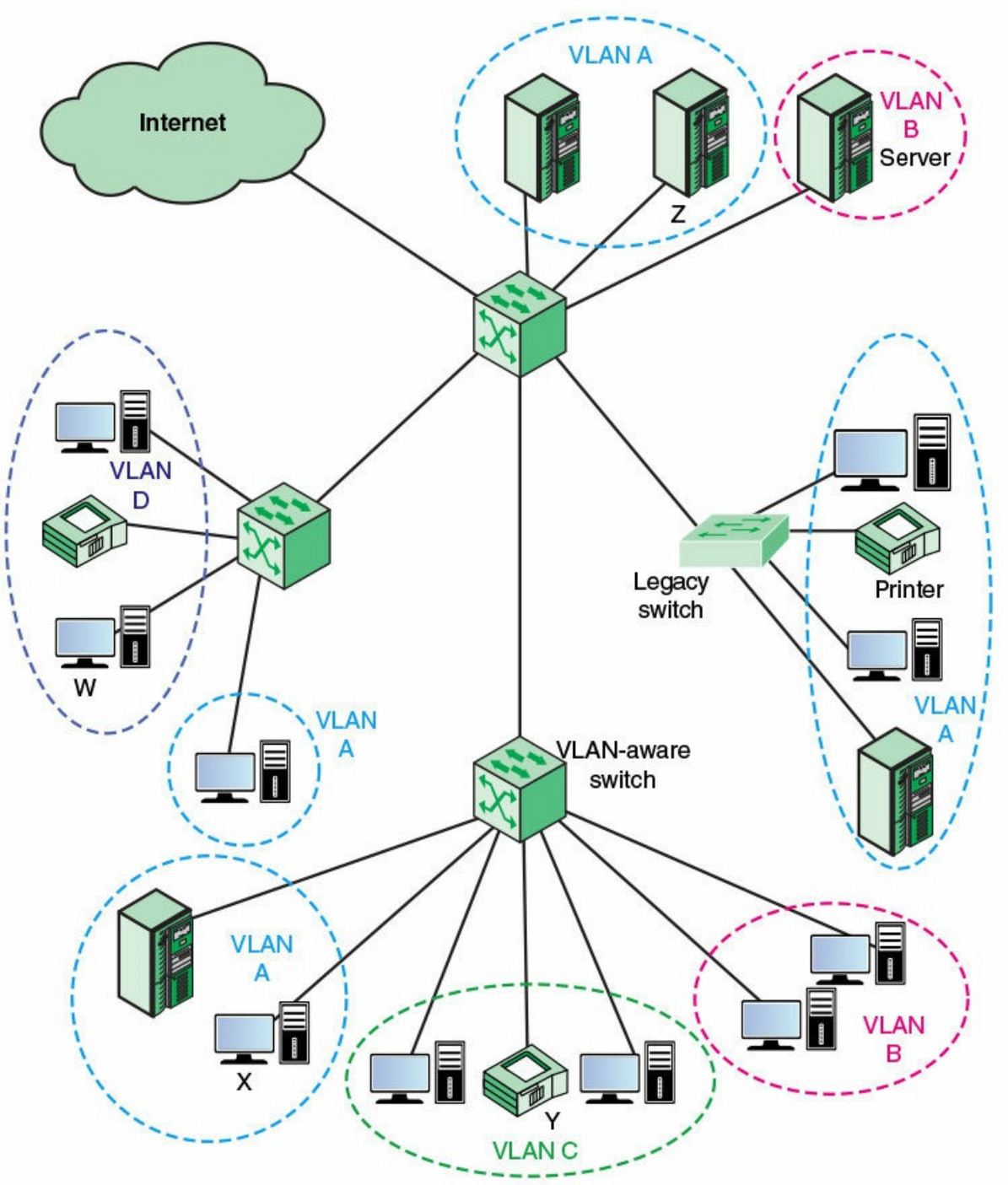


FIGURE 9.5 A VLAN Configuration with 802.1Q and Legacy Switches

Nested VLANs

The original 802.1Q specification allowed for a single VLAN tag field to be inserted into an Ethernet MAC frame. More recent versions of the standard allow for the insertion of two VLAN tag fields, allowing the definition of multiple sub-VLANs within a single VLAN. This additional

flexibility might be useful in some complex configurations.

For example, a single VLAN level suffices for an Ethernet configuration entirely on a single premises. However, it is not uncommon for an enterprise to make use of a network service provider to interconnect multiple LAN locations, and to use metropolitan area Ethernet links to connect to the provider. Multiple customers of the service provider may wish to use the 802.1Q tagging facility across the service provider network (SPN).

One possible approach is for the customer's VLANs to be visible to the service provider. In that case, the service provider could support a total of only 4094 VLANs for all its customers. Instead, the service provider inserts a second VLAN tag into Ethernet frames. For example, consider two customers with multiple sites, both of which use the same SPN (see part a of Figure 9.6). Customer A has configured VLANs 1 to 100 at their sites, and similarly Customer B has configured VLANs 1 to 50 at their sites. The tagged data frames belonging to the customers must be kept separate while they traverse the service provider's network. The customer's data frame can be identified and kept separate by associating another VLAN for that customer's traffic. This results in the tagged customer data frame being tagged again with a VLAN tag, when it traverses the SPN (see part b of Figure 9.6). The additional tag is removed at the edge of the SPN when the data enters the customer's network again. Packed VLAN tagging is known as VLAN stacking or as Q-in-Q.

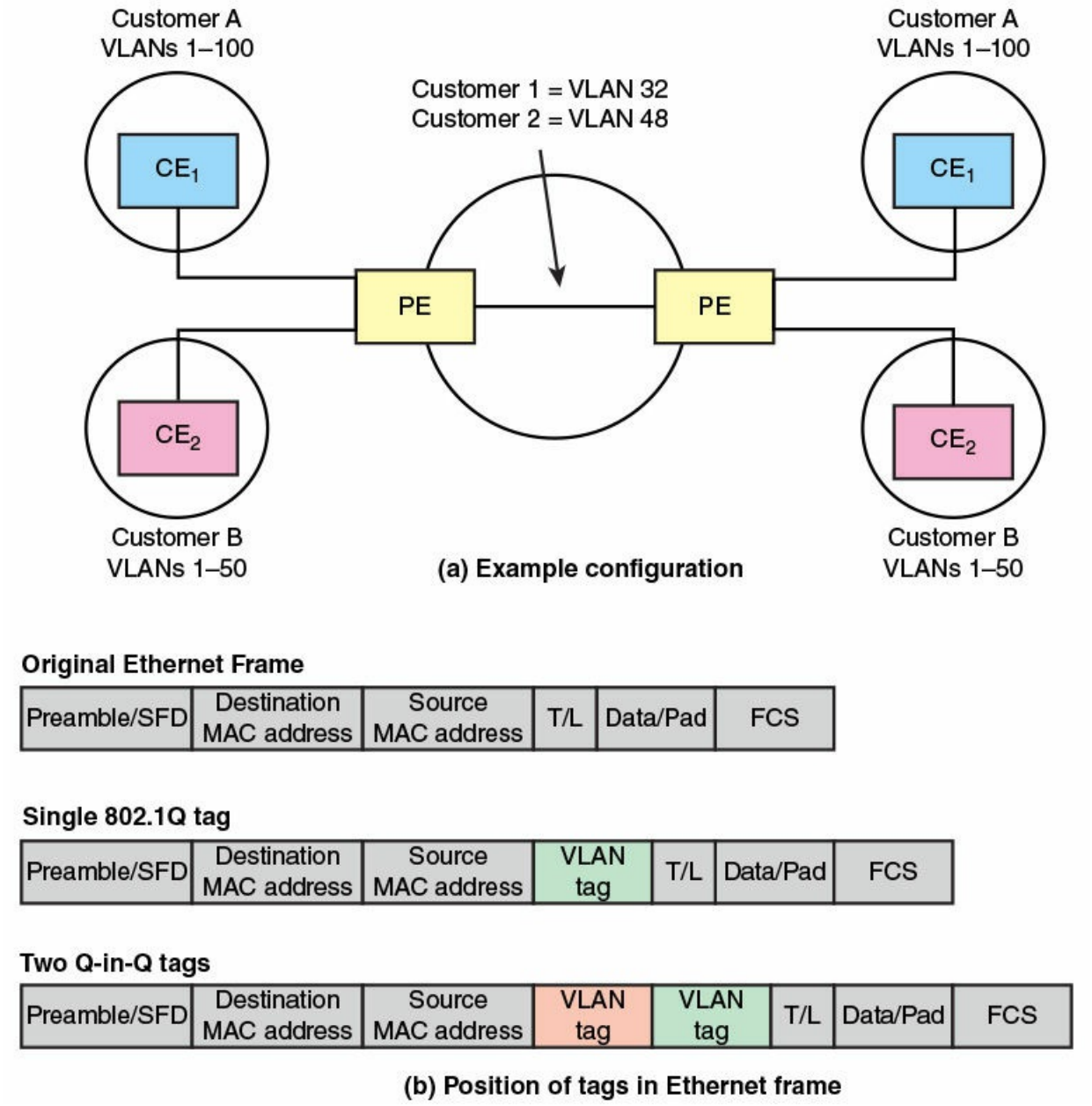


FIGURE 9.6 Use of Stacked VLAN Tags

9.2 OpenFlow VLAN Support

A traditional 802.1Q VLAN requires that the network switches have a complete knowledge of the VLAN mapping. This knowledge may be manually configured or acquired automatically. Another drawback is related to the choice of one of three ways of defining group membership (port group, MAC address, protocol information). The network administrator must evaluate the trade-offs according to the type of network they wish to deploy and choose one of the possible approaches. It would be difficult to deploy a more flexible definition of a VLAN or even a custom definition (for example, use a combination of IP addresses and ports) with traditional

networking devices. Reconfiguring VLANs is also a daunting task for administrators: Multiple switches and routers have to be reconfigured whenever VMs are relocated.

SDN, and in particular OpenFlow, allows for much more flexible management and control of VLANs. It should be clear how OpenFlow can set up flow table entries for forwarding based on one or both VLAN tags, and how tags can be added, modified, and removed.

9.3 Virtual Private Networks

In today's distributed computing environment, the **virtual private network (VPN)** offers an attractive solution to network managers. A VPN is a private network that is configured within a public network (a carrier's network or the Internet) to take advantage of the economies of scale and management facilities of large networks. VPNs are widely used by enterprises to create WANs that span large geographic areas, to provide site-to-site connections to branch offices, and to allow mobile users to dial up their company LANs. From the point of view of the provider, the public network facility is shared by many customers, with the traffic of each customer segregated from other traffic. Traffic designated as VPN traffic can only go from a VPN source to a destination in the same VPN. It is often the case that encryption and authentication facilities are provided for the VPN.

A typical scenario for an enterprise that uses VPNs is the following. At each corporate site, one or more LANs link workstations, servers, and databases. The LANs are under the control of the enterprise and can be configured and tuned for cost-effective performance. VPNs over the Internet or some other public network can be used to interconnect sites, providing a cost savings over the use of a private network and offloading the WAN management task to the public network provider. That same public network provides an access path for telecommuters and other mobile employees to log on to corporate systems from remote sites.

The subject of VPNs is extraordinarily complex and this section can only provide a concise overview of the two most common technologies for creating VPNs: **IP security (IPsec)** and Multiprotocol Label Switching (MPLS).

IPsec VPNs

Use of a shared network, such as the Internet or a public carrier network, as part of an enterprise network architecture exposes corporate traffic to eavesdropping and provides an entry point for unauthorized users. To counter this problem, IPsec can be used to construct VPNs. The principal feature of IPsec that enables it to support these varied applications is that it can encrypt/authenticate traffic at the IP level. Therefore, all distributed applications, including remote logon, client/server, e-mail, file transfer, web access, and so on, can be secured.

Part a of Figure 9.7 shows the packet format for an IPsec option known as tunnel mode. Tunnel mode makes use of the combined authentication/encryption function IPsec called Encapsulating Security Payload (ESP), and a key exchange function. For VPNs, both authentication and encryption are generally desired, because it is important both to (1) ensure that unauthorized users do not penetrate the VPN, and (2) ensure that eavesdroppers on the Internet cannot read messages sent over the VPN.

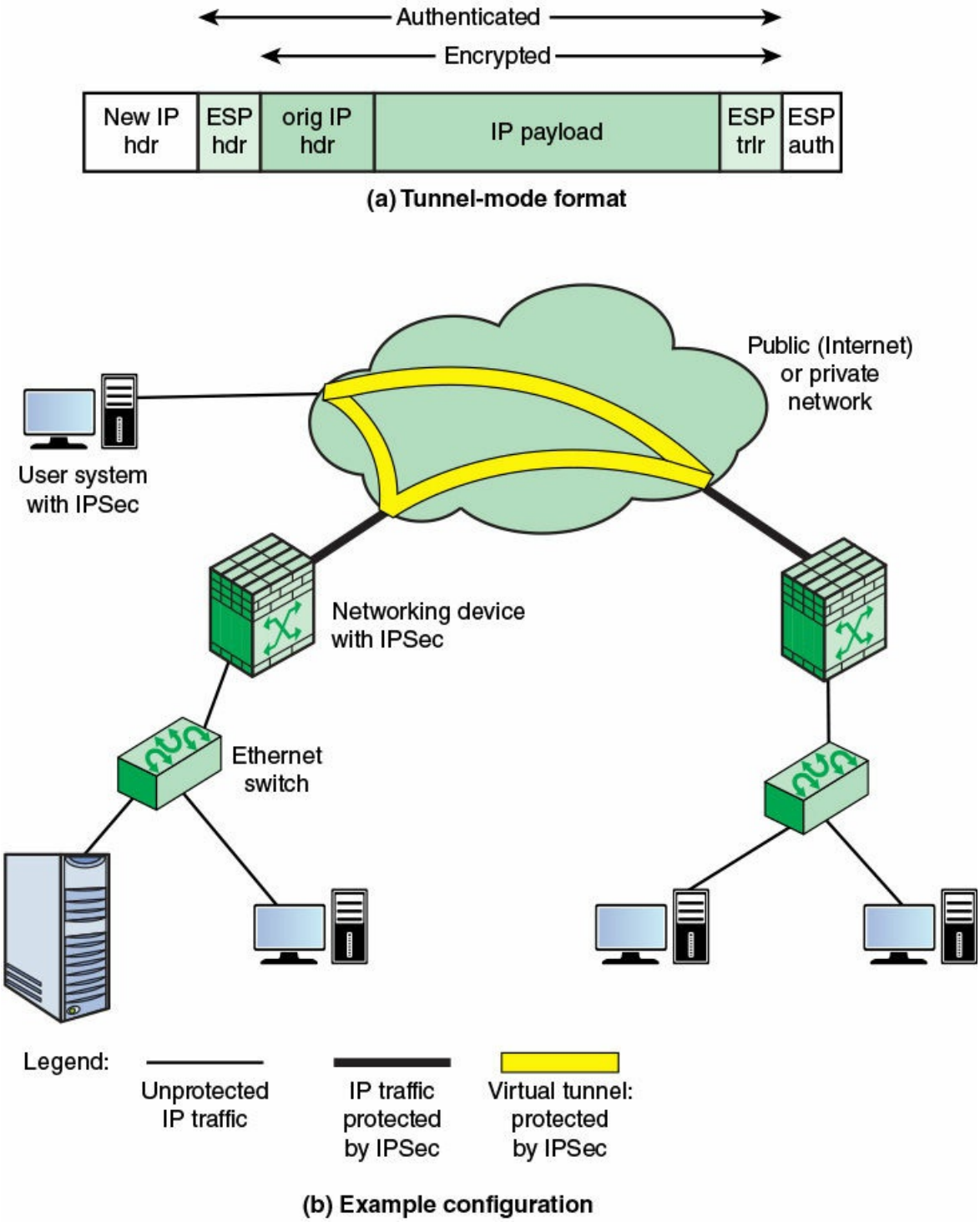


FIGURE 9.7 An IPsec VPN Scenario

Part b of Figure 9.7 is a typical scenario of IPsec usage. An organization maintains LANs at dispersed locations. Nonsecure IP traffic is conducted on each LAN. For traffic offsite, through some sort of private or public WAN, IPsec protocols are used. These protocols operate in networking devices, such as a router or firewall, that connect each LAN to the outside world.

The IPsec networking device will typically encrypt all traffic going into the WAN, and decrypt and authenticate traffic coming from the WAN; these operations are transparent to workstations and servers on the LAN. Secure transmission is also possible with individual users who connect to the WAN. Such user workstations must implement the IPsec protocols to provide security.

Using IPsec to construct a VPN has the following benefits:

- When IPsec is implemented in a firewall or router, it provides strong security that can be applied to all traffic crossing the perimeter. Traffic within a company or workgroup does not incur the overhead of security-related processing.
- IPsec in a firewall is resistant to bypass if all traffic from the outside must use IP and the firewall is the only means of entrance from the Internet into the organization.
- IPsec is below the transport layer (TCP, UDP) and so is transparent to applications. There is no need to change software on a user or server system when IPsec is implemented in the firewall or router. Even if IPsec is implemented in end systems, upper-layer software, including applications, is not affected.
- IPsec can be transparent to end users. There is no need to train users on security mechanisms, issue keying material on a per-user basis, or revoke keying material when users leave the organization.
- IPsec can provide security for individual users if needed. This is useful for offsite workers and for setting up a secure virtual subnetwork within an organization for sensitive applications.

MPLS VPNs

An alternative, and popular, means of constructing VPNs is using MPLS. This discussion begins with a brief summary of MPLS, followed by a an overview of two of the most common approaches to VPN implementation using MPLS: the Layer 2 VPN (L2VPN) and the Layer 3 VPN (L3VPN).

MPLS Overview

Multiprotocol Label Switching (MPLS) is a set of Internet Engineering Task Force (IETF) specifications for including routing and traffic engineering information in packets. MPLS comprises a number of interrelated protocols, which can be referred to as the MPLS protocol suite. It can be used in IP networks but also in other types of packet-switching networks. MPLS is used to ensure that all packets in a particular flow take the same route over a backbone. Deployed by many telecommunication companies and service providers, MPLS delivers the QoS required to support real-time voice and video as well as service level agreements (SLAs) that guarantee bandwidth.

In essence, MPLS is an efficient technique for forwarding and routing packets. MPLS was designed with IP networks in mind, but the technology can be used without IP to construct a network with any link-level protocol. In an ordinary packet-switching network, packet switches must examine various fields within the packet header to determine destination, route, QoS, and

any traffic management functions (such as discard or delay) that may be supported. Similarly, in an IP-based network, routers examine a number of fields in the IP header to determine these functions. In an MPLS network, a fixed-length label encapsulates an IP packet or a data link frame. The MPLS label contains all the information needed by an MPLS-enabled router to perform routing, delivery, QoS, and traffic management functions. Unlike IP, MPLS is connection oriented.

An MPLS network or internet consists of a set of nodes, called **label-switching routers (LSRs)** capable of switching and routing packets on the basis of a label appended to each packet. Labels define a flow of packets between two endpoints or, in the case of multicast, between a source endpoint and a multicast group of destination endpoints. For each distinct flow, called a **forwarding equivalence class (FEC)**, a specific path through the network of LSRs is defined, called a **label-switched path (LSP)**. In essence, an FEC represents a group of packets that share the same transport requirements. All packets in an FEC receive the same treatment en route to the destination. These packets follow the same path and receive the same QoS treatment at each hop. In contrast to forwarding in ordinary IP networks, the assignment of a particular packet to a particular FEC is done just once, when the packet enters the network of MPLS routers.

The list that follows, based on RFC 4026, *Provider Provisioned Virtual Private Network Terminology*, defines key VPN terms used in the following discussion:

- **Attachment circuit (AC):** In a Layer 2 VPN, the CE is attached to PE via an AC. The AC may be a physical or logical link.
- **Customer edge (CE):** A device or set of devices on the customer premises that attaches to a provider-provisioned VPN.
- **Layer 2 VPN (L2VPN):** An L2VPN interconnects sets of hosts and routers based on Layer 2 addresses.
- **Layer 3 VPN (L3VPN):** An L3VPN interconnects sets of hosts and routers based on Layer 3 addresses.
- **Packet-switched network (PSN):** A network through which the tunnels supporting the VPN services are set up.
- **Provider edge (PE):** A device or set of devices at the edge of the provider network with the functionality that is needed to interface with the customer.
- **Tunnel:** Connectivity through a PSN that is used to send traffic across the network from one PE to another. The tunnel provides a means to transport packets from one PE to another. Separation of one customer’s traffic from another customer’s traffic is done based on tunnel multiplexers
- **Tunnel multiplexer:** An entity that is sent with the packets traversing the tunnel to make it possible to decide which instance of a service a packet belongs to and from which sender it was received. In an MPLS network, the tunnel multiplexor is formatted as an MPLS label.
- **Virtual channel (VC):** A VC is transported within a tunnel and identified by its tunnel multiplexer. In an MPLS-enabled IP network, a VC label is an MPLS label used to identify traffic within a tunnel that belongs to a particular VPN; that is, the VC label is the tunnel multiplexer in networks that use MPLS labels.

- **Virtual private network (VPN):** A generic term that covers the use of public or private networks to create groups of users that are separated from other network users and that may communicate among them as if they were on a private network.

Layer 2 MPLS VPN

With a Layer 2 MPLS VPN, there is mutual transparency between the customer network and the provider network. In effect, the customer requests a mesh of unicast LSPs among customer switches that attach to the provider network. Each LSP is viewed as a Layer 2 circuit by the customer. In an L2VPN, the provider's equipment forwards customer data based on information in the Layer 2 headers, such as an Ethernet MAC address.

Figure 9.8 depicts key elements in an L2VPN. Customers connect to the provider by means of a Layer 2 device, such as an Ethernet switch; the customer device that connects to the MPLS network is generally referred to as a customer edge (CE) device. The MPLS edge router is referred to as a provider edge (PE) device. The link between the CE and the PE operates at the link layer (for example, Ethernet), and is referred to as an attachment circuit (AC). The MPLS network then sets up an LSP that acts as a tunnel between two edge routers (that is, two PEs) that attach to two networks of the same enterprise. This tunnel can carry multiple virtual channels (VCs) using label stacking. In a manner very similar to VLAN stacking, the use of multiple MPLS labels enables the nesting of VCs.

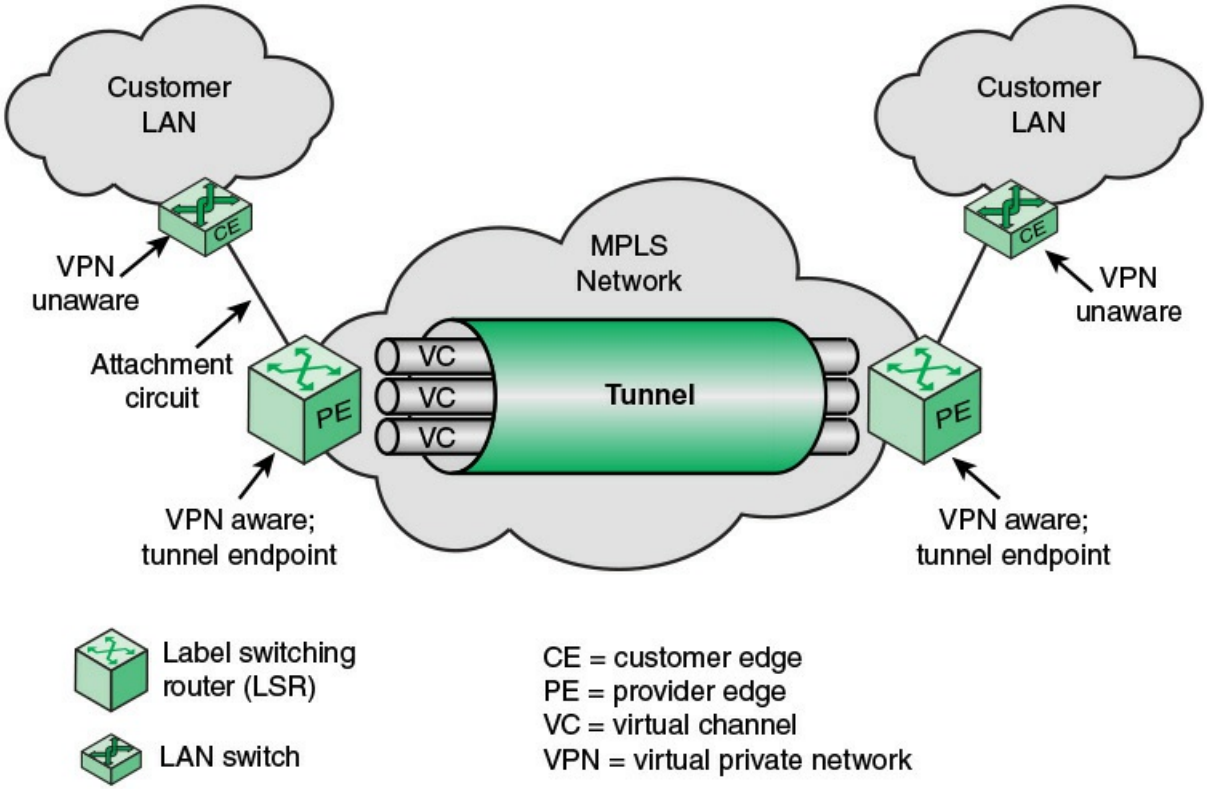


FIGURE 9.8 MPLS Layer 2 VPN Concepts

When a link-layer frame arrives at the PE from the CE, the PE creates an MPLS packet. The PE

pushes a label that corresponds to the VC assigned to this frame. Then the PE pushes a second label onto the label stack for this packet that corresponds to the tunnel between the source and destination PE for this VC. The packet is then routed across the LSP associated with this tunnel, using the top label for label switched routing. At the destination edge, the destination PE pops the tunnel label and examines the VC label. This tells the PE how to construct a link-layer frame to deliver the payload across to the destination CE.

If the payload of the MPLS packet is an Ethernet frame, the destination PE needs to be able to infer from the VC label the outgoing interface, and perhaps the VLAN identifier. This process is unidirectional, and will be repeated independently for bidirectional operation.

The VCs in the tunnel can all belong to a single enterprise, or it is possible for a single tunnel to manage VCs from multiple enterprises. In any case, from the point of view of the customer, a VC is a dedicated link-layer point-to-point channel. If multiple VCs connect a PE to a CE, this is logically the multiplexing of multiple link-layer channels between the customer and the provider.

Layer 3 MPLS VPN

Whereas L2VPNs are constructed based on link-level addresses (for example, MAC addresses), L3VPNs are based on VPN routes between CEs based on IP addresses. As with an L2VPN, an MPLS-based L3VPN typically uses a stack of two labels. The inner label identifies a specific VPN instance; the outer label identifies a tunnel or route through the MPLS provider network. The tunnel label is associated with an LSP and is used for label swapping and forwarding. At the egress PE, the tunnel label is stripped off, and the VPN label is used to direct the packet to the proper CE and to the proper logical flow at that CE.

For an L3VPN, the CE implements IP and is thus a router. The CE routers advertise their networks to the provider. The provider network can then use an enhanced version of Border Gateway Protocol (BGP) to establish VPNs between CEs. Inside the provider network, MPLS tools are used to establish routes between edge PEs supporting a VPN. Thus, the provider's routers participate in the customer's L3 routing function.

9.4 Network Virtualization

This section looks at the important area of network virtualization. One immediate difficulty is that this term is defined differently in a number of academic and industry publications. So we begin by defining some terms, based on definitions in ITU-T Y.3011 (*Framework of Network Virtualization for Future Networks*, January 2012):

- **Physical resource:** In the context of networking, physical resources include the following: network devices, such as routers, switches, and firewalls; and communication links, including wire and wireless. Hosts such as cloud servers may also be considered as physical network resources.
- **Logical resource:** An independently manageable partition of a physical resource, which inherits the same characteristics as the physical resource and whose capability is bound to the capability of the physical resource. An example is a named partition of disk memory.
- **Virtual resource:** An abstraction of a physical or logical resource, which may have

different characteristics from the physical or logical resource and whose capability may be not bound to the capability of the physical or logical resource. As examples, virtual machines (VMs) may be moved dynamically, VPN topologies can be altered dynamically, and access control restrictions may be imposed on a resource.

- **Virtual network:** A network composed of multiple virtual resources (that is, a collection of virtual nodes and virtual links) that is logically isolated from other virtual networks. Y.3011 refers to a virtual network as a logically isolated network partition (LINP).
- **Network virtualization (NV):** A technology that enables the creation of logically isolated virtual networks over shared physical networks so that heterogeneous collections of multiple virtual networks can simultaneously coexist over the shared physical networks. This includes the aggregation of multiple resources in a provider and appearing as a single resource.

NV is a far broader concept than VPNs, which only provide traffic isolation, or VLANs, which provide a basic form of topology management. NV implies full administrative control for customizing virtual networks both in terms of the physical resources used and the functionalities provided by the virtual networks.

The virtual network presents an abstracted network view whose virtual resources provide users with services similar to those provided by physical networks. Because the virtual resources are software defined, the manager or administrator of a virtual network potentially has a great deal of flexibility in altering topologies, moving resources, and changing the properties and service of various resources. In addition, virtual network users can include not only users of services or applications but also service providers. For example, a cloud service provider can quickly add new services or expanded coverage by leasing virtual networks as needed.

A Simplified Example

To get some feel for the concepts involved in network virtualization, we begin with a simplified example. Figure 9.9, adapted from the ebook *Software Defined Networking—A Definitive Guide* [KUMA13], shows a network consisting of three servers and five switches. One server is a trusted platform with a secure operating system that hosts firewall software. All the servers run a hypervisor (virtual machine monitor) enabling them to support multiple VMs. The resources for one enterprise (Enterprise 1) are hosted across the servers and consist of three VMs (VM1a, VM1b, and VM1c) on physical server 1, two VMs (VM1d and VM1e) on physical server 2, and firewall 1 on physical server 3. The virtual switches are used to set up any desired connectivity between the VMs across the servers through the physical switches. The physical switches provide the connectivity between the physical servers. Each enterprise network is layered as a separate virtual network on top of the physical network. Thus, the virtual network for Enterprise 1 is indicated in Figure 9.9 by a dashed circle and labeled VN1. The labeled circle VN2 indicates another virtual network.

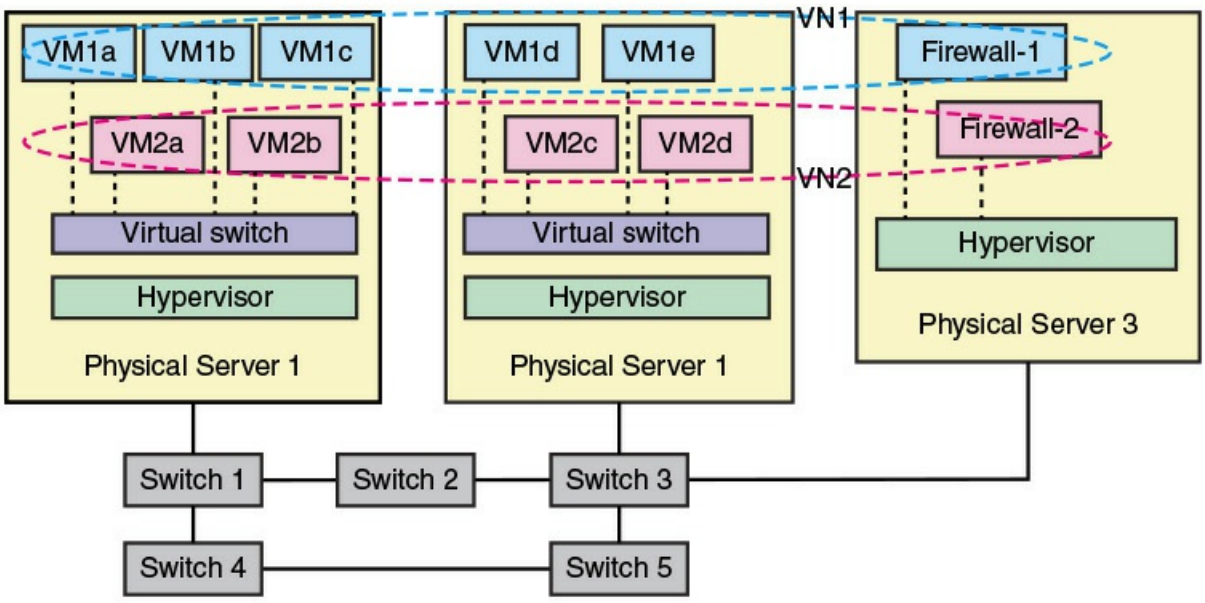


FIGURE 9.9 Simple Network with Virtual Machines Assigned to Different Administrative Groups

This example illustrates three layers of abstraction (see Figure 9.10). At the bottom are the physical resources, managed across one or more administrative domains. The servers are logically partitioned to support multiple VMs. This includes, at least, a partitioning of memory, but may also include a partitioning of the pool of I/O and communications ports and even of the processors or cores of the server. There is then an abstraction function that maps these physical and logical resources into virtual resources. This type of abstraction could be enabled by SDN and NFV functionality, and is managed by software at the virtual resource level.

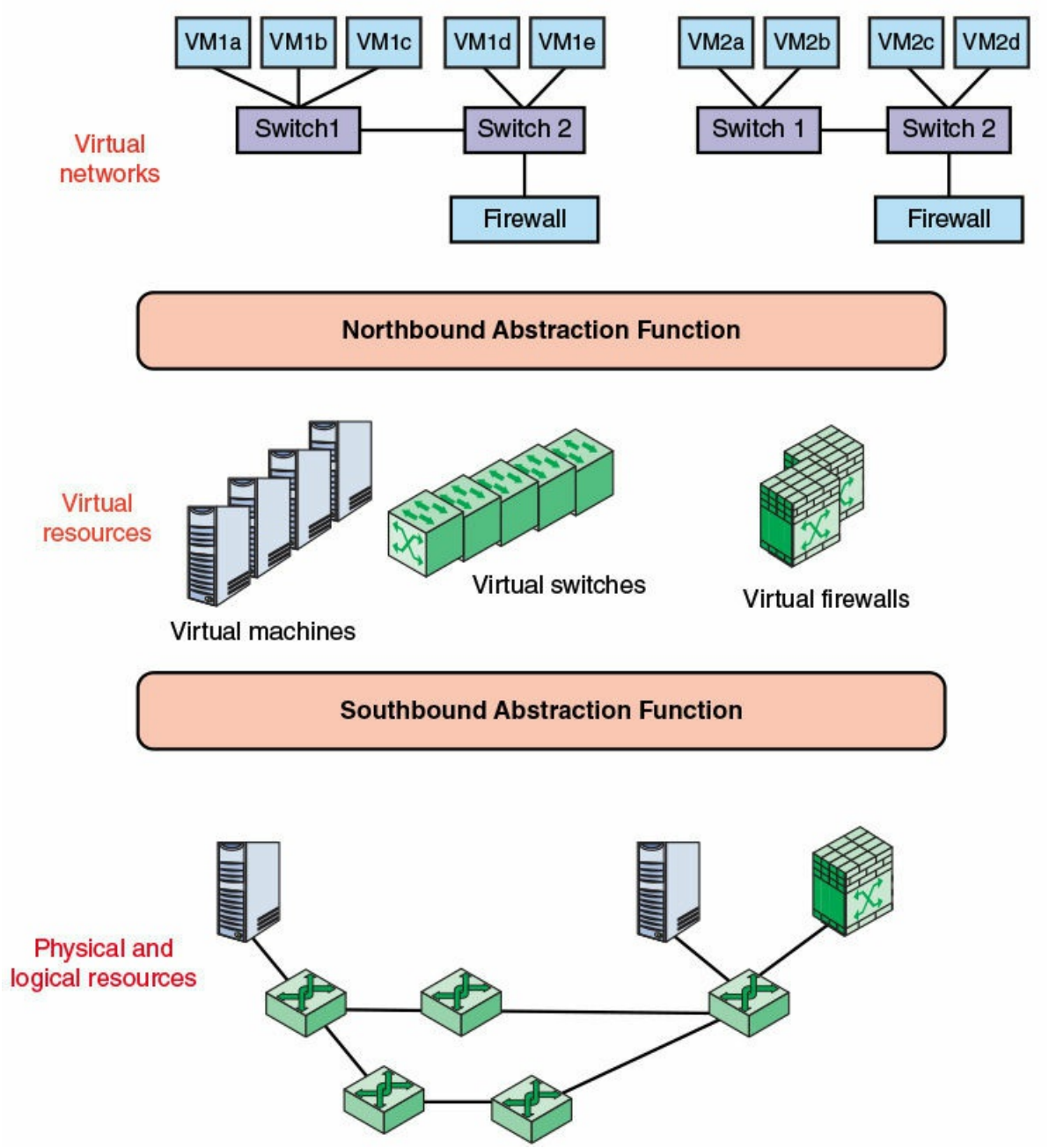


FIGURE 9.10 Levels of Abstraction for Network Virtualization

Another abstraction function is used to create network views organized as distinct virtual networks. Each virtual network is managed by a separate virtual network management function.

Because resources are defined in software, network virtualization provides a great deal of flexibility, as this example suggests. The manager of virtual network 1 may specify certain QoS requirements for traffic between VMs attached to switch 1 and VMs attached to switch 2, and may specify firewall rules for traffic external to the virtual network. These specification must ultimately be translated into forwarding rules configured on the physical switches and filtering rules on the physical firewall. Because it is all done in software and without the need for the virtual network manager to understand the physical topology and physical suite of servers,

changes are easily implemented.

Network Virtualization Architecture

An excellent overview of the many elements that contribute to an NV environment is provided by the conceptual architecture defined in Y.3011 and shown in Figure 9.11. The architecture depicts NV as consisting of four levels:

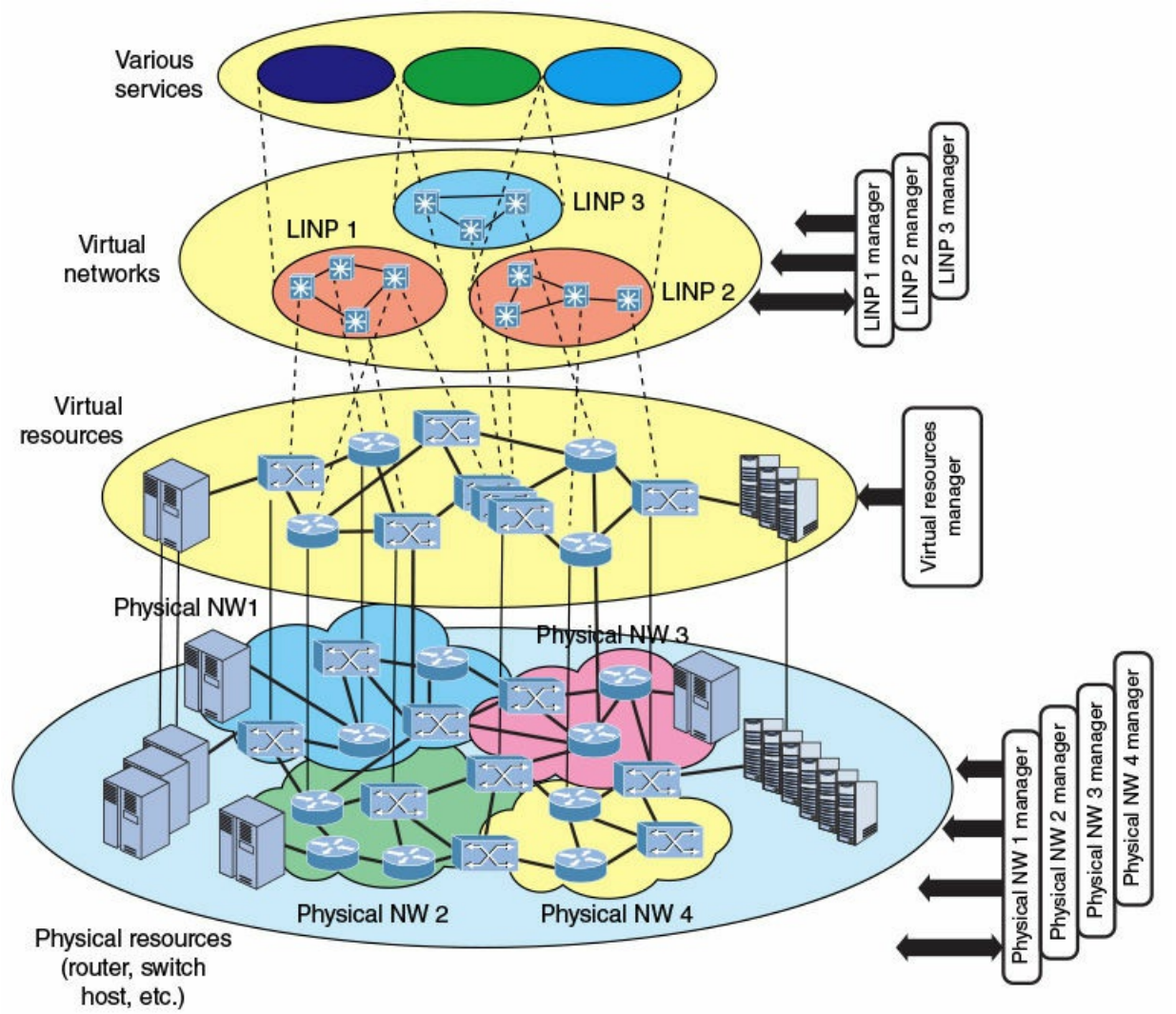


FIGURE 9.11 Conceptual Architecture of Network Virtualization (Y.3011)

- Physical resources
- Virtual resources
- Virtual networks
- Services

A single physical resource can be shared among multiple virtual resources. In turn, each LINP (virtual network) consists of multiple virtual resources and provides a set of services to users.

Various management and control functions are performed at each level, not necessarily by the same provider. There are management functions associated with each physical network and its associated resources. A virtual resource manager (VRM) manages a pool of virtual resources created from the physical resources. A VRM interacts with physical network managers (PNMs) to obtain resource commitments. The VRM constructs LINPs, and an LINP manager is allocated to each LINP.

Figure 9.12 provides another view of the NV architectural elements. Physical resource management manages physical resources and may create multiple logical resources that have the same characteristics as physical resources. Physical and logical resources are available to the virtual resource management at the interface between physical and virtual layers. The virtual resource management abstracts from the physical and logical resources to create virtual resources. It can also construct a virtual resource that combines other virtual resources. Virtual network management can build VNs on multiple virtual resources that are provided by the virtual resource management. Once a VN is created, the VN management starts to manage its own VN.

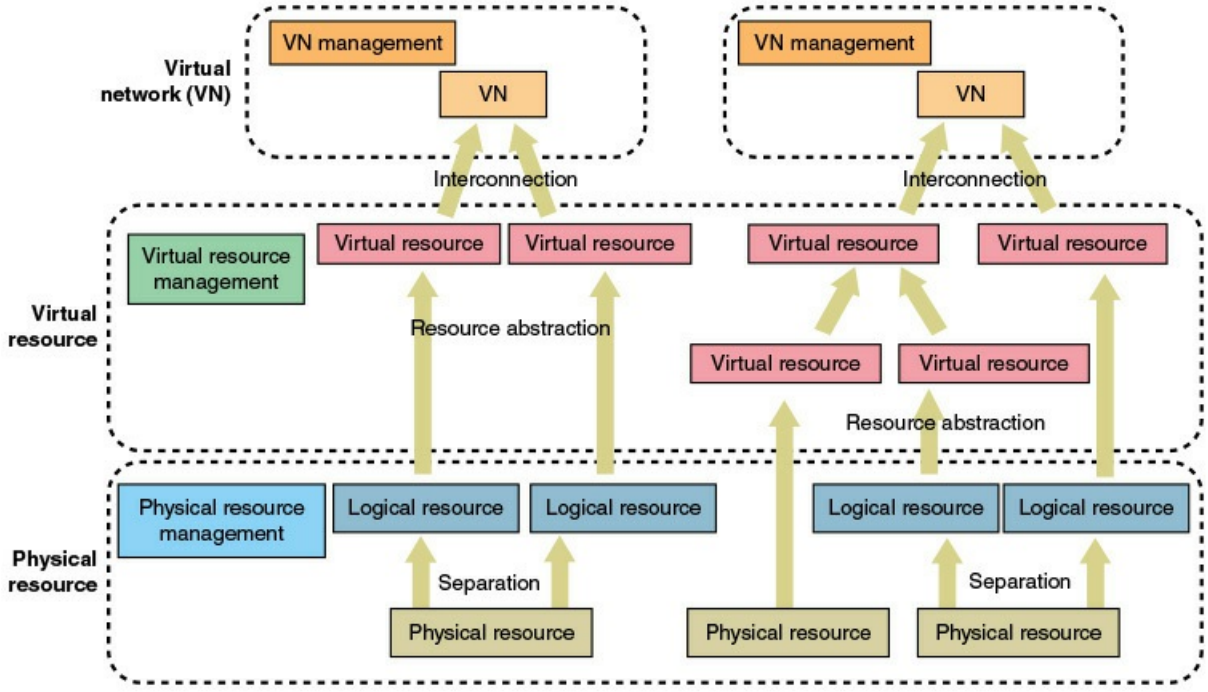


FIGURE 9.12 Network Virtualization Resource Hierarchical Model

Benefits of Network Virtualization

A 2014 survey [SDNC14] by SDxCentral of 220 organizations, including network service providers, small and medium-size businesses (SMB), large enterprises, and cloud service providers, reported the following benefits of NV (see Figure 9.13):

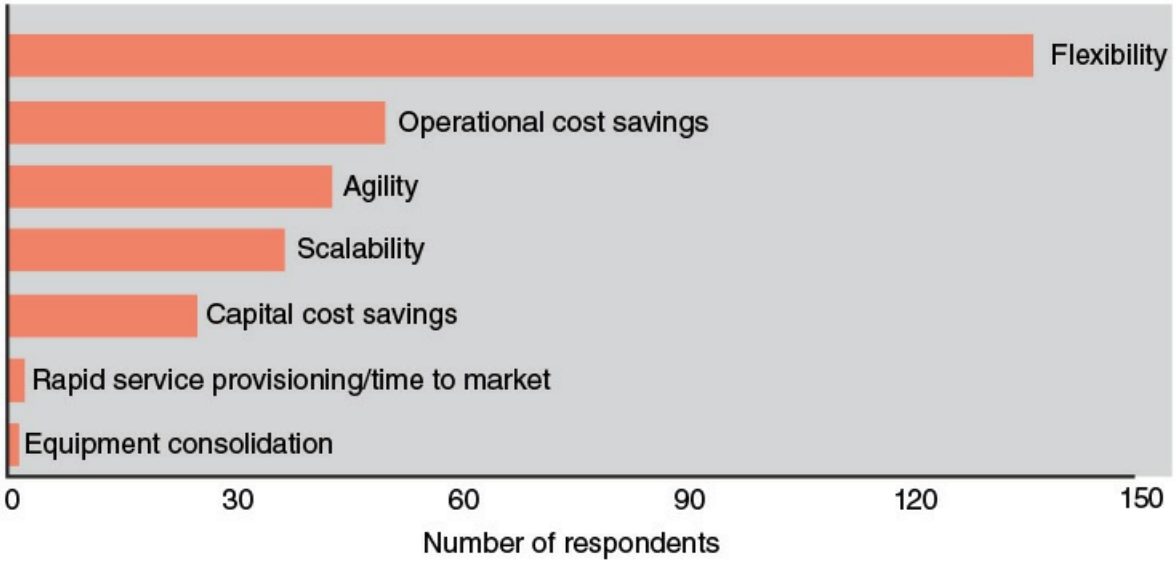


FIGURE 9.13 Reported Benefits of Network Virtualization

- **Flexibility:** NV enables the network to be quickly moved, provisioned, and scaled to meet the ever-changing needs of virtualized compute and storage infrastructures.
- **Operational cost savings:** Virtualization of the infrastructure streamlines the operational processes and equipment used to manage the network. Similarly, base software can be unified and more easily supported, with a single unified infrastructure to manage services. This unified infrastructure also allows for automation and orchestration within and between different services and components. From a single set of management components, administrators can coordinate resource availability and automate the procedures necessary to make services available, reducing the need for human operators to manage the process and reducing the potential for error.
- **Agility:** Modifications to the network's topology or how traffic is handled can be tried in different ways, without needing to modify the existing physical networks.
- **Scalability:** A virtual network can be rapidly scaled to respond to shifting demands by adding or removing physical resources from the pool of available resources.
- **Capital cost savings:** A virtualized deployment can reduce the number of devices needed, providing capital as well as operational costs savings.
- **Rapid service provisioning/time to market:** Physical resources can be allocated to virtual networks on demand, so that within an enterprise resources can be quickly shifted as demand by different users or applications changes. From a user perspective, resources can be acquired and released to minimize utilization demand on the system. New services require minimal training and can be deployed with minimal disruption to the network infrastructure.
- **Equipment consolidation:** NV enables the more efficient use of network resources, thus allowing for consolidating equipment purchases to fewer, more off-the-shelf products.

9.5 OpenDaylight's Virtual Tenant Network

Virtual Tenant Network (VTN) is an OpenDaylight (ODL) plug-in developed by NEC. It provides multitenant virtual networks on an SDN, using VLAN technology. The VTN abstraction functionality enables users to design and deploy a virtual network without knowing the physical network topology or bandwidth restrictions. VTN allows the users to define the network with a look and feel of a conventional L2/L3 (LAN switch/IP router) network. Once the network is designed on VTN, it is automatically mapped onto the underlying physical network, and then configured on the individual switches leveraging the SDN control protocol.

VTN consists of two components (see Figure 5.6 in Chapter 5, "SDN Control Plane"):

- **VTN Manager:** An ODL controller plug-in that interacts with other modules to implement the components of the VTN model. It also provides a REST interface to configure VTN components in the controller.
- **VTN Coordinator:** An external application that provides a REST interface to users for VTN virtualization. It interacts with VTN Manager plug-in to implement the user configuration. It is also capable of multiple controller orchestration.

Table 9.1 shows the elements that are building blocks for constructing a virtual network. A virtual network is constructed using virtual nodes (vBridge, vRouter) and virtual interfaces and links. It is possible, by connecting the virtual interfaces made on virtual nodes via virtual links, to configure a network that has L2 and L3 transfer function.

Name of Element		Description
Virtual node	vBridge	Logical representation of L2 switch function.
	vRouter	Logical representation of L3 router function. Only one vRouter can be defined in a VTN, and it can connect only to the vBridge.
	vTerminal	Logical representation of a virtual node that is connected to an interface mapped to a physical port that is the source or target of a redirect section attribute in a flow filter.
	vTunnel	Logical representation of Tunnel (consists of vTeps and vBypasses).
	vTep	Logical representation of tunnel endpoint (TEP).
	vBypass	Logical representation of connectivity between controlled networks.
Virtual interface	Interface	Representation of endpoint on the virtual node (VM, servers, vBridge, vRouter, and so on).
Virtual link	vLink	Logical representation of L1 connectivity between virtual interfaces.

TABLE 9.1 Virtual Tenant Network Elements

The upper part of Figure 9.14 is a virtual network example. VRT is defined as the vRouter, and BR1 and BR2 are defined as vBridges. Interfaces of the vRouter and vBridges are connected using vLinks. Once a user of VTN Manager has defined a virtual network, the VTN Coordinator maps physical network resources to the constructed virtual network. Mapping identifies which virtual network each packet transmitted or received by an OpenFlow switch belongs to, as well

as which interface in the OpenFlow switch transmits or receives that packet. There are two mapping methods:

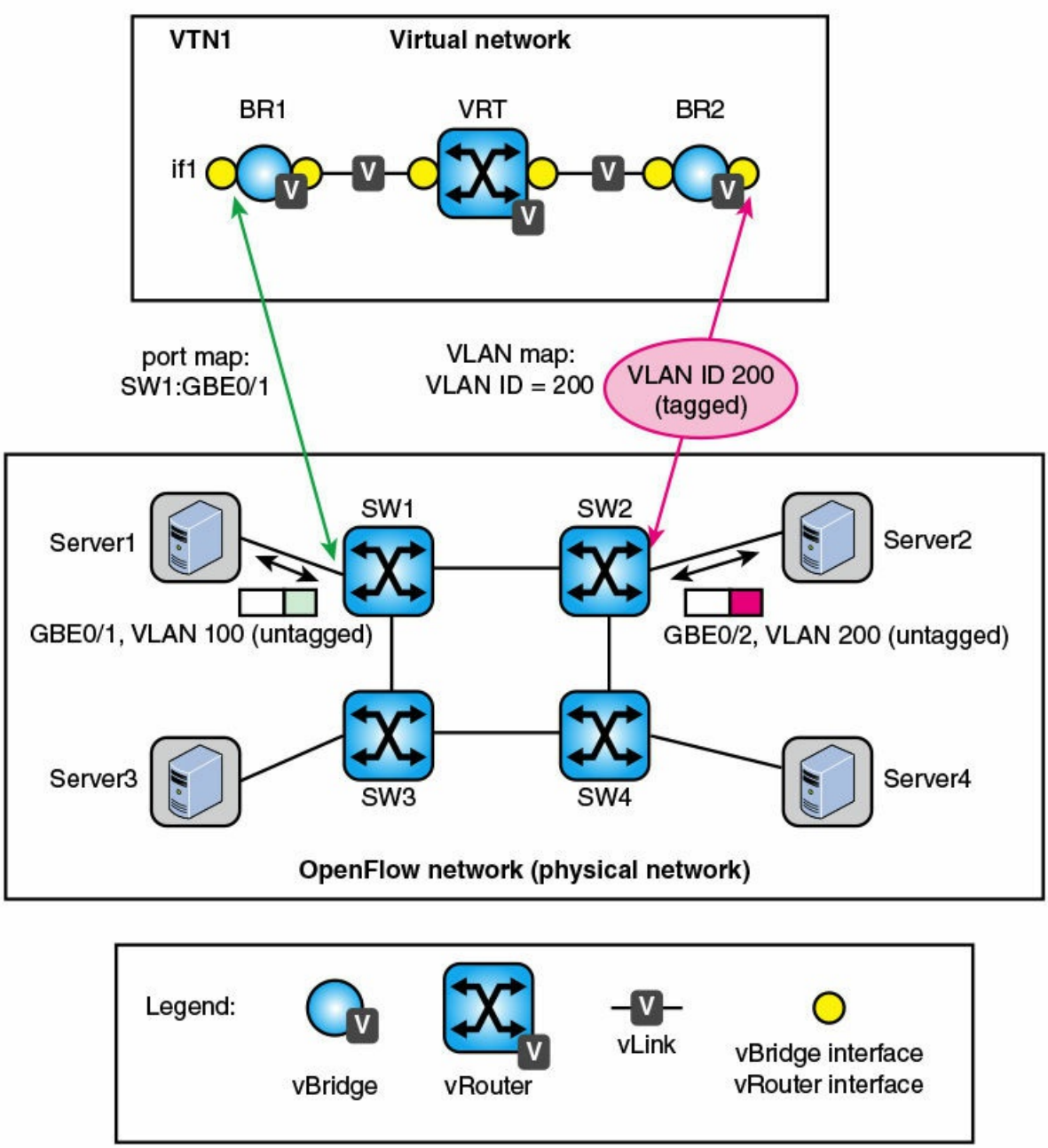


FIGURE 9.14 VTN Mapping Example

- **Port mapping:** This mapping method is used to map a physical port as an interface of virtual node (vBridge/vTerminal). Port-map is enabled when the network topology is known in advance.
- **VLAN mapping:** This mapping method is used to map VLAN ID of VLAN tag in incoming Layer 2 frame with the vBridge. This mapping is used when the affiliated network and its VLAN tag are known. Whenever this mapping method is used, it is

possible to reduce the number of commands to be set.

Figure 9.14 shows a mapping example. An interface of BR1 is mapped to a port on OpenFlow switch SW1. Packets received from that SW1 port are regarded as those from the corresponding interface of BR1. The interface if1 of vBridge (BR1) is mapped to the port GBE0/1 of switch1 using **port-map**. Packets received or transmitted by GBE0/1 of switch1 are considered as those from or to the interface if1 of vBridge. vBridge BR2 is mapped to VLAN 200 using **vlan-map**. Packets having the VLAN ID of 200 received or transmitted by the port of any switch in the network are mapped to the vBridge BR2.

VTN provides the capability to define and manage traffic flows across a virtual network. As with OpenFlow, flows are defined based on the value of various fields in packets. A flow can be defined using one or a combination of the following fields:

- Source MAC Address
- Destination MAC Address
- Ethernet Type
- VLAN Priority
- Source IP Address
- Destination IP Address
- IP Version
- **Differentiated Services Codepoint (DSCP)**
- TCP/UDP Source Port
- TCP/UDP Destination Port
- ICMP Type
- ICMP Code

Table 9.2 outlines the types of Action that can be applied on packets that match the Flow Filter conditions.

Action	Description
Pass	Pass packets matching specified conditions.
Drop	Drop packets matching specified conditions.
Redirect	Redirect packets to a specified virtual interface. Both transparent redirection (not changing MAC address) and router redirection (changing MAC address) are supported.
Priority	Set a priority for packets, using IP DSCP field.
Bandwidth	Set policing parameters. Set action based on threshold data rate statistics. Actions include pass, drop, and reduce priority.
Statistics	Collect statistics information.

TABLE 9.2 Virtual Tenant Flow Filter Actions

Figure 9.15 shows the overall architecture of VTN. The VTN Manager is part of the

OpenDaylight controller and uses base network service functions to learn the topology and statistics of the underlying network. A user or application creates virtual networks and specifies network behavior to the VTN Coordinator across a web or REST interface. The VTN Coordinator translates these commands into detailed instructions to the VTN Manager, which in turn uses OpenFlow to map virtual networks to the physical network infrastructure.

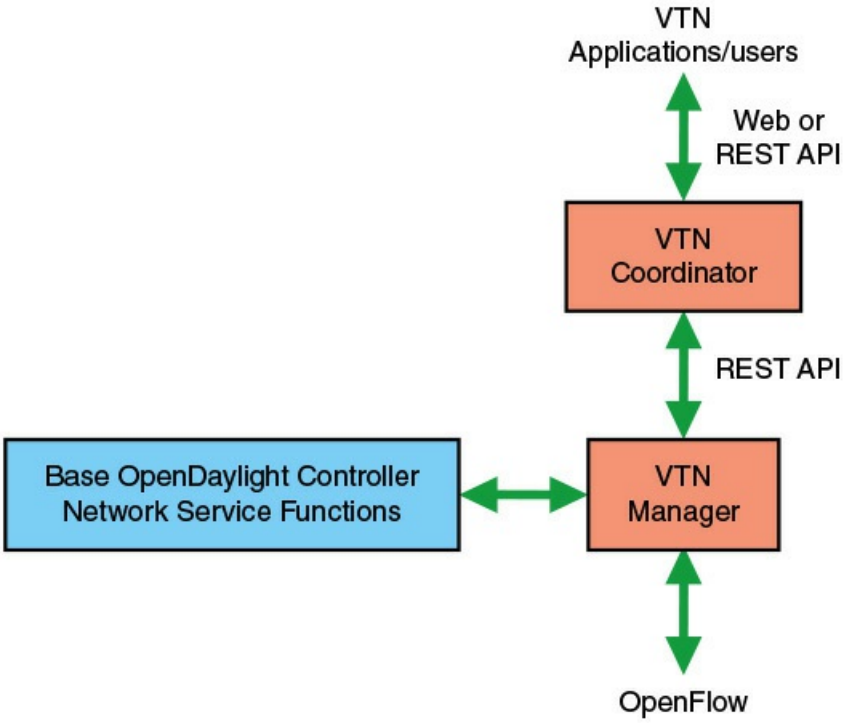


FIGURE 9.15 OpenDaylight VTN Architecture

9.6 Software-Defined Infrastructure

Recent years have seen explosive growth in the complexity of data centers, cloud computing facilities, and network infrastructures for enterprises and carriers. An emerging design philosophy to address the challenges of this complexity is software-defined infrastructure (SDI). With SDI, a data center or network infrastructure can autoconfigure itself at run time based on application/business requirements and operator constraints. Automation in SDIs enables infrastructure operators to achieve higher conformance to SLAs, avoid overprovisioning, and automate security and other network-related functions.

Another key characteristic of SDI is that it is highly application driven. Applications tend to change much more slowly than the ecosystem (hardware, system software, networks) that supports them. Individuals and enterprises stay with chosen applications for long periods of time, whereas they replace the hardware and other infrastructure elements at a fast pace. So, providers are at an advantage if the entire infrastructure is software defined and thus able to cope with rapid changes in infrastructure technology.

SDN and NFV are the key enabling technologies for SDI. SDN provides network control systems with the flexibility to steer and provision network resources dynamically. NFV

virtualizes network functions as prepackaged software services that are easily deployable in a cloud or network infrastructure environment. So instead of hard-coding a service deployment and its network services, these can now be dynamically provisioned; traffic is then steered through the software services, significantly increasing the agility with which these are provisioned. Although SDN and NFV are necessary components of an SDI, they do not by themselves provide the intelligence that can generate or recommend the required configuration that can then be automatically implemented. Therefore, we can think of SDN and NFV as providing a platform for deploying SDI-enabling software.

A recent paper by Pott [POTT14] lists the following as some of the key features of an SDI offering:

- Distributed storage resources with fully inline **data deduplication** and compression.
- Fully automated and integrated backups that are application aware, with autoconfiguring and autotesting. This new generation will be as close to "zero touch" as is possible.
- Fully automated and integrated disaster recovery that is application aware, with autoconfiguring and autotesting. This new generation will be as close to "zero touch" as is possible.
- Fully integrated hybrid cloud computing, with resources in the public cloud consumed as easily as local. The ability to move between multiple cloud providers, based on cost, data sovereignty requirements, or latency/locality needs. The providers that want to win the hybrid cloud portion of the exercise will build in awareness of privacy and security and allow administrators to easily select not only geolocal providers, but those known to have zero foreign legal attack surface, and they will clearly differentiate between them.
- WAN optimization technology.
- A hypervisor or hypervisor/container hybrid running on the metal.
- Management software to allow administrators to manage the hardware and the hypervisor.
- Adaptive monitoring software that will detect new applications and operating systems and automatically monitor them properly. Adaptive monitoring will not require manual configuration.
- Predictive analytics software that will determine when resources will exceed capacity, when hardware is likely to fail, or when licensing can no longer be worked around.
- Automation and load maximization software that will make sure the hardware and software components are used to their maximum capacity, given the existing hardware and existing licensing bounds.
- Orchestration software that will not only spin up groups of applications on demand or as needed, but will provide an "App Store"-like experience for selecting new workloads and getting them up and running on your local infrastructure in just a couple of clicks.
- Autobursting, as an adjunct of orchestration, will intelligently decide between hot-adding capacity to legacy workloads (CPU, RAM, and so on) or spinning up new instances of modern burstable applications to handle load. It would, of course, scale them back down when possible.

- Hybrid identity services that work across private infrastructure and public cloud spaces. They will not only manage identity but also provide complete user experience management solutions that work anywhere.
- Complete software-defined networking stack, including Layer 2 extension between data centers as well as the public and private cloud. This means that spinning up a workload will automatically configure networking, firewalls, intrusion detection, application layer gateways, mirroring, load balancing, content distribution network registration, certificates, and so forth.
- Chaos creation in the form of randomized automated testing for failure of all nonlegacy workloads and infrastructure elements to ensure that the network still meets requirements.

Software-Defined Storage

As mentioned, SDN and NFV are key elements of SDI. A third, equally important element is the emerging technology known as software-defined storage (SDS). SDS is a framework for managing a variety of storage systems in the data center that are traditionally not unified. SDS provides the ability to manage these storage assets to meet specific SLAs and to support a variety of applications. The dominant physical architecture for SDS is based on distributed storage, with storage devices distributed across a network.

Figure 9.16 illustrates the main elements of a typical SDS architecture. Physical storage consists of a number of magnetic and solid-state disk arrays, possibly from multiple vendors. Separate from this physical storage plane is a unified set of control software. This must include adaptation logic that can interface with a variety of vendor equipment and controlling and monitoring that equipment. On top of this adaptation layer are a number of basic storage services. An application interface provides an abstracted view of data storage so that applications need not be concerned with the location, attributes, or capacity of individual storage systems. There is also an administrative interface to enable the SDS administrator to manage the distributed storage suite.

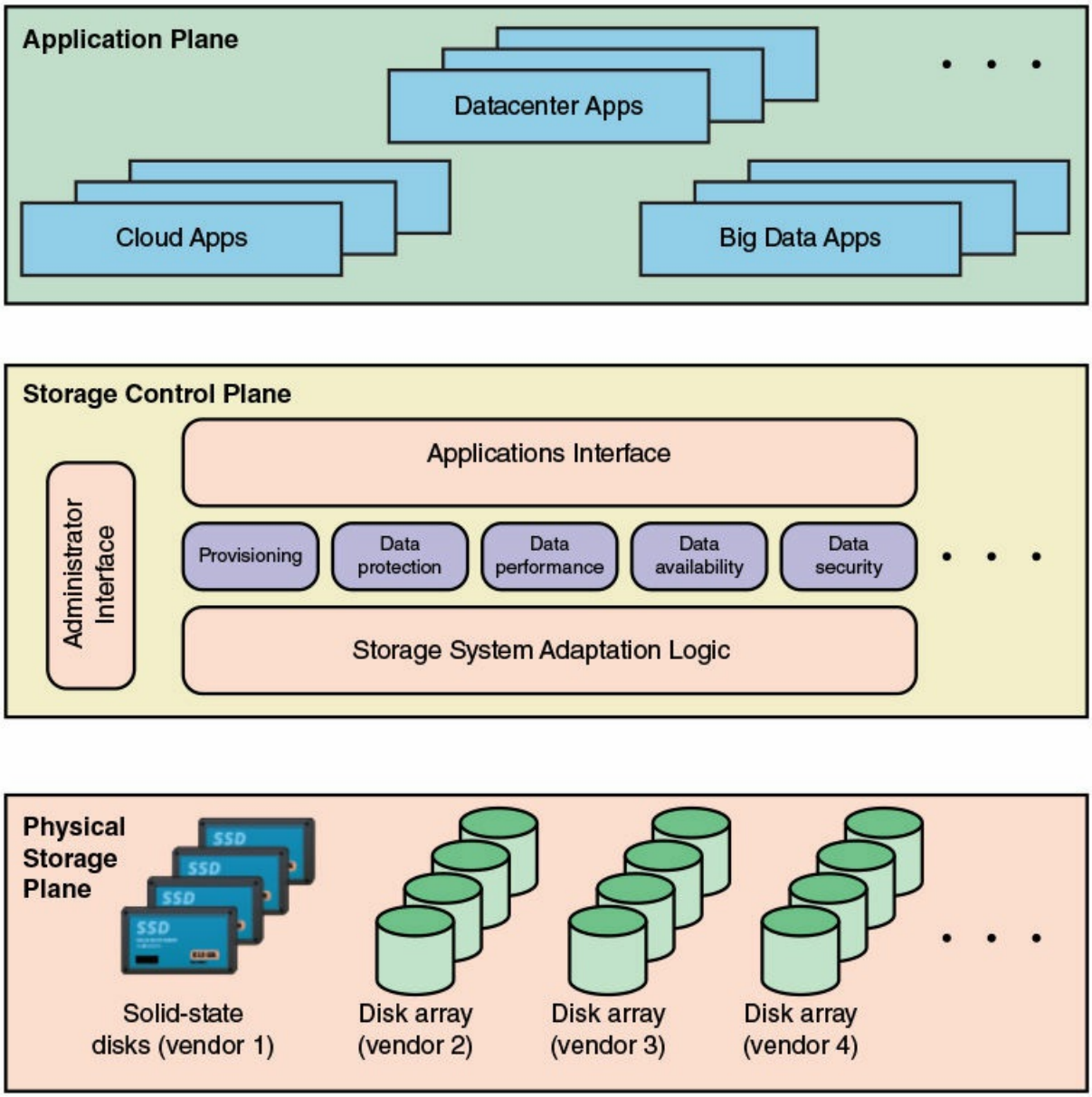


FIGURE 9.16 Software-Defined Storage Architecture

SDS puts the emphasis on storage services instead of storage hardware. By decoupling the storage control software from the hardware, a storage resource can be used more efficiently and its administration simplified. For example, a storage administrator can use SLAs when deciding how to provision storage without needing to consider specific hardware attributes. In essence, resources are aggregated into storage pools assigned to users. Data services are applied to meet user or application requirements, and service levels are maintained. When additional resources are needed by an application, the storage control software automatically adds the resources. Conversely, resources are freed up when not in use. The storage control software automatically removes failed components and systems that fail.

SDI Architecture

A number of companies, including IBM, Cisco, Intel, and HP, either have produced or are working on SDI offerings. There is no standardized specification for SDI, and there are numerous differences in the different initiatives. Nevertheless, the overall SDI architecture is quite similar among the different efforts. A typical example is the SDI architecture defined by Intel. This architecture is organized into three layers, as illustrated in Figure 9.17 and described in the list that follows.

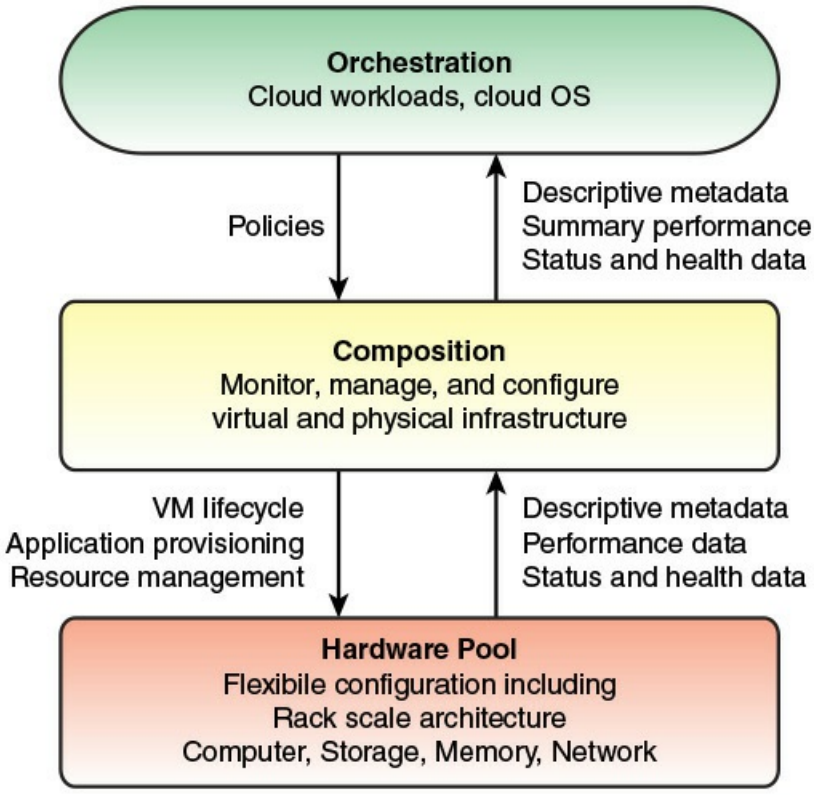


FIGURE 9.17 Intel's 3-Layer SDI Model

Orchestration: A policy engine that allows higher level frameworks to manage composition dynamically without interrupting ongoing operations.

Composition: A low-level layer of system software that continually and automatically manages the pool of hardware resources.

Hardware pool: An abstracted pool of modular hardware resources.

The orchestration layer drives the architecture. This layer is concerned with efficient configuration or resources while at the same time meeting application service requirements. Intel's initial focus appears to be on cloud providers, but other application areas, such as big data and other data center applications, lend themselves to the SDI approach. This layer continually monitors status data, enabling it to solve service issues faster and to continually optimize hardware resource assignment.

The composition layer is a control layer that manages VMs, storage, and network assets. In this architecture, the VM is seen as a dynamic federation of compute, storage, and network resources assembled to run an application instance. Although current VM technology provides a level of

flexibility and cost savings over the use of nonvirtualized servers, there is still considerable inefficiency. Suppliers tend to size systems to meet the maximum demand that a VM might impose and hence overprovision so as to guarantee service. With software-defined allocation of resources, more flexibility is available in creating, provisioning, managing, moving, and retiring VMs. Similarly, SDS provides the opportunity to use storage more efficiently.

Composition enables the logical disaggregation of compute, network, and storage resources, so that each VM provides exactly what an application needs. Supporting this at the level of the hardware is Intel's rack scale architecture (RSA). RSA exploits extremely high data rate optical connection components to redesign the way computer rack systems are implemented. In an RSA design, the speed of the silicon interconnects means that individual components (processors, memory, storage, and network) no longer need to reside in the same box. Individual racks can be dedicated to each of the component classes and scaled to meet the demands of the data center.

Figure 9.18 provides another view of Intel's SDI architecture, which is in general terms typical of SDI architectures from other organizations. The resource pool consists of storage, network, and compute resources. From a hardware perspective, these can be deployed in an RSA. From a control perspective, SDS, SDN, and NFV technologies enable the management of these resources with an overall SDI framework.

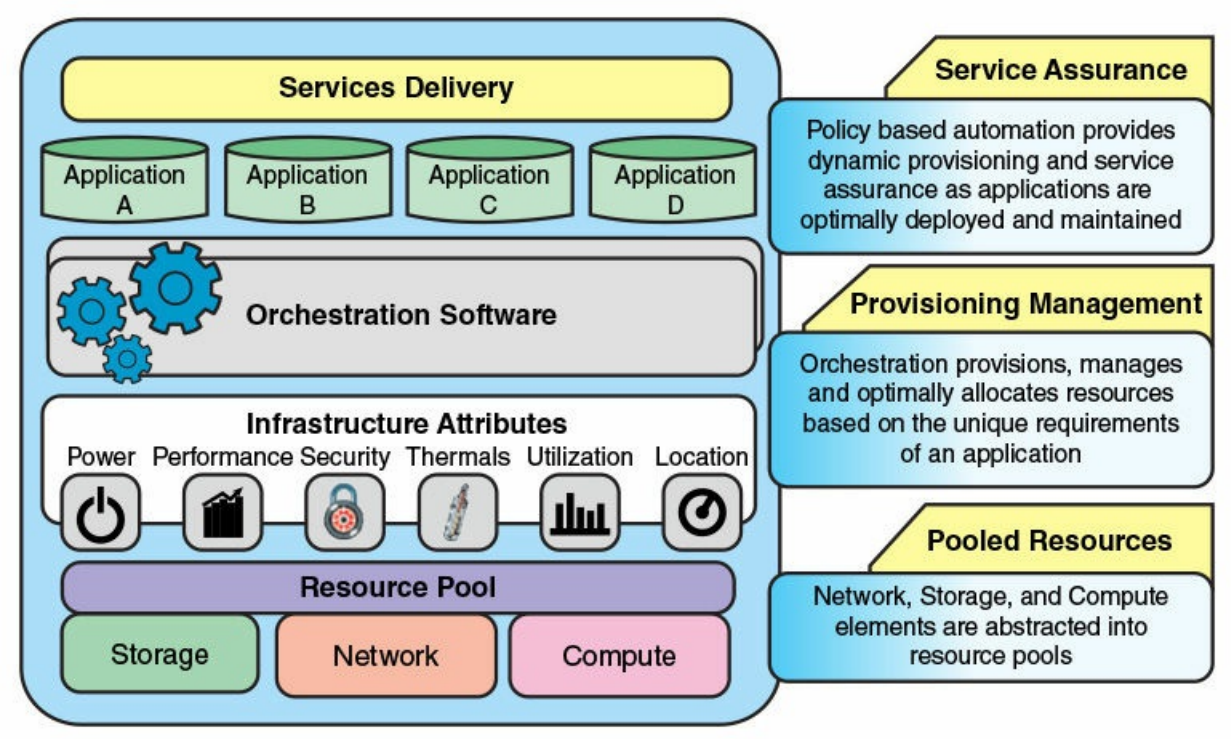


FIGURE 9.18 Intel's SDI Architecture

9.7 Key Terms

After completing this chapter, you should be able to define the following terms.

broadcast addressing

broadcast domain

data deduplication

differentiated services codepoint (DSCP)

IEEE 802.3

IP security (IPsec)

LAN switch

logical resource

MAC frame

network virtualization

physical resource

software-defined infrastructure (SDI)

software-defined storage (SDS)

unicast addressing

virtual LAN (VLAN)

virtual network

virtual private network (VPN)

virtual resource

Virtual Tenant Network (VTN)

9.8 References

KUMA13: Kumar, R. Software Defined Networking—a Definitive Guide. Smashwords.com, 2013.

POTT14: Pott, T. "SDI Wars: WTF Is Software Defined Center Infrastructure?" *The Register*, October 17, 2014. http://www.theregister.co.uk/2014/10/17/sdi_wars_what_is_software_defined_infrastruct

SDNC14: SDNCentral. SDNCentral Network Virtualization Report, 2014 Edition, 2014.

Part IV: Defining and Supporting User Needs

We are beginning to understand, or at least to appreciate, the cause of time delays and overloading phenomena in communication systems handling competing users with different levels of importance. There is a basis for hope that one day we may be able to automate highly sophisticated priority systems. Such systems may even be so effective as to provide the operational equivalent of exercised judgment.

—On Distributed Communications, Introduction to Distributed Communications Networks, Report RM-3420-PR, Paul Baran, August 1964

Fundamental to the acceptance and success of any complex shared networking architecture is that it meets users expectations for performance. Traditionally, the means of defining expected performance, measuring it, providing it, and entering into well-defined agreements relating to it has been the concept of quality of service (QoS). QoS remains an essential ingredient in any network design. Chapter 10 provides an overview of QoS concepts and standards. Recently, QoS has been augmented with the concept of quality of experience (QoE), which is particularly relevant to interactive video and multimedia network traffic. Chapter 11 provides an overview of QoE and discusses a number of practical aspects of implementing QoE mechanisms. Chapter 12 looks further into the network design implications of the combined use of QoS and QoE.

Chapter 10. Quality of Service

> In the schemes considered, precedence is determined moment-by-moment, automatically for all traffic in the network. Precedence is computed as a composite function of: (1) the ability of the network to accept additional traffic; (2) the "importance" of each user and the "utility" of his traffic; (3) the data rate of each input transmission medium or the transducer used; and (4) the tolerable delay time for delivery of the traffic.
>
> —*On Distributed Communications: Priority, Precedence, and Overload*, Rand Report RM-3638-PR, Paul Baran, August 1964

Chapter Objectives: **After studying this chapter, you should be able to**

- Describe the ITU-T QoS architectural framework.
- Summarize the key concepts of the Integrated Services Architecture.
- Compare and contrast elastic and inelastic traffic.
- Explain the concept of differentiated services.
- Understand the use of service level agreements.
- Describe IP performance metrics.
- Present an overview of OpenFlow QoS support.

The Internet and enterprise IP-based networks continue to see rapid growth in the volume and variety of data traffic. Cloud computing, big data, the pervasive use of mobile devices on enterprise networks, and the increasing use of video streaming all contribute to the increasing difficulty in maintaining satisfactory network performance. Two key tools in measuring the network performance that an enterprise desires to achieve are quality of service (QoS) and quality of experience (QoE). As is discussed in Chapter 2, "Requirements and Technology," QoS is the measurable end-to-end performance properties of a network service, which can be guaranteed in advance by a service level agreement (SLA) between a user and a service provider, so as to satisfy specific customer application requirements. QoE is a subjective measure of performance as reported by the user. Unlike QoS, which can be precisely measured, QoE relies on human opinion.

QoS and QoE enable the network manager to determine whether the network is meeting user needs and to diagnose problem areas that require adjustment to network management and network traffic control. This chapter looks in some detail at QoS. Chapter 11, "QoE: User Quality of Experience," and Chapter 12, "Network Design Implications of QoS and QoE," examine QoE, the relationship between QoS and QoE, and the design implications of a QoE/QoS architecture.

There is a strong need to be able to support a variety of traffic, with a variety of QoS requirements, on IP-based networks. This chapter begins with a look at an overall QoS architecture, which describes internetwork functions and services designed to meet this need.

Next, the chapter looks at the Integrated Services Architecture (ISA), which provides a framework for current and future Internet services. We then examine the key concept of differentiated services. The chapter concludes with an introduction to the topics of SLAs and IP performance metrics.

You might find it useful at this point to review Section 2.1, "Types of Network and Internet Traffic," which reviews various types of traffic and their QoS requirements.

10.1 Background

Historically, the Internet and other IP-based networks provided a **best effort** delivery service. This means that the network attempts to allocate its resources with equal availability and priority to all traffic flows, with no regard for application priorities, traffic patterns and load, and customer requirements. To protect the network from congestion collapse and to guarantee that some flows do not crowd out other flows, congestion control mechanisms were introduced, which tended to throttle traffic that consumed excessive resources.

One of the most important congestion control techniques, introduced early on and still in wide use, is the TCP congestion control mechanism. TCP congestion control has become increasingly complex and sophisticated, but it is worth briefly summarizing the principles involved here. For each TCP connection between two end systems across a network, in each direction, a concept known as sliding window is used. TCP segments on a connection are numbered sequentially. The sending and receiving TCP entities maintain a window, or buffer, that defines the range of sequence numbered segments that may be transmitted. As segments arrive and are processed by the receiver, the receiver returns an acknowledgment indicating which segments have been received and implicitly indicated to the sender that the window of sequence numbers has advanced to allow more segments to be sent. Various algorithms are used by the sender to deduce the amount of congestion on a connection based on the round-trip delay for acknowledgments plus whether an acknowledgment is even received for a particular segment. As congestion is detected, the sending TCP entity reduces its transmission of segments to help ease congestion on the intervening network.

TCP can work well if all the TCP connections across an Internet conform to the congestion control mechanism. The scheme is less effective if some connections, on behalf of "selfish" applications, ignore the congestion control rules and attempt to send segments as rapidly as possible.

Although TCP congestion control and other network congestion control techniques can reduce the risk of excessive congestion, these techniques do not directly address QoS requirements. As the intensity and variety of traffic increased, various QoS mechanisms were developed, including Integrated Services Architecture (ISA) and differentiated services (DiffServ), accompanied by service level agreements (SLAs) so that the service provided to various customers was tunable and somewhat predictable. These mechanisms and services serve two purposes:

- Allocate network resources efficiently so as to maximize effective capacity
- Enable networks to offer different levels of QoS to customers on the basis of customer requirements

In this more sophisticated environment, the term *best effort* refers not to the network service as a

whole but to a class of traffic treated in best effort fashion. All packets in the best effort traffic class are transmitted with no guarantee regarding the speed with which the packets will be transmitted to the recipient or that the data will even be delivered entirely. Typically, in a network that provides multiple levels of service, best effort is the classification for the lowest priority traffic. However, for some applications, a class of traffic known as *lower than best effort*, or *lower effort* (LE), may be used. LE classification permits a network operator to strictly limit the effect of LE traffic on best effort/normal or all other network traffic. This classification may be suitable for background data transfer applications (such as file sharing and update fetching) or traffic that could be delayed to off-peak times.

10.2 QoS Architectural Framework

Before looking at the Internet standards that deal with provision of QoS in the Internet and private internetworks, it is useful to consider an overall architectural framework that relates the various elements that go into QoS provision. Such a framework has been developed by the Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T) as part of its Y series of Recommendations.[1] Recommendation Y.1291, *An Architectural Framework for Support of Quality of Service in Packet Networks*, gives a "big picture" overview of the mechanisms and services that comprise a QoS facility.

[1] The Y series, titled *Global Information Infrastructure, Internet Protocol Aspects and Next-Generation Networks*, contains a number of very useful documents dealing with QoS, congestion control, and traffic management.

The Y.1291 framework consists of a set of generic network mechanisms for controlling the network service response to a service request, which can be specific to a network element, or for signaling between network elements, or for controlling and administering traffic across a network. Figure 10.1 shows the relationship among these elements, which are organized into three planes: data, control, and management. This architectural framework is an excellent overview of QoS functions and their relationships and provides a useful basis for summarizing QoS.

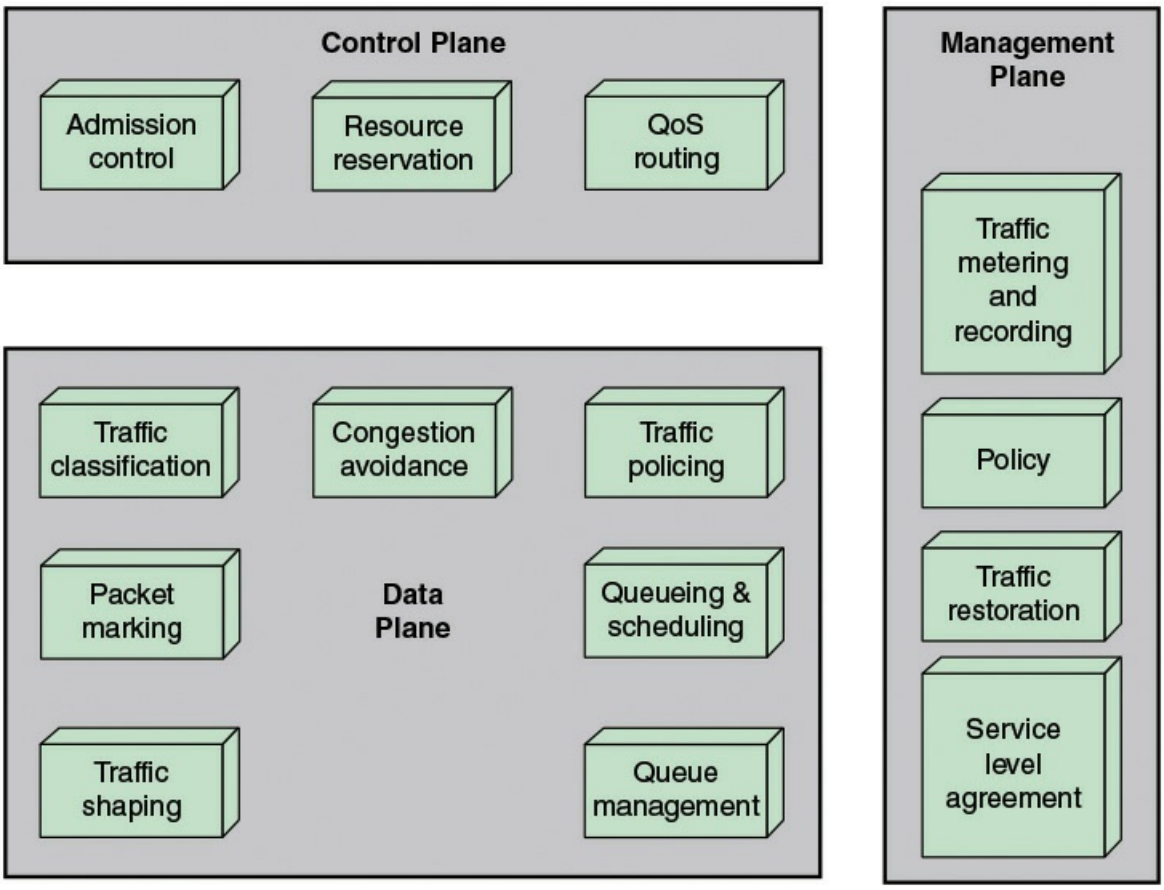


FIGURE 10.1 Architectural Framework for QoS Support

Data Plane

The data plane includes those mechanisms that operate directly on flows of data. The following discussion briefly describes each mechanism in turn.

Traffic classification refers to the assignment of packets to a traffic class by the ingress router at the ingress edge of the network. Typically, the classification entity looks at multiple fields of a packet, such as source and destination address, application payload, and QoS markings, and determines the aggregate to which the packet belongs. This classification provides network elements a method to weigh the relative importance of one packet over another in a different class. All traffic assigned to a particular flow or other aggregate can be treated similarly. The flow label in the IPv6 header can be used for traffic classification. Other routers en route perform a classification function as well, but the classification does not change as the packets traverse the network.

Packet marking encompasses two distinct functions. First, packets may be marked by ingress edge nodes of a network to indicate some form of QoS that the packet should receive. An example is the Differentiated Services (DiffServ) field in the IPv4 and IPv6 packets and the Traffic Class field in MPLS labels. An ingress edge node can set the values in these fields to indicate a desired QoS. Such markings may be used by intermediate nodes to provide differential

treatment to incoming packets. Second, packet marking can also be used to mark packets as nonconformant, either by the ingress node or intermediate nodes, which may be dropped later if congestion is experienced.

Traffic shaping controls the rate and volume of traffic entering and transiting the network on a per-flow basis. The entity responsible for traffic shaping buffers nonconformant packets until it brings the respective aggregate in compliance with the traffic limitations for this flow. The resulting traffic thus is not as bursty as the original and is more predictable. For example, Y.1221 recommends the use of leaky bucket/token bucket for traffic shaping. Typically, this is a function performed at the ingress edge.

Congestion avoidance deals with means for keeping the load of the network under its capacity such that it can operate at an acceptable performance level. The specific objectives are to avoid significant queuing delays and, especially, to avoid congestion collapse. A typical congestion avoidance scheme acts by senders reducing the amount of traffic entering the network upon an indication that network congestion is occurring (or about to occur). Unless there is an explicit indication, packet loss or timer expiration is normally regarded as an implicit indication of network congestion.

Traffic policing determines whether the traffic being presented is, on a hop-by-hop basis, compliant with prenegotiated policies or contracts. Nonconformant packets may be dropped, delayed, or labeled as nonconformant. As an example, ITU-T Recommendation Y.1221, *Traffic Control and Congestion Control in IP-Based Networks*, recommends the use of token bucket to characterize traffic for purposes of traffic policing.

Queuing and scheduling algorithms, also referred to as queuing discipline algorithms, determine which packet to send next and are used primarily to manage the allocation of transmission capacity among flows. Queuing discipline is discussed in Section 9.3.

Queue management algorithms manage the length of packet queues by dropping packets when necessary or appropriate. Active management of queues is concerned primarily with congestion avoidance. In the early days of the Internet, the queue management discipline was to drop any incoming packets when the queue was full, referred to as the **tail drop** technique. As pointed out in RFC 2309, *Recommendations on Queue Management and Congestion Avoidance in the Internet*, there are a number of drawbacks to tail drop, including the following:

1. There is no reaction to congestion until it is necessary to drop packets, whereas a more aggressive congestion avoidance technique would likely improve overall network performance.
2. Queues tend to be close to full, which causes an increase in packet delay through a network and which can result in a large batch of drop packets for bursty traffic, necessitating many packet retransmissions.
3. Tail drop may allow a single connection or a few flows to monopolize queue space, preventing other connections from getting room in the queue.

One noteworthy example of queue management is random early detection (RED), defined in RFC 2309. RED drops incoming packets probabilistically based on an estimated average queue size. The probability for dropping increases as the estimated average queue size grows. There are a number of variants of RED that are in more common use than the original RED, with weighted RED (WRED) perhaps the most commonly implemented. WRED prevents network congestion

by detecting and slowing flows (according to service class) before congestion occurs. WRED drops selected packets, which alerts the TCP sender to reduce its transmission rate. Weights are assigned to service classes, resulting in low priority flows being slowed more aggressively than high priority ones.

Control Plane

The control plane is concerned with creating and managing the pathways through which user data flows. It includes admission control, QoS routing, and resource reservation.

Admission control determines what user traffic may enter the network. This may be in part determined by the QoS requirements of a data flow compared to the current resource commitment within the network. But beyond balancing QoS requests with available capacity to determine whether to accept a request, there are other considerations in admission control. Network managers and service providers must be able to monitor, control, and enforce use of network resources and services based on policies derived from criteria such as the identity of users and applications, traffic/bandwidth requirements, security considerations, and time of day/week. RFC 2753, *A Framework for Policy-Based Admission Control*, discusses such policy-related issues.

QoS routing determines a network path that is likely to accommodate the requested QoS of a flow. This contrasts with the philosophy of the traditional routing protocols, which generally are looking for a least-cost path through the network. RFC 2386, *A Framework for QoS-Based Routing in the Internet*, provides an overview of the issues involved in QoS routing. This is an area of ongoing study. An example of a current implementation is Cisco's Performance Routing (PfR). PfR monitors network performance and selects the best path for each application based upon advanced criteria such as reachability, delay, jitter, and loss. PfR can evenly distribute traffic to maintain equivalent link utilization levels using an advanced load-balancing technique.

Resource reservation is a mechanism that reserves network resources on demand for delivering desired network performance to a requesting flow. An example of a protocol that uses this capability is the Resource Reservation Protocol (RSVP). However, this approach has been found to not scale well and is rarely used today.

Management Plane

The management plane contains mechanisms that affect both control plane and data plane mechanisms. The control plane deals with the operation, administration, and management aspects of the network. It includes SLAs, traffic restoration, traffic metering and recording, and policy.

A **service level agreement (SLA)** typically represents the agreement between a customer and a provider of a service that specifies the level of availability, serviceability, performance, operation, or other attributes of the service. SLAs are discussed in Section 10.5.

Traffic metering and recording concerns monitoring the dynamic properties of a traffic stream using performance metrics such as data rate and packet loss rate. It involves observing traffic characteristics at a given network point and collecting and storing the traffic information for

analysis and further action. Depending on the conformance level, a meter can invoke necessary treatment (for example, dropping or shaping) for the packet stream. Section 10.6 discusses the types of metrics that are used in this function.

Traffic restoration refers to the network response to failures. This encompasses a number of protocol layers and techniques.

Policy is a category that refers to a set of rules for administering, managing, and controlling access to network resources. They can be specific to the needs of the service provider or reflect the agreement between the customer and service provider, which may include reliability and availability requirements over a period of time and other QoS requirements.

10.3 Integrated Services Architecture

To define the requirements for QoS-based service, the IETF developed a suite of standards under the general umbrella of the Integrated Services Architecture (ISA). ISA, intended to provide QoS transport over IP-based internets, is defined in overall terms in RFC 1633, while a number of other documents fill in the details. ISA as such is not implemented in any current products. However, the architectural principles are in wide use, and ISA provides a convenient structure for discussing a number of QoS mechanisms.

ISA Approach

The purpose of ISA is to enable the provision of QoS support over IP-based internets. The central design issue for ISA is how to share the available capacity in times of congestion.

For an IP-based Internet that provides only a best effort service, the tools for controlling congestion and providing service are limited. In essence, routers have two mechanisms to work with:

- **Routing algorithm**: Some routing protocols in use in internets allow routes to be selected to minimize delay. Routers exchange information to get a picture of the delays throughout the Internet. Minimum-delay routing helps to balance loads, thus decreasing local congestion, and helps to reduce delays seen by individual TCP connections. Interface data rate may also be used as a metric.
- **Packet discard**: When a router's buffer overflows, it discards packets. Typically, the most recent packet is discarded. The effect of lost packets on a TCP connection is that the sending TCP entity backs off and reduces its load, thus helping to alleviate Internet congestion.

These tools have worked reasonably well. However, as the discussion in Section 2.1 (Types of Network and Internet Traffic) shows, such techniques are inadequate for the variety of traffic now coming to internets.

In ISA, each IP packet can be associated with a flow. A flow is a distinguishable stream of related IP packets that results from a single user activity and requires the same QoS. For example, a flow might consist of traffic on one transport connection in one direction or one video stream distinguishable by the ISA. A flow differs from a TCP connection in two respects: A flow

is unidirectional, and there can be more than one recipient of a flow (multicast). Typically, an IP packet is identified as a member of a flow on the basis of source and destination IP addresses and port numbers, and protocol type. The flow identifier in the IPv6 header is not necessarily equivalent to an ISA flow, but in future the IPv6 flow identifier could be used in ISA.

ISA makes use of the following functions to manage congestion and provide QoS transport:

- **Admission control**: For QoS transport (other than default best effort transport), ISA requires that a reservation be made for a new flow. If the routers collectively determine that there are insufficient resources to guarantee the requested QoS, the flow is not admitted. The protocol RSVP is used to make reservations.
- **Routing algorithm**: The routing decision may be based on a variety of QoS parameters, not just minimum delay.
- **Queuing discipline**: A vital element of the ISA is an effective queuing policy that takes into account the differing requirements of different flows.
- **Discard policy**: A discard policy determines which packets to drop when a buffer is full and new packets arrive. A discard policy can be an important element in managing congestion and meeting QoS guarantees.

ISA Components

Figure 10.2 is a general depiction of the implementation architecture for ISA within a router. Below the thick horizontal line are the forwarding functions of the router; these are executed for each packet and therefore must be highly optimized. The remaining functions, above the line, are background functions that create data structures used by the forwarding functions. Thus, the lower portion of Figure 10.2 corresponds roughly to the data plane of Figure 10.1, and the upper portion corresponds to the control plane.

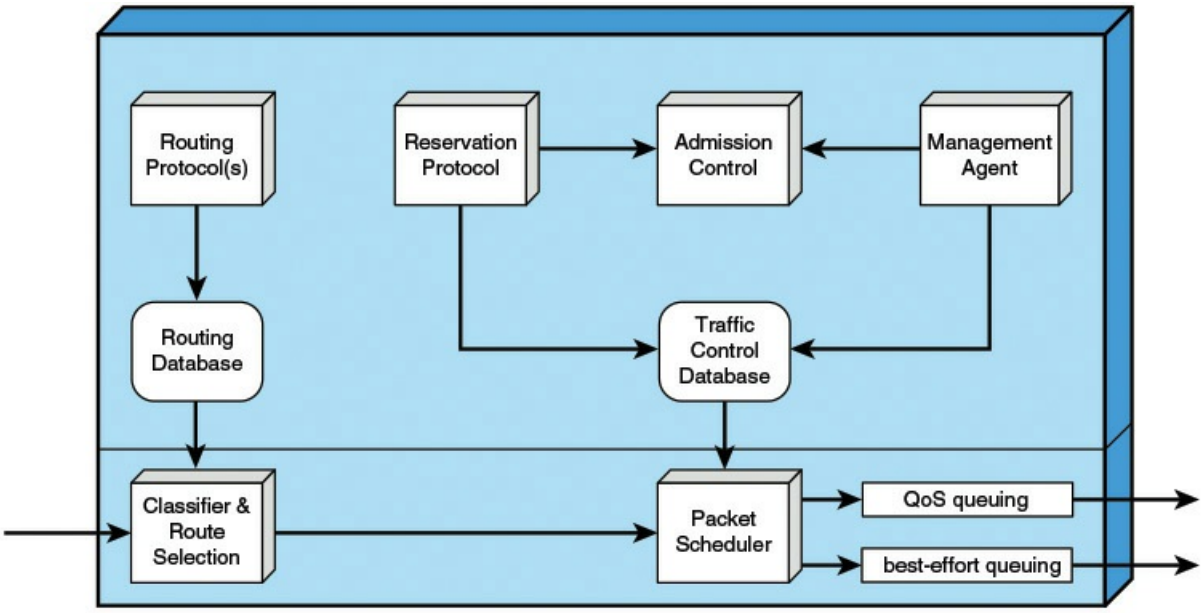


FIGURE 10.2 Integrated Services Architecture Implemented in Router

The principal background functions are as follows:

- **Reservation protocol**: This protocol reserves resources for a new flow at a given level of QoS. It is used among routers and between routers and end systems. The reservation protocol is responsible for maintaining flow-specific state information at the end systems and at the routers along the path of the flow. RSVP is used for this purpose. The reservation protocol updates the traffic control database used by the packet scheduler to determine the service provided for packets of each flow.
- **Admission control**: When a new flow is requested, the reservation protocol invokes the admission control function. This function determines if sufficient resources are available for this flow at the requested QoS. This determination is based on the current level of commitment to other reservations or on the current load on the network.
- **Management agent**: A network management agent can modify the traffic control database and to direct the admission control module to set admission control policies.
- **Routing protocol**: The routing protocol is responsible for maintaining a routing database that gives the next hop to be taken for each destination address and each flow.

These background functions support the main task of the router, which is the forwarding of packets. The two principal functional areas that accomplish forwarding are the following:

- **Classifier and route selection**: For the purposes of forwarding and traffic control, incoming packets must be mapped into classes. A class may correspond to a single flow or to a set of flows with the same QoS requirements. For example, the packets of all video flows or the packets of all flows attributable to a particular organization may be treated identically for purposes of resource allocation and queuing discipline. The selection of class is based on fields in the IP header. Based on the packet's class and its destination IP address, this function determines the next-hop address for this packet.
- **Packet scheduler**: This function manages one or more queues for each output port. It determines the order in which queued packets are transmitted and the selection of packets for discard, if necessary. Decisions are made based on a packet's class, the contents of the traffic control database, and current and past activity on this outgoing port. Part of the packet scheduler's task is that of policing, which is the function of determining whether the packet traffic in a given flow exceeds the requested capacity and, if so, deciding how to treat the excess packets.

ISA Services

ISA service for a flow of packets is defined on two levels. First, a number of general categories of service are provided, each of which provides a certain general type of service guarantees. Second, within each category, the service for a particular flow is specified by the values of certain parameters; together, these values are referred to as a traffic specification (TSpec). Three categories of service are defined:

- Guaranteed
- Controlled load

- Best effort

An application can request a reservation for a flow for a guaranteed or controlled load QoS, with a TSpec that defines the exact amount of service required. If the reservation is accepted, the TSpec is part of the contract between the data flow and the service. The service agrees to provide the requested QoS as long as the flow's data traffic continues to be described accurately by the TSpec. Packets that are not part of a reserved flow are by default given a best effort delivery service.

Guaranteed Service

The key elements of the guaranteed service are as follows:

- The service provides assured capacity, or data rate.
- There is a specified upper bound on the queuing delay through the network. This must be added to the propagation delay, or latency, to arrive at the bound on total delay through the network.
- There are no queuing losses. That is, no packets are lost because of buffer overflow; packets may be lost because of failures in the network or changes in routing paths.

With this service, an application provides a characterization of its expected traffic profile, and the service determines the end-to-end delay that it can guarantee.

One category of applications for this service is those that need an upper bound on delay so that a delay buffer can be used for real-time playback of incoming data, and that do not tolerate packet losses because of the degradation in the quality of the output. Another example is applications with hard real-time deadlines.

The guaranteed service is the most demanding service provided by ISA. Because the delay bound is firm, the delay has to be set at a large value to cover rare cases of long queuing delays.

Controlled Load

The key elements of the controlled load service are as follows:

- The service tightly approximates the behavior visible to applications receiving best effort service under unloaded conditions.
- There is no specified upper bound on the queuing delay through the network. However, the service ensures that a very high percentage of the packets do not experience delays that greatly exceed the minimum transit delay (that is, the delay due to propagation time plus router processing time with no queuing delays).
- A very high percentage of transmitted packets will be successfully delivered (that is, almost no queuing loss).

As was mentioned, the risk in an internet that provides QoS for real-time applications is that best effort traffic is crowded out. This is because best effort types of applications are assigned a low priority and their traffic is throttled in the face of congestion and delays. The controlled load service guarantees that the network will set aside sufficient resources so that an application that

receives this service will see a network that responds as if these real-time applications were not present and competing for resources.

The controlled service is useful for applications that have been referred to as adaptive real-time applications. Such applications do not require an *a priori* upper bound on the delay through the network. Rather, the receiver measures the jitter experienced by incoming packets and sets the playback point to the minimum delay that still produces a sufficiently low loss rate. (For example, video can be adaptive by dropping a frame or delaying the output stream slightly; voice can be adaptive by adjusting silent periods.)

Queuing Discipline

An important component of an ISA implementation is the queuing discipline used at the routers. The simplest approach that can be used by a router is a first-in, first-out (FIFO) queuing discipline at each output port. A single queue is maintained at each output port. When a new packet arrives and is routed to an output port, it is placed at the end of the queue. As long as the queue is not empty, the router transmits packets from the queue, taking the oldest remaining packet next.

There are several drawbacks to the FIFO queuing discipline:

- No special treatment is given to packets from flows that are of higher priority or are more delay sensitive. If a number of packets from different flows are ready to be forwarded, they are handled strictly in FIFO order.
- If a number of smaller packets are queued behind a long packet, FIFO queuing results in a larger average delay per packet than if the shorter packets were transmitted before the longer packet. In general, flows of larger packets get better service.
- A selfish TCP connection, which ignores the TCP congestion control rules, can crowd out conforming connections. If congestion occurs and one TCP connection fails to back off, other connections along the same path segment must back off more than they would otherwise have to do.

To overcome the drawbacks of FIFO queuing, a number of more complex routing algorithms have been implemented in routers. These algorithms involve the use of multiple queues at each output port and some method of prioritizing the traffic to provide better service. Typical of the networking industry are the routers from Cisco which, in addition to FIFO, include the following queuing approaches outlined in the Cisco *Internetworking Technology Handbook* [CISC15]:

- Priority queuing (PQ)
- Custom queuing (CQ)
- Flow-based weighted fair queuing (WFQ)
- Class-based weighted fair queuing (CBWFQ)

For **priority queuing**, each packet is assigned a priority level, and there is one queue for each priority level. In the Cisco implementation, four levels are used: high, medium, normal, and low. Packets not otherwise classified are assigned to the normal priority. PQ can flexibly prioritize according to network protocol, incoming interface, packet size, source/destination address, or

other parameters. The queuing discipline gives absolute preference based on priority. Thus, if there are packets waiting in multiple queues, the router dispatches packets on a FIFO basis from the highest-priority queue that is not empty. Only after that queue is empty are packets dispatched from the next lower priority queue. When new packets arrive in a higher priority queue, they immediately take precedence over any packets already waiting in lower priority queues. PQ is useful for assuring that mission-critical application traffic is handled as well as possible, but it risks crowding out lower priority traffic for very long periods of time.

Custom queuing is designed to allow various applications or organizations to share the network among applications with specific minimum throughput or latency requirements. For CQ, there are multiple queues, with each having a configured byte count. The queues are serviced in round-robin fashion. As each queue is visited, a number of packets are dispatched up to the configured byte count. By providing different byte counts for different queues, traffic on each queue is guaranteed a minimum fraction of the overall capacity. Application or protocol traffic can then be assigned to the desired queue.

The remaining queuing algorithms on the preceding list are based on a mechanism known as fair queuing. With simple fair queuing, each incoming packet is placed in the queue for its flow. The queues are serviced in round-robin fashion, taking one packet from each nonempty queue in turn. Empty queues are skipped over. This scheme is fair in that each busy flow gets to send exactly one packet per cycle. Further, this is a form of load balancing among the various flows. There is no advantage in being greedy. A greedy flow finds that its queues become long, increasing its delays, whereas other flows are unaffected by this behavior.

The term *weighted fair queuing* (WFQ) is used in the literature to refer to a class of scheduling algorithms that use multiple queues to support capacity allocation and delay bounds. Some WFQ schemes take into account the amount of traffic through each queue and gives busier queues more capacity without completely shutting out less busy queues. WFQ may also take into account the amount of service requested by each traffic flow and adjust the queuing discipline accordingly.

Flow-based WFQ, which Cisco simply refers to as WFQ, creates flows based on a number of characteristics in a packet, including source and destination addresses, socket numbers, and session identifiers. The flows are assigned different weights to based on IP precedent bits to provide greater service for certain queues.

Class-based WFQ (CBWFQ) allows a network administrator to create minimum guaranteed bandwidth classes. Instead of providing a queue for each individual flow, a class is defined that consists of one or more flows. Each class can be guaranteed a minimum amount of bandwidth.

10.4 Differentiated Services

The differentiated services (DiffServ) architecture (RFC 2475) is designed to provide a simple, easy-to-implement, low-overhead tool to support a range of network services that are differentiated on the basis of performance.

Several key characteristics of DiffServ contribute to its efficiency and ease of deployment:

- IP packets are labeled for differing QoS treatment using the existing IPv4 or IPv6 DSField. Thus, no change is required to IP.

- A service level specification (SLS) is established between the service provider (Internet domain) and the customer prior to the use of DiffServ. This avoids the need to incorporate DiffServ mechanisms in applications. Therefore, existing applications need not be modified to use DiffServ. The SLS is a set of parameters and their values that together define the service offered to a traffic stream by a DiffServ domain.
- A traffic conditioning specification (TCS) is a part of the SLS that specifies traffic classifier rules and any corresponding traffic profiles and metering, marking, discarding/shaping rules which are to apply to the traffic stream.
- DiffServ provides a built-in aggregation mechanism. All traffic with the same DiffServ octet is treated the same by the network service. For example, multiple voice connections are not handled individually but in the aggregate. This provides for good scaling to larger networks and traffic loads.
- DiffServ is implemented in individual routers by queuing and forwarding packets based on the DiffServ octet. Routers deal with each packet individually and do not have to save state information on packet flows.

Today, DiffServ is the most widely accepted QoS mechanism in enterprise networks.

Although DiffServ is intended to provide a simple service based on relatively simple mechanisms, the set of RFCs related to DiffServ is relatively complex. Table 10.1 summarizes some of the key terms from these specifications.

Term	Definition
Behavior aggregate	A set of packets with the same DiffServ codepoint crossing a link in a particular direction.
Classifier	Selects packets based on the DSField (BA classifier) or on multiple fields within the packet header (MF classifier).
DiffServ boundary node	A DiffServ node that connects one DiffServ domain to a node in another domain
DSField	The 6 most significant bits of the (former) IPv4 TOS octet or the (former) IPv6 Traffic Class octet.
DiffServ codepoint	A value that is encoded in the DSField.
DiffServ domain	A contiguous (connected) set of nodes, capable of implementing differentiated services, that operate with a common set of service provisioning policies and per-hop behavior definitions.
DiffServ interior node	A DiffServ node that is not a DiffServ boundary node.
DiffServ node	A node that supports differentiated services. Typically, a DiffServ node is a router. A host system that provides differentiated services for applications in the host is also a DiffServ node.
Dropping	The process of discarding packets based on specified rules; also called policing.

Marking	The process of setting the DiffServ codepoint in a packet. Packets may be marked on initiation and may be re-marked by an en route DiffServ node.
Metering	The process of measuring the temporal properties (for example, rate) of a packet stream selected by a classifier. The instantaneous state of that process may affect marking, shaping, and dropping functions.
Per-hop behavior (PHB)	The externally observable forwarding behavior applied at a node to a behavior aggregate.
Service level agreement (SLA)	A service contract between a customer and a service provider that specifies the forwarding service a customer should receive.
Shaping	The process of delaying packets within a packet stream to cause it to conform to some defined traffic profile.
Traffic conditioning	Control functions performed to enforce rules specified in a TCA, including metering, marking, shaping, and dropping.
Traffic conditioning agreement (TCA)	An agreement specifying classifying rules and traffic conditioning rules that are to apply to packets selected by the classifier.

TABLE 10.1 Terminology for Differentiated Services

Services

The DiffServ type of service is provided within a DiffServ domain, which is defined as a contiguous portion of the Internet over which a consistent set of DiffServ policies are administered. Typically, a DiffServ domain would be under the control of one administrative entity. The services provided across a DiffServ domain are defined in an SLA, which is a service contract between a customer and the service provider that specifies the forwarding service that the customer should receive for various classes of packets. A customer may be a user organization or another DiffServ domain. Once the SLA is established, the customer submits packets with the DiffServ octet marked to indicate the packet class. The service provider must ensure that the customer gets at least the agreed QoS for each packet class. To provide that QoS, the service provider must configure the appropriate forwarding policies at each router (based on DiffServ octet value) and must measure the performance being provided for each class on an ongoing basis.

If a customer submits packets intended for destinations within the DiffServ domain, the DiffServ domain is expected to provide the agreed service. If the destination is beyond the customer's DiffServ domain, the DiffServ domain will attempt to forward the packets through other domains, requesting the most appropriate service to match the requested service.

A DiffServ framework document lists the following detailed performance parameters that might be included in an SLA:

- Detailed service performance parameters such as expected throughput, drop probability, and latency.
- Constraints on the ingress and egress points at which the service is provided, indicating

the scope of the service.

- Traffic profiles that must be adhered to for the requested service to be provided, such as token bucket parameters.
- Disposition of traffic submitted in excess of the specified profile.

The framework document also gives some examples of services that might be provided:

- Traffic offered at service level A will be delivered with low latency.
- Traffic offered at service level B will be delivered with low loss.
- 90 percent of in-profile traffic delivered at service level C will experience no more than 50 ms latency.
- 95 percent of in-profile traffic delivered at service level D will be delivered.
- Traffic offered at service level E will be allotted twice the bandwidth of traffic delivered at service level F.
- Traffic with drop precedence X has a higher probability of delivery than traffic with drop precedence Y.

The first two examples are qualitative and are valid only in comparison to other traffic, such as default traffic that gets a best effort service. The next two examples are quantitative and provide a specific guarantee that can be verified by measurement on the actual service without comparison to any other services offered at the same time. The final two examples are a mixture of quantitative and qualitative.

DiffServ Field

Packets are labeled for service handling by means of the 6-bit DSField in the IPv4 header or the IPv6 header (Figure 10.3). The value of the DSField, referred to as the **DiffServ codepoint (DSCP),** is the label used to classify packets for differentiated services.

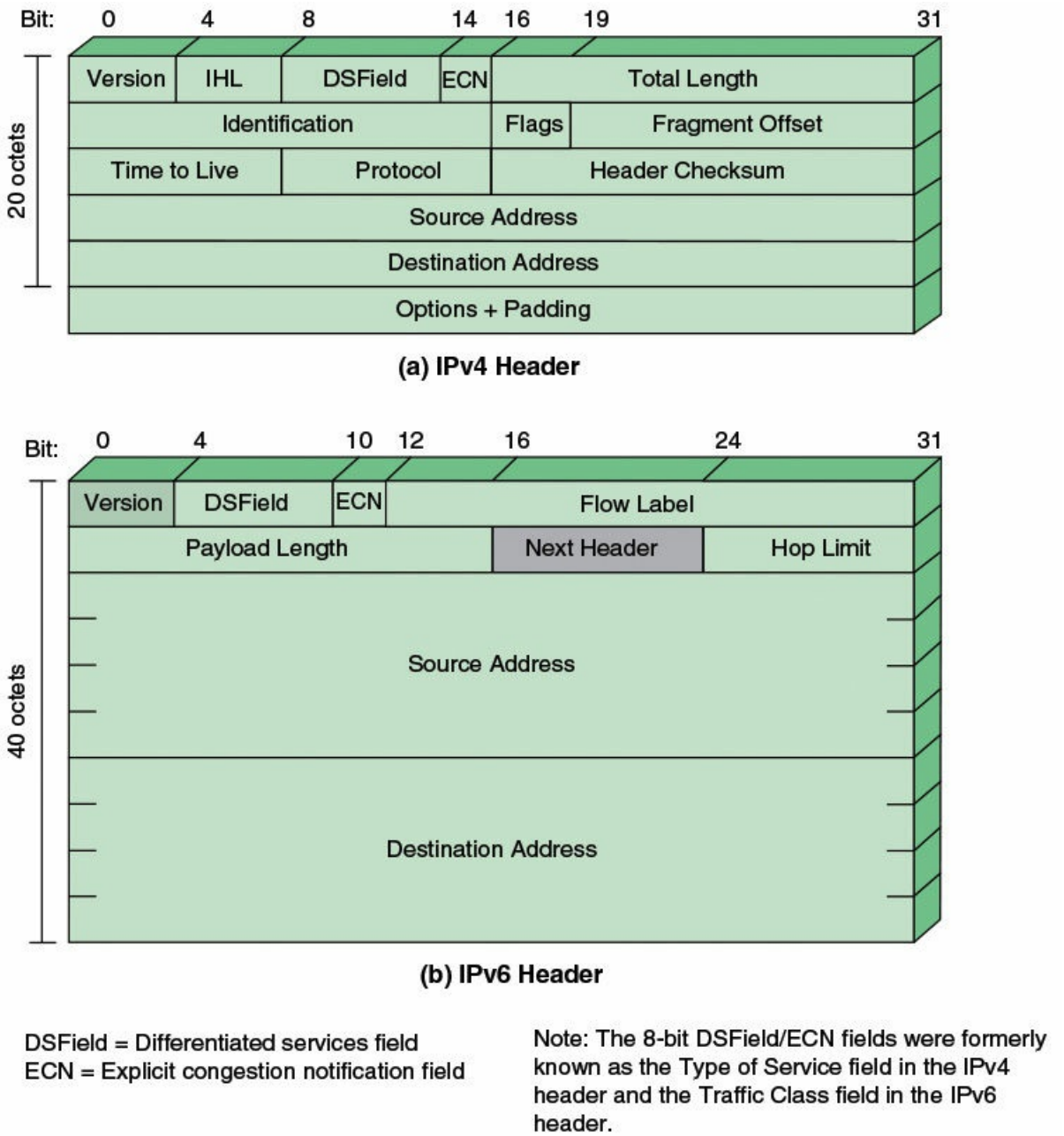


FIGURE 10.3 IP Headers

With a 6-bit codepoint, there are in principle 64 different classes of traffic that could be defined. These 64 codepoints are allocated across three pools of codepoints, as follows:

- Codepoints of the form xxxxx0, where x is either 0 or 1, are reserved for assignment as standards.
- Codepoints of the form xxxx11 are reserved for experimental or local use.
- Codepoints of the form xxxx01 are also reserved for experimental or local use but may be allocated for future standards action as needed.

DiffServ Configuration and Operation

Figure 10.4 illustrates the type of configuration envisioned in the DiffServ documents. A DiffServ domain consists of a set of contiguous routers; that is, it is possible to get from any router in the domain to any other router in the domain by a path that does not include routers outside the domain. Within a domain, the interpretation of DS codepoints is uniform, so that a uniform, consistent service is provided.

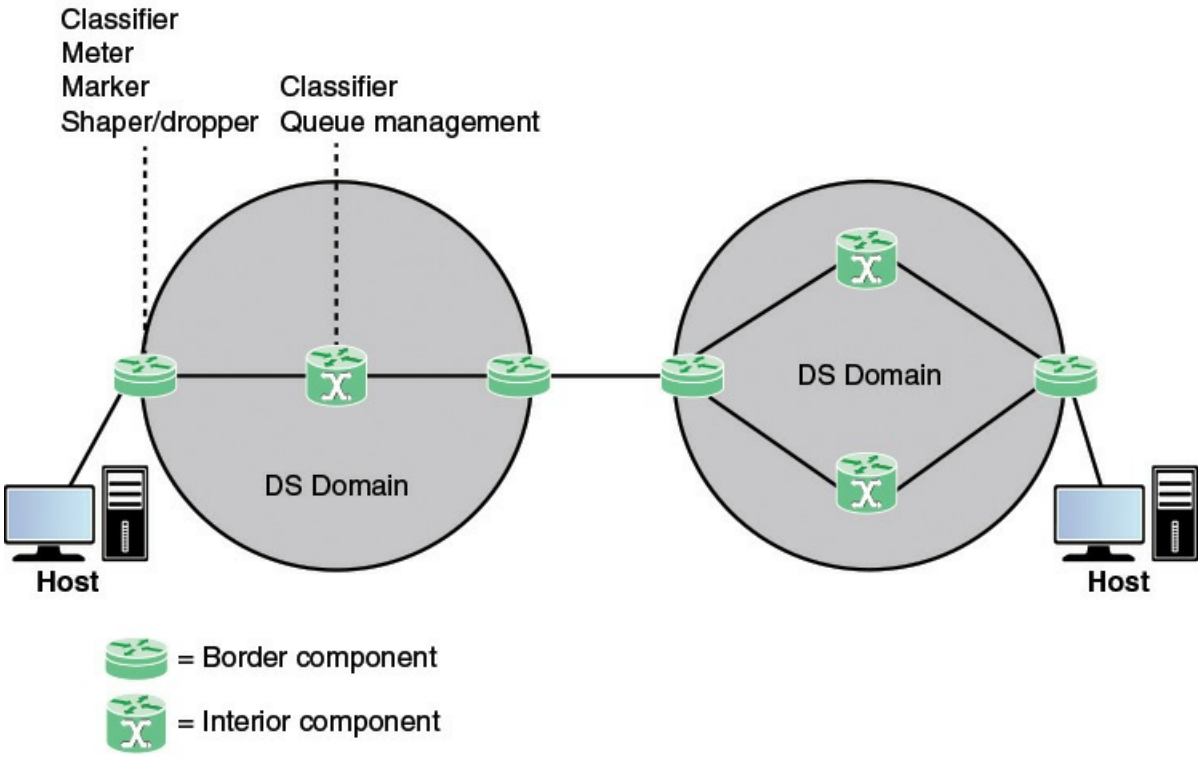


FIGURE 10.4 DS Domains

Routers in a DiffServ domain are either boundary nodes or interior nodes. Typically, the interior nodes implement simple mechanisms for handling packets based on their DS codepoint values. This includes queuing discipline to give preferential treatment depending on codepoint value, and packet dropping rules to dictate which packets should be dropped first in the event of buffer saturation. The DiffServ specifications refer to the forwarding treatment provided at a router as per-hop behavior (PHB). This PHB must be available at all routers, and typically PHB is the only part of DiffServ implemented in interior routers.

The boundary nodes include PHB mechanisms but more sophisticated traffic conditioning mechanisms are also required to provide the desired service. Therefore, interior routers have minimal functionality and minimal overhead in providing the DiffServ service; most of the complexity is in the boundary nodes. The boundary node function can also be provided by a host system attached to the domain, on behalf of the applications at that host system.

The traffic conditioning function consists of five elements:

- **Classifier**: Separates submitted packets into different classes. This is the foundation of providing differentiated services. A classifier may separate traffic only on the basis of the

DS codepoint (behavior aggregate classifier) or based on multiple fields within the packet header or even the packet payload (multifield classifier).

- **Meter**: Measures submitted traffic for conformance to a profile. The meter determines whether a given packet stream class is within or exceeds the service level guaranteed for that class.
- **Marker**: Re-marks packets with a different codepoint as needed. This may be done for packets that exceed the profile; for example, if a given throughput is guaranteed for a particular service class, any packets in that class that exceed the throughput in some defined time interval may be re-marked for best effort handling. Also, re-marking may be required at the boundary between two DiffServ domains. For example, if a given traffic class is to receive the highest supported priority, and this is a value of 3 in one domain and 7 in the next domain, packets with a priority 3 value traversing the first domain are re-marked as priority 7 when entering the second domain.
- **Shaper**: Delays packets as necessary so that the packet stream in a given class does not exceed the traffic rate specified in the profile for that class.
- **Dropper**: Drops packets when the rate of packets of a given class exceeds that specified in the profile for that class.

Figure 10.5 illustrates the relationship between the elements of traffic conditioning. After a flow is classified, its resource consumption must be measured. The metering function measures the volume of packets over a particular time interval to determine a flow's compliance with the traffic agreement. If the host is bursty, a simple data rate or packet rate may not be sufficient to capture the desired traffic characteristics. A **token bucket** scheme is an example of a way to define a traffic profile to take into account both packet rate and burstiness.

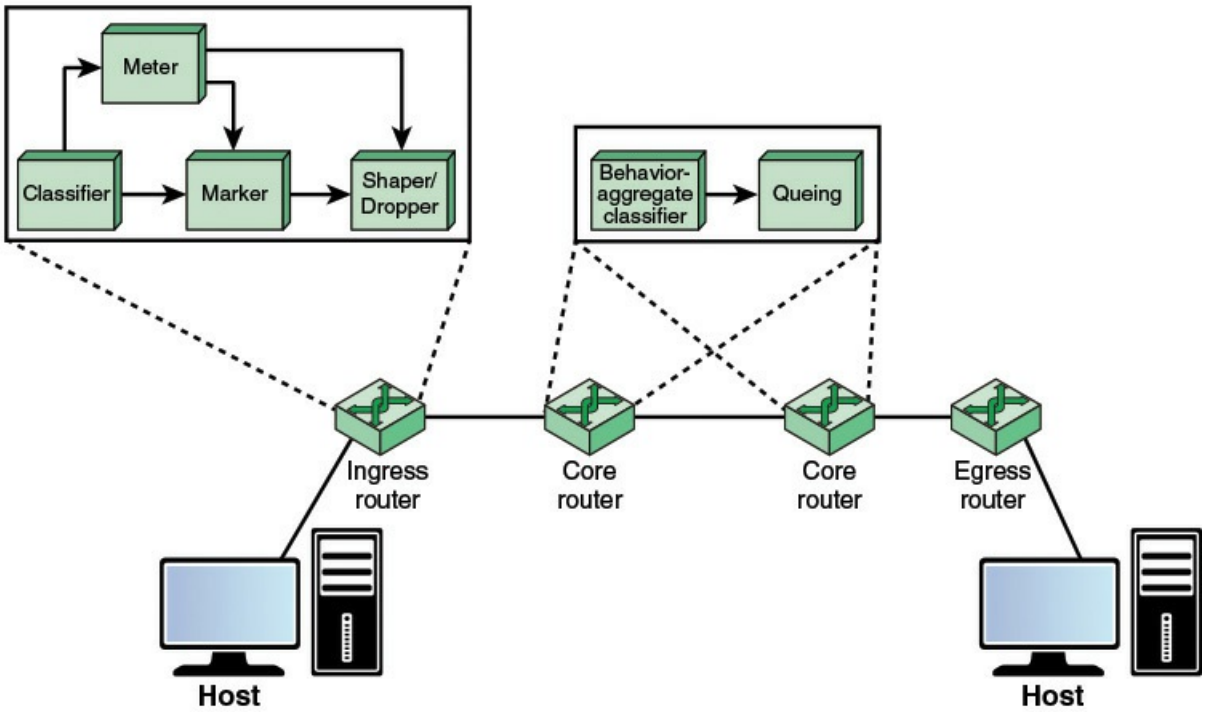


FIGURE 10.5 DS Functions

If a traffic flow exceeds some profile, several approaches can be taken. Individual packets in excess of the profile may be re-marked for lower-quality handling and allowed to pass into the DiffServ domain. A traffic shaper may absorb a burst of packets in a buffer and pace the packets over a longer period. A dropper may drop packets if the buffer used for pacing becomes saturated.

Per-Hop Behavior

DiffServ is a general architecture that can be used to implement a variety of services. As part of the DS standardization effort, specific types of PHB need to be defined, which can be associated with specific differentiated services. Three fundamental forwarding behaviors have been defined and characterized for general use, plus a "legacy" forwarding behavior class has been defined. The four behavior classes are as follows:

- Default forwarding (DF) for elastic traffic
- Assured forwarding (AF) for general QoS requirements
- Expedited forwarding (EF) for real-time (inelastic) traffic
- Class selector for historical codepoint definitions and PHB requirements

Figure 10.6 shows the DSCP encodings corresponding to the four classes. The remainder of this section discusses each class in turn.

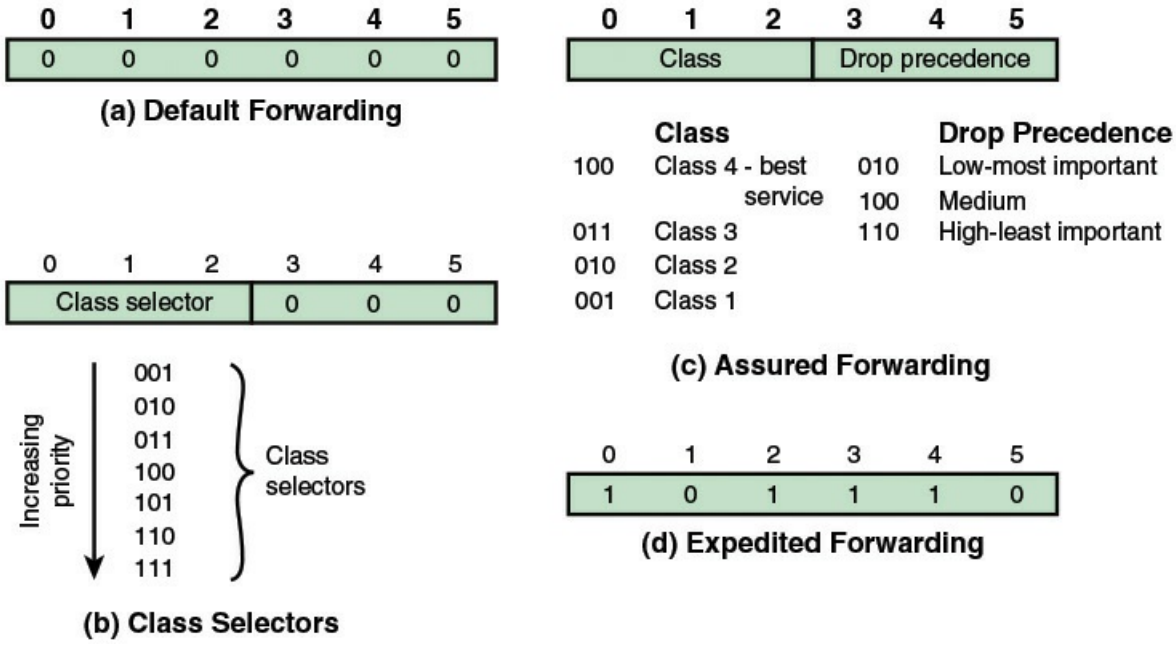


FIGURE 10.6 DiffServ Forwarding Behavior Classes and Corresponding DSField Encoding

Default Forwarding PHB

The default class, referred to as default forwarding (DF), is the best effort forwarding behavior in existing routers. Such packets are forwarded in the order that they are received as soon as link capacity becomes available. If other higher-priority packets in other DiffServ classes are available for transmission, the latter are given preference over best effort default packets. Application traffic in the Internet that uses default forwarding is expected to be elastic in nature. The sender of traffic is expected to adjust its transmission rate in response to changes in available rate, loss, or delay.

Expedited Forwarding PHB

RFC 3246 defines the expedited forwarding (EF) PHB as a building block for low-loss, low-delay, and low-jitter end-to-end services through DiffServ domains. In essence, such a service should appear to the endpoints as providing close to the performance of a point-to-point connection or leased line.

In an internet or packet-switching network, a low-loss, low-delay, and low-jitter service is difficult to achieve. By its nature, an internet involves queues at each node, or router, where packets are buffered waiting to use a shared output link. It is the queuing behavior at each node that results in loss, delays, and jitter. Therefore, unless the internet is grossly oversized to eliminate all queuing effects, care must be taken in handling traffic for EF PHB to ensure that queuing effects do not result in loss, delay, or jitter above a given threshold. RFC 3246 declares that the intent of the EF PHB is to provide a PHB in which suitably marked packets usually encounter short or empty queues. The relative absence of queuing effects minimizes delay and jitter. Furthermore, if queues remain short relative to the buffer space available, packet loss is also kept to a minimum.

The EF PHB is designed to configure nodes so that the traffic aggregate[2] has a well-defined minimum departure rate. (*Well-defined* means “independent of the dynamic state of the node,” in particular, independent of the intensity of other traffic at the node.) The general concept outlined in RFC 3246 is this: The border nodes control the traffic aggregate to limit its characteristics (rate, burstiness) to some predefined level. Interior nodes must treat the incoming traffic in such a way that queuing effects do not appear. In general terms, the requirement on interior nodes is that the aggregate’s maximum arrival rate must be less than the aggregate’s minimum departure rate.

2 The term *traffic aggregate* refers to the flow of packets associated with a particular service for a particular user.

RFC 3246 does not mandate a specific queuing policy at the interior nodes to achieve the EF PHB. The RFC notes that a simple priority scheme could achieve the desired effect, with the EF traffic given absolute priority over other traffic. So long as the EF traffic itself did not overwhelm an interior node, this scheme would result in acceptable queuing delays for the EF PHB. However, the risk of a simple priority scheme is that packet flows for other PHB traffic would be disrupted. Therefore, some more sophisticated queuing policy might be warranted.

Assured Forwarding PHB

The assured forwarding (AF) PHB is designed to provide a service superior to best effort but one that does not require the reservation of resources within an Internet and does not require the use

of detailed discrimination among flows from different users. The concept behind the AF PHB was first introduced in a paper by Clark and Fang [CLAR98] and is referred to as explicit allocation. The AF PHB is more complex than explicit allocation, but it is useful to first highlight the key elements of the explicit allocation scheme:

- Users are offered the choice of a number of classes of service for their traffic. Each class describes a different traffic profile in terms of an aggregate data rate and burstiness.
- Traffic from a user within a given class is monitored at a boundary node. Each packet in a traffic flow is marked out or in based on whether it does or does not exceed the traffic profile.
- Inside the network, there is no separation of traffic from different users or even traffic from different classes. Instead, all traffic is treated as a single pool of packets, with the only distinction being whether each packet has been marked in or out.
- When congestion occurs, the interior nodes implement a dropping scheme in which out packets are dropped before in packets.
- Different users will see different levels of service because they will have different quantities of in packets in the service queues.

The advantage of this approach is its simplicity. Very little work is required by the internal nodes. Marking of the traffic at the boundary nodes based on traffic profiles provides different levels of service to different classes.

The AF PHB defined in RFC 2597 expands on the preceding approach in the following ways:

- Four AF classes are defined, allowing the definition of four distinct traffic profiles. A user may select one or more of these classes to satisfy requirements.
- Within each class, packets are marked by the customer or by the service provider with one of three drop precedence values. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DiffServ node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value.

This approach is still simpler to implement than any sort of resource reservation scheme but provides considerable flexibility. Within an interior DiffServ node, traffic from the four classes can be treated separately, with different amounts of resources (buffer space, data rate) assigned to the four classes. Within each class, packets are handled based on drop precedence. Therefore, as RFC 2597 points out, the level of forwarding assurance of an IP packet depends on the following:

- How many forwarding resources have been allocated to the AF class to which the packet belongs.
- The current load of the AF class.
- In case of congestion within the class, the drop precedence of the packet.

RFC 2597 does not mandate any mechanisms at the interior nodes to manage the AF traffic. It does reference the RED algorithm as a possible way of managing congestion.

Part c of Figure 10.6 shows the recommended codepoints for AF PHB in the DSField.

Class Selector PHB

Codepoints of the form xxx000 are reserved to provide backward compatibility with the IPv4 precedence service. The IPv4 type of service (TOS) field, which has been replaced by the DSField and ECN field (Figure 10.3a), includes two subfields: a 3-bit precedence subfield and a 4-bit TOS subfield. These subfields serve complementary functions. The TOS subfield provides guidance to the IP entity (in the source or router) on selecting the next hop for this datagram, and the precedence subfield provides guidance about the relative allocation of router resources for this datagram.

The precedence field is set to indicate the degree of urgency or priority to be associated with a datagram. If a router supports the precedence subfield, there are three approaches to responding:

- **Route selection**: A particular route may be selected if the router has a smaller queue for that route or if the next hop on that route supports network precedence or priority (for example, a Token Ring network supports priority).
- **Network service**: If the network on the next hop supports precedence, that service is invoked.
- **Queuing discipline**: A router may use precedence to affect how queues are handled. For example, a router may give preferential treatment in queues to datagrams with higher precedence.

RFC 1812, *Requirements for IP Version 4 Routers*, provides recommendations for queuing discipline that fall into two categories:

- Queue service:

  Routers *should* implement precedence-ordered queue service. Precedence-ordered queue service means that when a packet is selected for output on a (logical) link, the packet of highest precedence that has been queued for that link is sent.

  Any router *may* implement other policy-based throughput management procedures that result in other than strict precedence ordering, but it *must* be configurable to suppress them (that is, use strict ordering).

- Congestion control. When a router receives a packet beyond its storage capacity, it must discard it or some other packet or packets:

  A router *may* discard the packet it has just received; this is the simplest but not the best policy.

  Ideally, the router should select a packet from one of the sessions most heavily abusing the link, given that the applicable QoS policy permits this. A recommended policy in datagram environments using FIFO queues is to discard a packet randomly selected from the queue. An equivalent algorithm in routers using fair queues is to discard from the longest queue. A router *may* use these algorithms to determine which packet to discard.

  If precedence-ordered queue service is implemented and enabled, the router *must not* discard a packet whose IP precedence is higher than that of a packet that is not discarded.

A router *may* protect packets whose IP headers request the maximize reliability TOS, except where doing so would be in violation of the previous rule.

A router *may* protect fragmented IP packets, on the theory that dropping a fragment of a datagram may increase congestion by causing all fragments of the datagram to be retransmitted by the source.

To help prevent routing perturbations or disruption of management functions, the router may protect packets used for routing control, link control, or network management from being discarded. Dedicated routers (that is, routers that are not also general purpose hosts, terminal servers, and so on) can achieve an approximation of this rule by protecting packets whose source or destination is the router itself.

The class selector PHB should provide a service that at minimum is equivalent to that of the IPv4 precedence functionality.

10.5 Service Level Agreements

A service level agreement (SLA) is a contract between a network provider and a customer that defines specific aspects of the service that is to be provided. The definition is formal and typically defines quantitative thresholds that must be met. An SLA typically includes the following information:

- **A description of the nature of service to be provided**: A basic service would be IP-based network connectivity of enterprise locations plus access to the Internet. The service may include additional functions such as web hosting, maintenance of domain name servers, and operation and maintenance tasks.
- **The expected performance level of the service**: The SLA defines a number of metrics, such as delay, reliability, and availability, with numerical thresholds.
- **The process for monitoring and reporting the service level:** This describes how performance levels are measured and reported.

Figure 10.7 shows a typical configuration that lends itself to an SLA. In this case, a network service provider maintains an IP-based network. A customer has a number of private networks (for example, LANs) at various sites. Customer networks are connected to the provider via access routers at the access points. The SLA dictates service and performance levels for traffic between access routers across the provider network. In addition, the provider network links to the Internet and thus provides Internet access for the enterprise. For example, for the standard SLA provided by Cogent Communications for its backbone networks includes the following items:

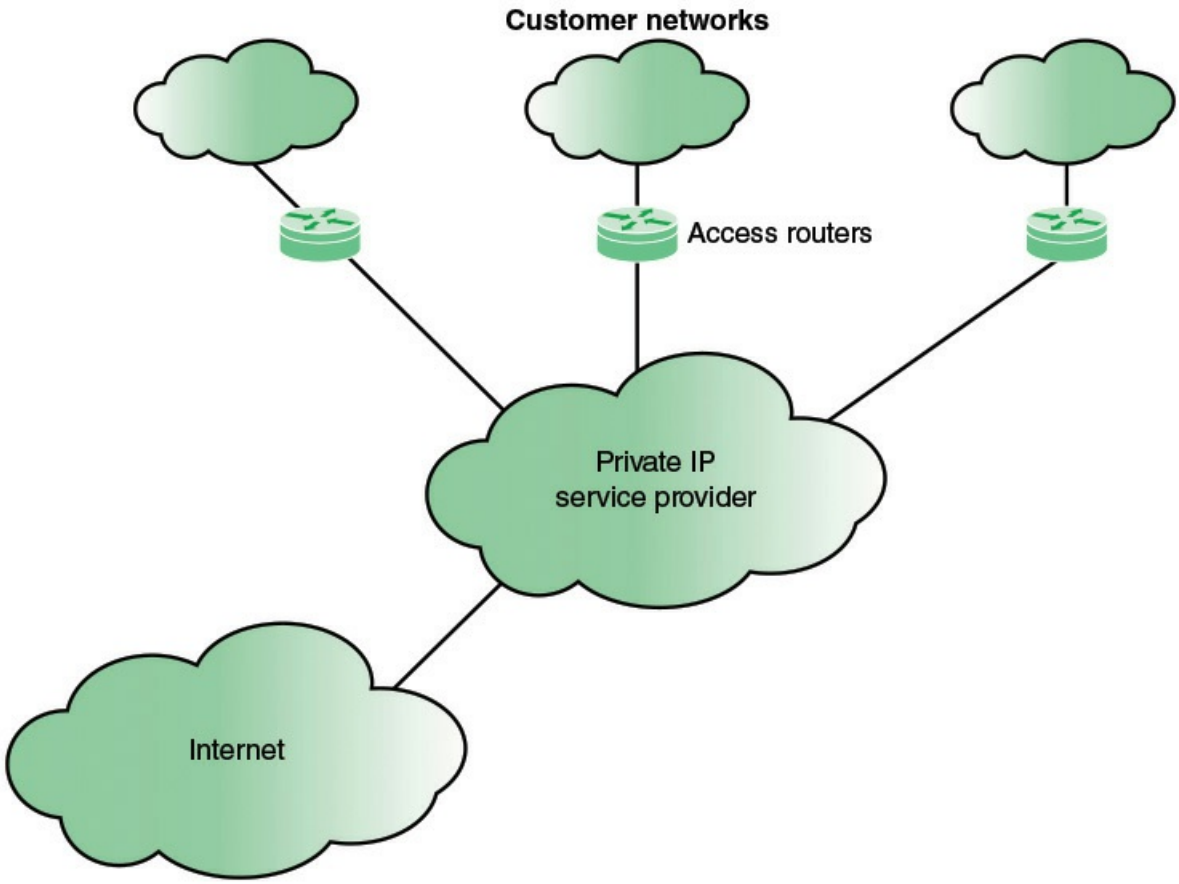


FIGURE 10.7 Typical Framework for Service Level Agreement

- **Availability**: 100 percent availability.
- **Latency (delay):** Monthly average network latency for packets carried over the Cogent Network between backbone hubs for the following regions is as specified here:

  Intra-North America: 45 milliseconds or less

  Intra-Europe: 35 milliseconds or less

  New York to London (transatlantic): 85 milliseconds or less

  Los Angeles to Tokyo (transpacific): 120 milliseconds or less

  Network latency (or round-trip time) is defined as the average time taken for an IP packet to make a round-trip between backbone hubs within the regions specified above on the Cogent Network. Cogent monitors aggregate latency within the Cogent Network by monitoring round-trip times between a sampling of backbone hubs on an ongoing basis.

- Network packet delivery (reliability): Average monthly packet loss no greater than 0.1 percent (or successful delivery of 99.9 percent of packets). Packet loss is defined as the percentage of packets that are dropped between backbone hubs on the Cogent Network.

An SLA can be defined for the overall network service. In addition, SLAs can be defined for specific end-to-end services available across the carrier's network, such as a virtual private network, or differentiated services.

10.6 IP Performance Metrics

The IP Performance Metrics Working Group (IPPM) is chartered by IETF to develop standard metrics that relate to the quality, performance, and reliability of Internet data delivery. Two trends dictate the need for such a standardized measurement scheme:

- The Internet has grown and continues to grow at a dramatic rate. Its topology is increasingly complex. As its capacity has grown, the load on the Internet has grown at an even faster rate. Similarly, private internets, such as corporate intranets and extranets, have exhibited similar growth in complexity, capacity, and load. The sheer scale of these networks makes it difficult to determine quality, performance, and reliability characteristics.
- The Internet serves a large and growing number of commercial and personal users across an expanding spectrum of applications. Similarly, private networks are growing in terms of user base and range of applications. Some of these applications are sensitive to particular QoS parameters, leading users to require accurate and understandable performance metrics.

A standardized and effective set of metrics enables users and service providers to have an accurate common understanding of the performance of the Internet and private internets. Measurement data is useful for a variety of purposes, including the following:

- Supporting capacity planning and troubleshooting of large complex internets.
- Encouraging competition by providing uniform comparison metrics across service providers.
- Supporting Internet research in such areas as protocol design, congestion control, and QoS.
- Verification of SLAs.

Table 10.2 lists the metrics that have been defined in RFCs at the time of this writing. Section a of Table 10.2 lists those metrics which result in a value estimated based on a sampling technique.

(a) Sampled Metrics		
Metric Name	**Singleton Definition**	**Statistical Definitions**
One-way delay	Delay = dT, where Src transmits first bit of packet at T and Dst received last bit of packet at T + dT	Percentile, median, minimum, inverse percentile
Round-trip delay	Delay = dT, where Src transmits first bit of packet at T and Src received last bit of packet immediately returned by Dst at T + dT	Percentile, median, minimum, inverse percentile
One-way loss	Packet loss = 0 (signifying successful transmission and reception of packet); = 1 (signifying packet loss)	Average
One-way loss pattern	Loss distance: Pattern showing the distance between successive packet losses in terms of the sequence of packets Loss period: Pattern showing the number of bursty losses (losses involving consecutive packets)	Number or rate of loss distances below a defined threshold, number of loss periods, pattern of period lengths, pattern of interloss period lengths
Packet delay variation	Packet delay variation (pdv) for a pair of packets with a stream of packets = difference between the one-way-delay of the selected packets	Percentile, inverse percentile, jitter, peak-to-peak pdv
(b) Other Metrics		
Metric	**General Definition**	**Metrics**
Connectivity	Ability to deliver a packet over a transport connection.	One-way instantaneous connectivity, two-way instantaneous connectivity, one-way interval connectivity, two-way interval connectivity, two-way temporal connectivity
Bulk transfer capacity	Long-term average data rate (bps) over a single congestion-aware transport connection.	BTC = (Data sent) / (Elapsed time)

Src = IP address of a host

Dst = IP address of a host

TABLE 10.2 IP Performance Metrics

These metrics are defined in three stages:

- **Singleton metric**: The most elementary, or atomic, quantity that can be measured for a given performance metric. For example, for a delay metric, a singleton metric is the delay experienced by a single packet.
- **Sample metric**: A collection of singleton measurements taken during a given time period. For example, for a delay metric, a sample metric is the set of delay values for all the

measurements taken during a one-hour period.

- **Statistical metric**: A value derived from a given sample metric by computing some statistic of the values defined by the singleton metric on the sample. For example, the mean of all the one-way delay values on a sample might be defined as a statistical metric.

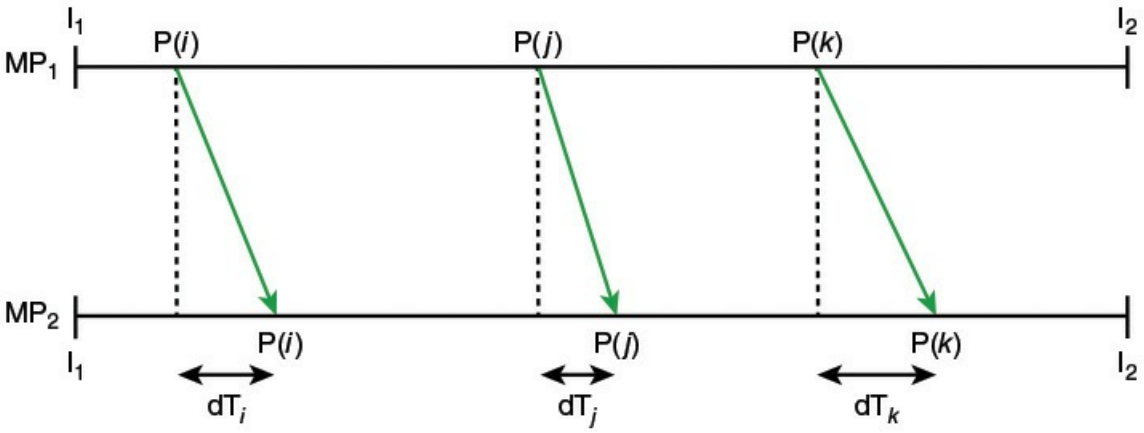
The measurement technique can be either active or passive. **Active techniques** require injecting packets into the network for the sole purpose of measurement. There are several drawbacks to this approach. The load on the network is increased. This, in turn, can affect the desired result. For example, on a heavily loaded network, the injection of measurement packets can increase network delay, so that the measured delay is greater than it would be without the measurement traffic. In addition, an active measurement policy can be abused for denial-of-service attacks disguised as legitimate measurement activity. **Passive techniques** observe and extract metrics from existing traffic. This approach can expose the contents of Internet traffic to unintended recipients, creating security and privacy concerns. So far, the metrics defined by the IPPM working group are all active.

For the sample metrics, the simplest technique is to take measurements at fixed time intervals, known as periodic sampling. There are several problems with this approach. First, if the traffic on the network exhibits periodic behavior, with a period that is an integer multiple of the sampling period (or vice versa), correlation effects may result in inaccurate values.

Also, the act of measurement can perturb what is being measured (for example, injecting measurement traffic into a network alters the congestion level of the network), and repeated periodic perturbations can drive a network into a state of synchronization, greatly magnifying what might individually be minor effects. Accordingly, RFC 2330, *Framework for IP Performance Metrics*, recommends Poisson sampling. This method uses a Poisson distribution to generate random time intervals with the desired mean value.

Most of the statistical metrics listed in part a of Table 10.2 are self-explanatory. The percentile metric is defined as follows: The xth percentile is a value y such that x% of measurements $\geq y$. The inverse percentile of x for a set of measurements is the percentage of all values $\leq x$.

Figure 10.8 illustrates the packet delay variation metric. This metric is used to measure jitter, or variability, in the delay of packets traversing the network. The singleton metric is defined by selecting two packet measurements and measuring the difference in the two delays. The statistical measures make use of the absolute values of the delays.



I_1, I_2 = times that mark that beginning and ending of the interval in which the packet stream from which the singleton measurement is taken occurs.
MP_1, MP_2 = source and destination measurement points
P(*i*) = *i*th measured packet in a stream of packets
dT_i = one-way delay for P(*i*)

FIGURE 10.8 Model for Defining Packet Delay Variation

Section b of Table 10.2 lists two metrics that are not defined statistically. Connectivity deals with the issue of whether a transport-level connection is maintained by the network. The current specification (RFC 2678) does not detail specific sample and statistical metrics but provides a framework within which such metrics could be defined. Connectivity is determined by the ability to deliver a packet across a connection within a specified time limit. The other metric, bulk transfer capacity, is similarly specified (RFC 3148) without sample and statistical metrics but begins to address the issue of measuring the transfer capacity of a network service with the implementation of various congestion control mechanisms.

10.7 OpenFlow QoS Support

OpenFlow offers two tools for implementing QoS in data plane switches. The sections that follow examine each of these in turn.

Queue Structures

An OpenFlow switch provides limited QoS support through a simple queuing mechanism. One or more queues can be associated with a port. Queues support the ability to provide minimum data rate guarantees and maximum data rate limits. Queue configuration takes place outside the OpenFlow protocol, either through a command-line tool or through an external dedicated configuration protocol.

A data structure defines each queue. The data structure includes a unique identifier, port this queue is attached to, minimum data rate guaranteed, and maximum data rate. Counters associated with each queue capture the number of transmitted bytes and packets, number of packets

dropped because of overrun, and the elapsed time the queue has been installed in the switch.

The OpenFlow Set-Queue action is used to map a flow entry to an already configured port. Thus, when an arriving packet matches a flow table entry, the packet is directed to a given queue on a given port.

The behavior of the queue is determined beyond the scope of OpenFlow. Thus, although OpenFlow provides a way to define queues, direct packet flows to specific queues, and monitor traffic on each queue, any QoS feature must be implemented outside of OpenFlow.

Meters

A meter is a switch element that can measure and control the rate of packets or bytes. Associated with each meter is a set of one or more bands. If the packet or byte rate exceeds a predefined threshold, the meter triggers the band. The band may drop the packet, in which case it is called a **rate limiter**. Other QoS and policing mechanisms can be designed using meter bands. Each meter is defined by an entry in the meter table for a switch. Each meter has a unique identifier. Meters are not attached to a queue or a port; rather, a meter can be invoked by an instruction from a flow table entry. Multiple flow entries can point to the same meter.

With that brief overview, let's look at the details of meters. A meter measures the rate of packets assigned to it and enables controlling the rate of those packets. The meter measures and controls the rate of the aggregate of all flow entries to which it is attached. Multiple meters can be used in the same table, but in an exclusive way (disjoint set of flow entries). Multiple meters can be used on the same set of packets by using them in successive flow tables.

Figure 10.9 shows the structure of a meter table entry and how it is related to a flow table entry.

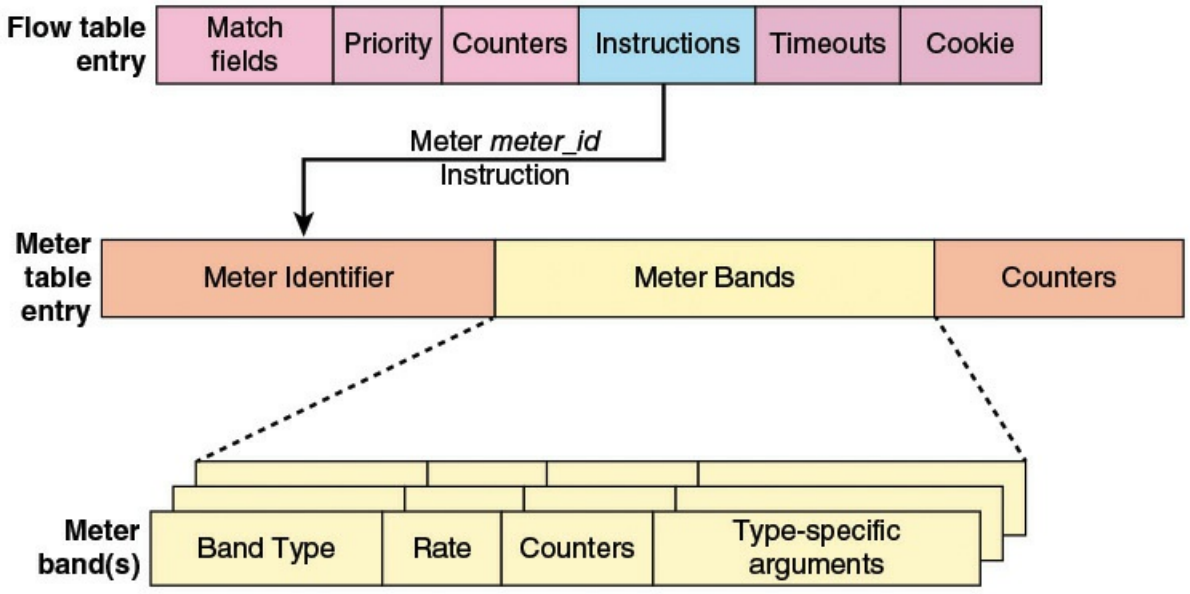


FIGURE 10.9 OpenFlow QoS-Related Formats

A flow table entry may include a `meter` instruction with a `meter_id` as an argument. Any packet that matches that flow entry is directed to the corresponding meter. Within the meter

table, each entry consists of three main fields:

- **Meter identifier:** A 32-bit unsigned integer uniquely identifying the meter.
- **Meter bands:** An unordered list of one or more meter bands, where each meter band specifies the rate of the band and the way to process the packet.
- **Counters:** Updated when packets are processed by a meter. These are aggregate counters. That is, the counters count the total traffic of all flows, and do not break the traffic down by flow.

Each band has the following structure:

- **Band type:** `drop` or `dscp remark`.
- **Rate:** Used by the meter to select the meter band, defines the lowest rate at which the band can apply.
- **Counters:** Updated when packets are processed by a meter band.
- **Type specific arguments:** Some band types may have optional arguments. Currently, the only optional argument is for the `dscp remark` band type, specifying the amount of drop in precedence.

The meter triggers a meter band if the packet rate or byte rate passing through the meter exceed a predefined threshold. A band of type drop drops packets when the band's rate is exceeded. This can be used to define a rate limiter band. A band of type `dscp remark` increases the drop precedence in the DS codepoint field in the IP header of the packet. This can be used to define a simple DiffServ policer.

Figure 10.10, from the *OpenFlow Switch Specification* (Version 1.5.1, March 2015), illustrates the use of OpenFlow to set, modify, and match on DSCPs. The figure shows three flow tables in one switch. Multiple flow entries in one flow table may use the same meter. Different entries in the same flow table may point to different meters, and a flow entry need not use a meter. By using different meters in a flow table, disjoint set of flow entries can be metered independently. Packets may go through multiple meters when using meters in successive flow tables, at each flow table the matching flow entry may direct it to one meter. The black arrowed lines indicate the progress of one flow through the flow tables. Figure 10.10 shows how multiple meters can be used for a given flow as the flow passes through the network, with the DCSP value changing based on traffic conditions observed by the meters.

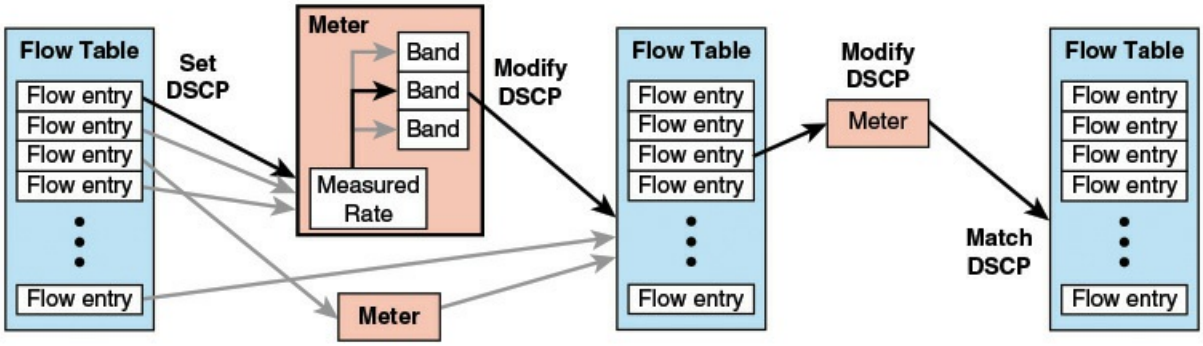


FIGURE 10.10 DSCP Metering

10.8 Key Terms

After completing this chapter, you should be able to define the following terms.

best effort

differentiated services

DS codepoint

elastic traffic

inelastic traffic

Integrated Services Architecture (ISA)

IP performance metrics

jitter

OpenFlow meter

quality of service (QoS)

service level agreements (SLA)

10.9 References

CISC15: Cisco Systems. *Internetworking Technology Handbook*. July 2015. http://docwiki.cisco.com/wiki/Internetworking_Technology_Handbook

CLAR98: Clark, D., and Fang, W. "Explicit Allocation of Best-Effort Packet Delivery Service." *IEEE/ACM Transactions on Networking*, August 1998.

Chapter 11. QoE: User Quality of Experience

By Florence Agboma
British Sky Broadcasting

> It is, of course, important to distinguish between the objective and subjective views, but we cannot pretend the latter are of no concern. Dismissal of subjective matters as being scientifically indecent springs from an excessive zeal for detachment. The objective view, which is predominant in the physical sciences and in strict behaviorist psychology, comes from regarding the observer as being "in" the world, which is out there around him and he can see it "through" his eyes. The subjective view comes from regarding the world as being in the mind of the observer, reality as mental experience.
>
> —*On Human Communication*, Colin Cherry, 1957

Chapter Objectives: **After studying this chapter, you should be able to**

- Explain the motivations for QoE.
- Define QoE.
- Explain the factors that could influence QoE.
- Present an overview of how QoE can be measured, including a discussion of the differences between subjective and objective assessment.
- Discuss the various application areas of QoE.

This chapter discusses quality of experience (QoE) by providing background information and motivations for its emergence and use. It also discusses the key features of QoE and the factors influencing it. The primary focus is that of QoE within the context of multimedia communication systems, given that bad network performance often highly affects the user's experience.

11.1 Why QoE?

Before the advent of the public Internet, video content delivery was a monopoly of content publishers who delivered their products and services over closed video delivery systems built and managed by cable and satellite TV operators. The operators owned and operated the entire distribution chain as well as the video reception devices (set-top boxes) in the home. These closed networks and devices were under the full control of these operators and were designed, deployed, provisioned, and optimized specifically to deliver high-quality video to consumers.

Figure 11.1 shows an abstraction of the typical satellite TV end-to-end delivery chain. In practice, however, such content delivery and distribution chains are made up of very complex integrations of applications and systems.

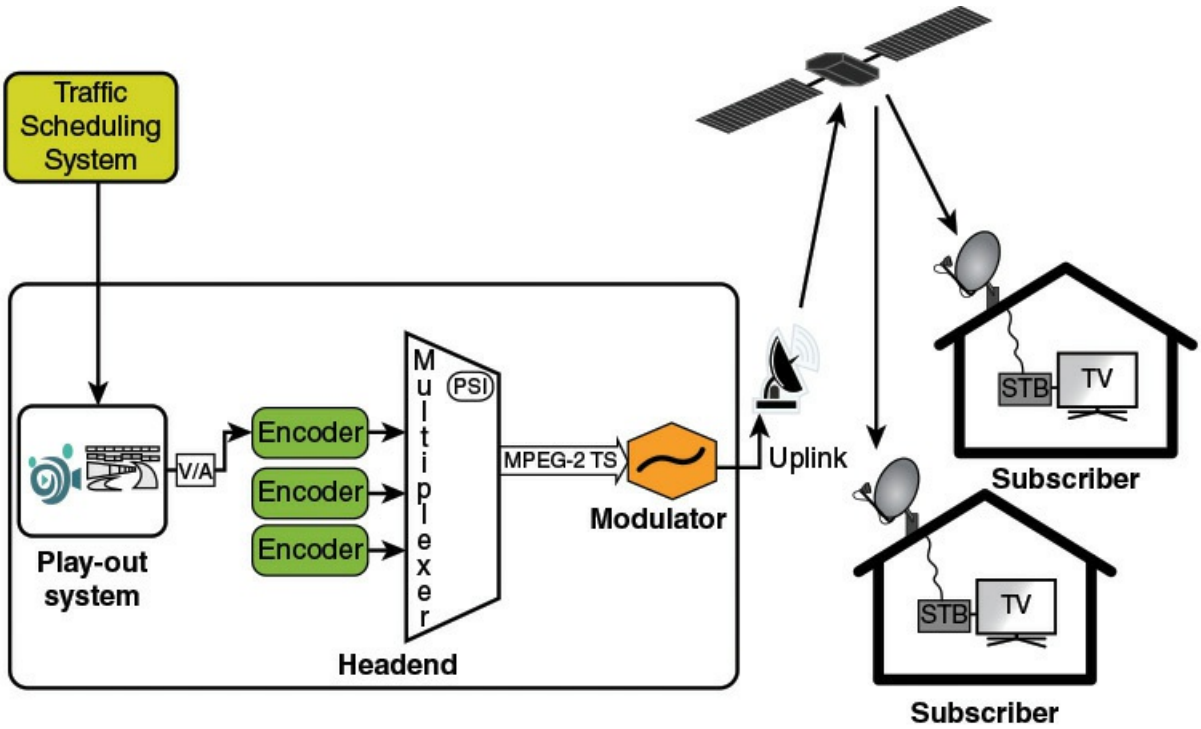


FIGURE 11.1 An Abstraction of a Content Distribution Network Using a Typical Satellite TV Distribution Network

As the illustration shows, the traffic (which in broadcasting means "program material") scheduling system provides audio and video (A/V) content via the play-out system to be encoded and aggregated into a single MPEG transport stream (TS). Together with the program specific information (PSI), the transport stream is transmitted to the subscriber's set-top box (STB) via a satellite.

Online Video Content Delivery

Video delivery over the Internet takes a different approach. Because numerous subnetworks and devices that constitute the Internet are situated in varied geographical locations, video streams reach the user by traversing through uncharted territories, as illustrated in Figure 11.2. With this arrangement, the guaranteeing of a good network performance is often a very challenging task.

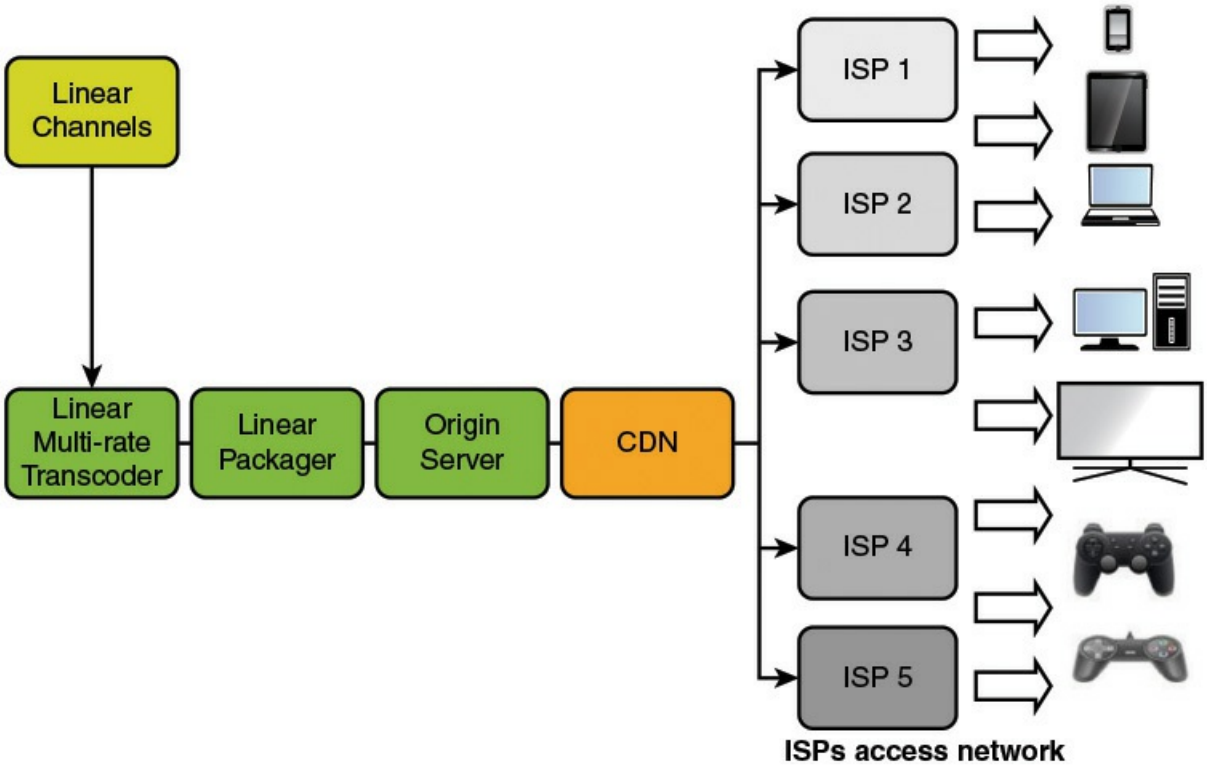


FIGURE 11.2 An Abstraction of a Content Distribution Network Using the Public Internet Distribution Network

Internet service providers (ISPs) do not own the entire content distribution network, and the risk of quality degradations is high. The access network may consist of coax, copper, fiber, or wireless (fixed and mobile) technology. Issues such as packet delay, jitter, and loss may plague such networks.

The growth and expansion of the Internet over the past couple of decades has led to an equally huge growth in the availability of network-enabled video streaming services. Giant technological strides have also been made in the development of network access devices.

With the current popularity of these services, providers need to ensure that user experiences are comparable to what the users would consider to be their reference standards. Users' standards are often influenced by the typically high video quality experience with the older technology, that is, those offered by the cable and satellite TV operators. User expectations can also be influenced by capabilities that currently can only be adequately offered by broadcast TV. These capabilities include the following:

- **Trick mode** functionalities, which are features of video streaming systems that mimic visual feedback given during fast-forward and rewind operations.
- **Contextual** experiences across multiple screens, which includes the ability to pause viewing on one screen and switch to another, thus letting users take the video experience with them on the go.

To manage user experiences for online services, quality of service (QoS) frameworks became the adopted set of technologies and tools employed in managing network traffic in the delivery

systems that provide these services. The aim of QoS is to manage the performance of networks and to provide performance guarantees to network traffic. QoS enables the measurement of network parameters, and the detection of changing network conditions (for example, congestion, availability of bandwidth), with the aim to implement stabilization strategies such as resource management and traffic prioritization.

There is now a growing realization, however, that QoS processes by themselves are not fully adequate in providing performance guarantees, because they do not take into account the user perception of network performance and service quality. It is this realization that has led to the emerging discipline of QoE.

The proliferation of different types of access devices further highlights the importance of QoE frameworks. As an illustration, the QoE for a user watching a news clip on a PDA will most likely differ from another user watching that same news clip on a 3G mobile phone. This is because the two terminals come with different display screens, bandwidth capabilities, frame rates, codecs, and processing power. Therefore, delivering multimedia content or services to these two terminal types, without carefully thinking about the users' quality expectations or requirements for these terminal types, might lead to service overprovisioning and network resource wastage.

Informally, QoE refers to the user perception of a particular service. QoE needs to be one of the central metrics employed during the design and management of networks, content delivery systems, and other engineering processes. This is because it refers to a measure of the end-to-end performance at the service level from the user's perspective as measured at the end user devices.

➔ *See Section 11.4, "Definition of Quality of Experience"*

11.2 Service Failures Due to Inadequate QoE Considerations

The stereoscopic 3D TV service is often cited as a prime example of a service that was a spectacular commercial failure because it had very poor QoE ratings.

In 2010, broadcasters such as Disney, Foxtel, BBC, and Sky began actively making 3D content delivery available as a service to their customers as a premium service experience. Indeed, each of these broadcasters rolled out their own dedicated 3D television channels. Within five years, all of them except Sky had to terminate their operations.

A number of factors contributed to the failure of these services. The first was the general unavailability of "wow video content" (that is, content that users are most likely to find exciting or take much interest in). The second was the need to wear special 3D glasses even when using these services in a home environment. Third, because broadcasters were initially in a rush to deploy the 3D TV technology, content was produced by inexperienced creators using inadequate systems and tools. This resulted in a great deal of poorly produced 3D content, which may have alienated the early subscribers.

11.3 QoE-Related Standardization Projects

Because the field of QoE has been growing rapidly, a number of projects have been initiated to address issues relating to best practices and standards. These projects have been aimed at

preventing commercial failures like the one described in the Section 11.2. Table 11.1 summarizes the prominent ones amongst these project initiatives, two of which are described in the paragraphs that follow.

Organization	Mission	QoE-Related effort
QUALINET	A multidisciplinary consortium for QoE research	A common terminology for QoE framework
Eureka Celtic	A collaborative industry-driven European research in the area of telecommunications	Quality of Experience Estimators in Networks (QuEEN) agent to estimate QoE for generic services
International Telecommunication Union—Telecommunication Standardization Sector (ITU-T)	United Nations agency that produces recommendations with a view to standardizing telecommunications on a worldwide basis	QoE standardization IPTV QoE requirements
IEEE Standards Association (IEEE-SA)	A standards-setting body within IEEE, develops consensus standards through an open process that engages industry and brings together a broad stakeholder community	Standard for Network-Adaptive Quality of Experience (QoE)

TABLE 11.1 QoE Initiatives and Projects

The Video Quality Experts Group (VQEG) is currently working on draft ITU recommendations for 3D video quality assessments for home entertainment systems (http://www.its.bldrdoc.gov/vqeg/projects/3dtv/3dtv.aspx). The VQEG is also working on producing reference documentation regarding the features that can impact 3D TV viewing experience, as well as ways in which they can be minimized. Examples of these features are crosstalk, visual discomfort, and visual fatigue.

Video Quality Experts Group (VQEG)

Another initiative is by the Quality of Experience Estimators in Networks (QuEEN) project [ETSI14], which is a multi-organizational and multinational initiative aimed at addressing issues relating to online services such as voice, video, and IPTV. These are areas where service and network providers seek to differentiate their service offerings, in terms of QoE and lower churn rates, from their competitors.

QuEEN developed an operational framework by categorizing the factors that may have an influence on QoE into well-defined layers. This provided an insight into how each layer could be associated with a quality value. The process of the QoE estimation employed the use of a

software agent that integrated with each of the layers, and also with software systems that attempt to model how a human subject would give approval ratings based on the values of these parameters. The agent had the capability of aggregating data from various probes across the network.

The QuEEN agent was at the core of the layered model. It enabled the flexible deployments of QoE estimators in a large-scale distributed environment. The three-year QuEEN project, which concluded in 2014, produced some impressive results. The QuEEN approach of using software QoE agents has been standardized in different ETSI and ITU standards. It is envisaged that this will encourage the development of new ways of using QoE, as well as new methods in QoE management.

11.4 Definition of Quality of Experience

There are a number of different, although similar, definitions of QoE. The nature of QoE, which turns out to vary from person to person, is difficult to grasp in a quantitative way. QoE requires a multidisciplinary approach, encompassing communication networks, cognitive processes, multimedia signal processing, and social psychology, focused on understanding the user perception of quality.

Researchers working within these various disciplines often use their own specialist language and terminology in describing identical concepts. Thus, studying and interpreting literature from a given discipline is usually not a trivial exercise for researchers from other disciplines. As a consequence, there is the lack of a consensus of how to measure or describe QoE and the wide range of factors that influence it.

A first step toward a multidisciplinary approach to QoE involves specifying a common terminology framework.

Work toward drawing up this common framework was begun in 2012 by the European Network on Quality of Experience in Multimedia Systems and Services (QUALINET) [MOLL12]. This is a group of researchers and industry experts whose main objectives were to foster discussions about the formal definitions of QoE and its related concepts.

The definitions of quality, experience, and quality of experience presented in this section are based on the ones provided in the QUALINET's white paper of definitions [MOLL12].

Definition of Quality

Quality is the resulting verdict produced by a user after he/she has carried a "comparison and judgment" process on an observable occurrence or event.

This process comprises the following key sequential steps:

- **Perception** of the event
- Reflection on the perception
- Description of the perception
- Evaluation and description of the result or outcome

Thus, quality is evaluated in terms of the degree to which the user's needs have been fulfilled within the context of the event. The result of this evaluation is usually referred to as the quality score (or rating) if it is presented with reference to a scale.

Definition of Experience

Experience is an individual's description of a stream of perceptions, and his/her interpretation of one or multiple events. An experience might result from an encounter with a system, service, or an artifact.

It is important to note that the description of an experience need not necessarily result in a judgment of its quality.

Quality Formation Process

As shown in Figure 11.3, there are two distinct subprocess paths to the formation of a quality score: the perception path and the reference path.

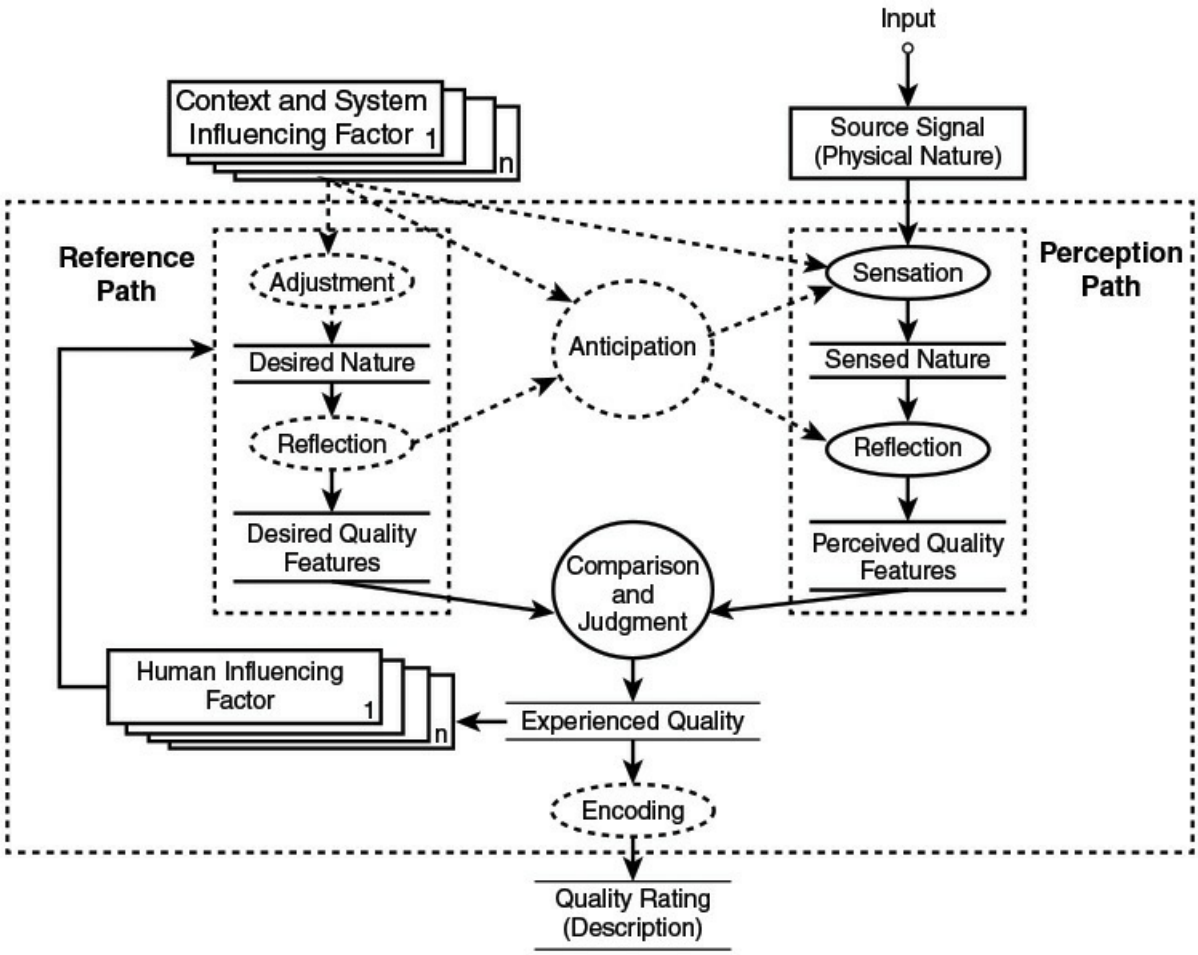


FIGURE 11.3 A Schematic Illustration of the Quality Formation Process from an Individual Point of View. Source: [MOLL12]

The reference path reflects the temporal and contextual nature of the quality formation process. This path is influenced by memories of former experienced qualities, as indicated by the arrow from experienced quality to the reference path.

The perception path is characterized by the physical input signal, which is to be assessed, reaching the sensory organs of the observer. This physical event is processed through low-level perceptual processes into a perceived feature within the constraints of the reference path. This perceived feature undergoes a reflection process, which interprets these sensory features through **cognitive** processing. At this point, the perceived concepts can be described and potentially quantified to become perceived quality features.

The quality features resulting from the reference and perception paths are then translated into the experienced quality on behalf of the comparison and judgment process. This experienced quality is delimited in time, space, and character, and thus can be called a quality **event**. The relevant information about the event can only be obtained on a descriptive level from the user.

The final step of the quality formation lies in some kind of comparison of the expected and experienced features. In this particular case, the output of the quality formation process corresponds to the quality of experiencing.

Definition of Quality of Experience

Combining the concepts and definitions from the preceding sections, the definition of QoE that reflects broad industry and academic consensus is as follows:

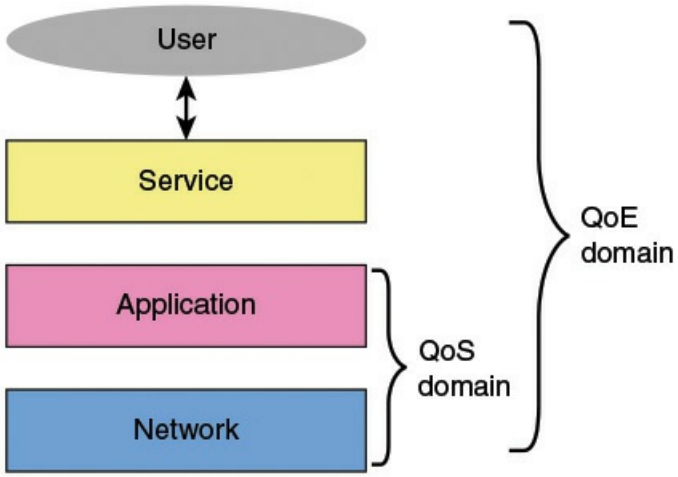
> Quality of experience (QoE) is the degree of delight or annoyance of the user of an application or service. It results from the fulfillment of his or her expectations with respect to the utility/enjoyment of the application or service in the light of the user's personality and current state.

11.5 QoE Strategies in Practice

Key findings from QoE-related projects show that for many services, multiple QoS parameters contribute toward the overall user's perception of quality. This has resulted in the emergence of the concept of the QoE/QoS layered approach in which the requirements of the users drive network-dimensioning strategies.

The QoE/QoS Layered Model

The QoE/QoS layered approach does not ignore the QoS aspect of the network, but instead, user and service level perspectives are complementary, as shown in Figure 11.4.



Note that because there is an overlap between the QoE and QoS domains, there is a considerable amount of information sharing/feedback between the frameworks.

FIGURE 11.4 QoE/QoS Layered Model with the Domains of Interest for the Frameworks

The levels in the layered approach are as follows:

- **User**: The user interacts with the service. It is their degree of delight or annoyance from using the service that is to be measured. Being linked to human perception, QoE is hard to describe in a quantitative way, and it varies from person to person. The complexities of QoE at the user level stem from the differences between individual user characteristics, of which some might be time-varying, whereas others are of a relatively stable nature. Examples could include gender, age, attitudes, prior experience, expectations, socio-economic status, cultural background, educational level, and so on. Therefore, it becomes a challenge to derive unified QoE metrics for all users and their contexts. The current practice in any QoE measurement is to identify and control for the relatively stable characteristics of a user in a way that is satisfactory to at least a large proportion of the potential user group.
- **Service**: The service level provides a virtual level where the user's experience of the overall performance of the service can be measured. It is the interface where the user interacts with the service (for example, the visual display to the user). It is also where tolerance thresholds are measured. As an illustration, the QoE measures from the user perspective for streaming applications could be startup time, audio/visual quality, channel change delay, and buffering interruptions. However, the QoE measures for web browsing applications could be page load waiting times.
- **Application-level QoS (AQoS)**: AQoS deals with the control of application-specific parameters such as content resolution, bit rate, frame rate, color depth, codec type, layering strategy, and sampling rate. The network capacity often dictates the bandwidth that will be allocated to a service for transmission. Because of this fixed underlying resource, some parameters at the application level are usually adjusted and controlled to achieve a desired quality level. For example, for an audio service, a sampling rate of 96 kHz might allow for more information to be audibly perceived as compared to a 48-kHz rate. But this larger sampling rate comes with the expense of generating bigger audio file

sizes. This is because the sampling rate is the number of times an analog sound signal is measured per second. Each of these measurements (or samples) is stored or transmitted as a digital value.

As another example, for video services, there is a huge variety in device screen sizes (each featuring varied aspect ratios) from which to choose. The one common feature in this array of equipment is that they are all capable of rescaling video images. For a given bit rate, there might be a trade-off between lower resolution images that are slightly blurred and with fewer digital artifacts (visual anomalies) versus higher resolution images that provide sharper images but possibly having more artifacts. The bit rate usually provides an indication of the quality of a video (or audio) file. This is because it represents the number of bits used in encoding each second of a file. Most compression standards use block-based and motion compensation coding schemes and as a result, additional compression artifacts are added to the decoded video.

- **Network-level QoS (NQoS)**: This level is concerned with the low-level network parameters such as service coverage, bandwidth, delay, throughput, and packet loss. There are a number of ways in which network-level QoS parameters impact QoE. One such way is via network delay, which impacts QoE especially for interactive services. For instance, the interactive nature of web browsing that requires multiple retrieval events within a certain window of time might be affected by delay variations of the network. Voice over IP (VoIP) services might have stringent response-time demands, whereas e-mail services might tolerate much longer delays.

The different distribution methods of streaming video over the network also affect QoE in different ways. For instance, HTTP-based adaptive streaming, which uses TCP, reacts to bandwidth constraints and CPU capacity in either of the following ways:

- Switching to streaming using other available bit rate encodings, depending on available resources
- Frame freeze (rebuffering) occurring because of incoming packet starvation in the player buffer

The bit rate switches and rebuffering have an adverse effect on QoE.

UDP streaming, however, uses multicast to replicate the streams throughout the network. Quite often a resilient coding scheme and a flow control mechanism are implemented to maintain the viewing experience despite the effects of bad network conditions.

Summarizing and Merging the QoE/QoS Layers

The preceding discussion suggests that the effect of QoE could be an attribute of only the application layer or a combination of both the application and network layers. Although the trade-offs between quality and network capacity may begin with application-level QoS because of network capacity considerations, an understanding of the user requirements at the service level (that is, in terms QoE measures) would enable a better choice of application-level QoS parameters to be mapped onto the network-level QoS parameters. A scenario that aims at controlling QoE using QoS parameters as actuators is discussed in Section 11.8.

11.6 Factors Influencing QoE

QoE must be studied and addressed by taking into account both technical and nontechnical factors. Many factors contribute to producing a good QoE. Here, the key factors are as follows:

- **User demographics**: The context of demographics herein refers to the relatively stable characteristics of a user that might have an indirect influence on perception, and intimately affects other technical factors to determine QoE. In a landmark project [QUIN12] studying the adoption of HD voice telephony, the different user groups produced significantly different quality ratings. The grouping of users was based on demographic characteristics such as their attitudes toward adoption of new technologies, socio-demographic information, socioeconomic status, and prior knowledge. Cultural background is another user demographic factor that might also have an influence on perception because of cultural attitude to quality.
- **Type of device**: Different device types possess different characteristics that may impact on QoE. An application designed to run on more than one device type, for example on a connected TV device such as Roku and on an iOS device such as an iPhone, may not deliver the same QoE on every device.
- **Content:** Content types can range from interactive content specifically curated according to personal interests, to content that is produced for linear TV transmission. Studies have suggested that people tend to watch video on-demand (VoD) content with a higher level of engagement than its competing alternative, linear TV. This may be because users will make an active decision to watch specific VoD content, and as a result, give their full attention to it. One could infer that for VoD users might be less tolerant of any quality degradations because of their high level of engagement.
- **Connection type:** The type of connection used to access the service influences users' expectations and their QoEs. Users have been found to have lower expectations when using 3G connections in contrast to a wire line connection even when the two connection types were identical in terms of their technical conditions. Users have also been found to lower their expectations considerably, and are more tolerant to visual impairments, on small devices.
- **Media (audio-visual) quality:** This is a significant factor affecting QoE, as it is the part of a service that is most noticeable by the user. The overall audio and video quality appears to be content dependent. For less-complex scenes (for example, head and shoulder content), audio quality is slightly more important than video quality. In contrast, for high-motion content, video quality tends to be significantly more important than audio quality.
- **Network:** Content delivery via the Internet is highly susceptible to the effects of delays, jitter, packet loss, and available bandwidth. Delay variation results in the user experiencing frame freeze and the lack of lip synchronization between what is heard (audio) and what is seen (video). Although video content can be delivered using a number of Internet protocols, not all of them are reliable. However, content delivery is guaranteed using TCP/IP. Nevertheless, bad network conditions degrade QoE because of increased rebuffering and increased interruptions in playback. Rebuffering interruptions in IP video playback is seen to be the worst degradation on user QoE and should be avoided at the

cost of startup delay. On the same note, QoE for a given startup delay strongly depends on the application context and the user expectations. In spite of the different QoE factors that are concerned with the network, reliability and a strong wireless signal are crucial for consuming TV-like services.

- **Usability:** Another QoE factor is the amount of effort that is required to use the service. The service design must render good quality without a great deal of technical input from the user.
- **Cost:** The long-established practice of judging quality by price implies that expectations are price dependent. If the tariff for a certain service quality is high, users may be highly sensitive to any quality degradations.

11.7 Measurements of QoE

QoE measurement techniques evolved through the adaptation and application of psychophysics methods during the early stages of television systems. This section introduces three QoE measurement methods: subjective assessment, objective assessment, and end-user device analytics.

Subjective Assessment

For subjective assessment of QoE, experiments are carefully designed to a high level of control (such as in a controlled laboratory, field tests, or crowdsourcing environments) so that the validity and reliability of the results can be trusted. It might be useful to consult expert advice during the initial design of the subjective experiment, because the topics of experimental design, experimental execution, and statistical analysis are complex. In general terms, a methodology to obtain subjective QoE data might consist of the following phases:

- **Characterize the service:** The task at this stage is to choose the QoE measures that affect user experience the most. As an example, for a multimedia conferencing service, the quality of the voice takes precedence over the quality of video. Also, the video quality required for such applications does not demand a very high frame rate, provided that audio-to-video synchronization is maintained. Therefore, the resolution of individual frames can be considerably lower than the case of other video streaming services, especially when the size of the screen is small (such as a mobile phone). So, in multimedia conferencing, the QoE measures might be prioritized as voice quality, audio-video synchronization, and image quality.
- **Design and define test matrix:** Once the service has been characterized, the QoS factors that affect the QoE measures can be identified. For instance, the video quality in streaming services might be directly affected by network parameters such as bandwidth, packet loss, and encoding parameters such as frame rate, resolution, and codec. The capability of the rendering device will also play a significant role in terms of screen size and processing power. However, testing such a large combination of parameters may not be feasible. This draft matrix could be reduced to more achievable test conditions by eliminating the combinations that have similar effects on QoE.

- **Specify test equipment and materials:** Subjective tests should be designed to specify test equipment that will allow the test matrix to be enforced in a controlled fashion. For instance, to assess the correlation between NQoS parameters and the perceived QoE in a streaming application, at least a client device and a streaming server separated by an emulated network are needed. If the objective is to evaluate how different device capabilities impact QoE, a video content is chosen to produce formats that can run in each of the client devices under scrutiny.
- **Identify sample population:** A representative sample population is identified, possibly covering different classes of users categorized by the user demographics that are of interest to the experimenter. Depending on the target environment for the subjective test, at least 24 test subjects has been suggested as the ideal number for a controlled environment (for example, a laboratory) and at least 35 test subjects for a public environment. Fewer subjects may be used for pilot studies to indicate trending. The use of crowdsourcing in the context of subjective assessment is still nascent, but it has the potential to further increase the size of the sample population and could reduce the completion time of the subjective test.
- **Subjective methods**: Several subjective assessment methodologies exist within the industry recommendations. However, in most of them, the typical recommendation is for each test subject to be presented with the test conditions under scrutiny along with a set of rating scales that allows the correlation of the users' responses with the actual QoS test conditions being tested. There are several rating scales, depending on the design of the experiment.
- **Analysis of results**: When the test subjects have rated all QoS test conditions, a post-screening process might be applied to the data to remove any erroneous data from a test subject that appears to have voted randomly. Depending on the design of the experiment, a variety of statistical approaches could be used to analyze results. The simplest and the most common quantification method is the mean opinion score (MOS), which is the average of the opinions collected for a particular QoS test condition. The results from subjective assessment experiments are used to quantify QoE, and to model the impacts of QoS factors. Subjective experiments require significant planning and design so as to produce reliable subjective MOS ratings. However, they are time-consuming, expensive to carry out, and are not feasible for real-time in-service monitoring. In such situations, the use of objective assessment is often desirable.

Objective Assessment

For objective assessment of QoE, computational algorithms provide estimates of audio, video, and audiovisual quality as perceived by the user. Each objective model targets a specific service type. The goal of any objective model is to find the optimum fit that strongly correlates with data obtained from subjective experiments. The following phases presented here should not be considered as exhaustive, but aim at illustrating a process of obtaining objective QoE data. A methodology to obtain objective QoE data might consist of the following phases:

- **Database of subjective data**: A starting point might be the collection of a group of subjective datasets as this could serve as benchmark for training and verifying the

performance of the objective model. A typical example of one of these datasets might be the subjective QoE data generated from well-established subjective testing procedures, as discussed earlier. The selection of the subjective datasets should typically reflect the use cases of the objective model.

- **Preparation of objective data**: The data preparation for the objective model might typically include a combination of the same QoS test conditions as found in the subjective datasets, as well as other complex QoS conditions. A variety of preprocessing procedures might be applied to the video data prior to training, and refinement of the algorithm.
- **Objective methods**: There are various algorithms in existence that can provide estimates of audio, video, and audiovisual quality as perceived by the user. Some algorithms are specific to a perceived quality artifact, while others can provide estimates for a wider scope of quality artifacts. Examples of the perceived artifacts might include blurring, blockiness, unnatural motion, pausing, skipping, rebuffering, and imperfect error concealment after transmission errors.
- **Verification of results**: After the objective algorithm has processed all QoS test conditions, the predicted values might benefit from a post-screening process to remove any outliers; this is the same concept applied to the subjective datasets. The predicted values from the objective algorithm might be in a different scale as compared to the subjective QoE datasets. The predicted values might be transformed to the same scale as obtained in the subjective experiments (for example, into the mean opinion scores) to enable like-for-like comparisons, and also so that an optimum fit between the predicted QoE values and subjective QoE data can be obtained.
- **Validation of objective model**: The objective data analysis might be evaluated with respect to its prediction accuracy, consistency, and linearity by using a different subjective dataset. It is worth noting that the performance of the model might depend on the training datasets and the verification procedures. The Video Quality Experts Group (VQEG) validates the performance of objective perceptual models so that they can become ITU recommendations and standards for objective quality models for both television and multimedia applications.

End-User Device Analytics

End-user device analytics is yet another alternative method of QoE measurement. Real-time data such as the connection time, bytes sent, and average playback rate are collected by the video player application for each video viewing session and fed back to a server module where the data is pre-aggregated and then turned into actionable QoE measures. Some of the metrics reported for per-user and aggregate viewing sessions include startup delay, rebuffering delays, average bit rates, and the frequency of bit rate switches.

Operators may be inclined to associate viewer engagement levels with their QoE because good QoEs usually make viewers less likely to abandon a viewing session. The definition of viewer engagement may have different meanings for different operators and context. First of all, operators might like to know which viewer engagement metrics affect QoE the most to guide the design of the delivery infrastructures. Second, they might also like to quickly identify and resolve service outages, and other quality issues. A minute of encoder glitch could replicate

throughout the ISPs, and the various delivery infrastructures, and affect all their customers. Operators might like to know the scale of this impact, and how it affects users' engagement. Finally, they would like to understand their customers' demographics (connection methods, type of device, bit rates of the consumed asset) within a demographic region so that resources can be strategically dimensioned.

QoE enthusiasts advocate QoE measurement to be a multidisciplinary approach that seeks to explain its findings, building on general laws of perception, sociology, and user psychology. With the use of end-user device analytics as a means of QoE measurement, there are many variables that cannot be accounted for (for example, why a user exits a service). A lack of interest in watching the content might result in a user exiting a service, and not necessarily because of poor QoE.

One method of tackling these unexplained variables is to use the fraction of video viewed as a measure of engagement because this can be measured objectively. The data that appears to belong to early quitters can then be systematically removed from any analyses to obtain a clearer understanding of how the QoE measures impact viewer engagement.

Summarizing the QoE Measurement Methods

The mean opinion score (MOS) appears to be the de facto standard metric for QoE. The possible reasons could be its long-term establishment in telephony networks, and perhaps its widespread acceptance on the merits that it can be easily understood. There are different types of MOS values and different test methodologies to produce them. See ITU-T Recommendation P 913, *Methods for the Subjective Assessment of Video Quality, Audio Quality and Audiovisual Quality of Internet Video and Distribution Quality Television in Any Environment*, 2014, for more details. Table 11.2 shows the five-point absolute category rating MOS scale that is commonly used

Score	Label
5	Excellent
4	Good
3	Fair
2	Poor
1	Bad

TABLE 11.2 Five Point MOS Rating Scale

The MOS value is the average opinion for the group of users, for a given QoS test condition. It is not necessarily the opinion score for an individual user, because different users have different opinions. Additional information such as statistical uncertainty in terms of confidence intervals is usually encouraged. The MOS is considered to be characteristic of only the experiment and the group of test subjects from which it was derived.

MOS has to be interpreted within context. First of all, the MOS value obtained for a particular QoS test condition, in a subjective experiment, may depend on the range of the QoS test conditions used in the experiment. This might be due to test subjects recalibrating their use of the rating scale to the conditions in the experiment. An appropriately designed experiment whereby

there is a practice period at the start of the experiment, and the test conditions include the best and worst conditions, minimizes the effects of the aforementioned behavior.

Direct comparisons of MOS scores obtained from separate experiments are generally not meaningful. They are only meaningful if the experiments have been specially designed to enable such comparisons. Data from such specially configured experiments must be studied and shown that their MOS comparisons are statistically valid. Biases in the rating scale interpretation might exist because of differences in the test subject profiles (for example, age and technology exposure, test environment, and the presentation order of the test conditions).

It is possible that different objective models that have been trained and optimized using different subjective contexts will predict nonidentical MOS values for the same QoS conditions. Objective models are usually developed and optimized for a specific scope of quality features. As a consequence, comparisons between MOS predictions and thresholds can only be reliably made if the thresholds are chosen in the context of the MOS model.

Objective assessment seems to offer real-time QoE measurements, but end-user device analytics as a method of QoE measurement appears to be an alternative approach. Currently, there is the lack of a reference methodology for end-user device analytics as a method of QoE measurement, analogous to mean opinion scores found in subjective assessments and objective assessments.

A limiting factor to this development might be the restricted rights governing service providers on the usage of their databases. This makes it challenging for researchers, service providers, and delivery infrastructure designers to develop better delivery infrastructures.

Subjective experiments are probably still the most accurate way to measure QoE, and also the only way to obtain reliable ground truth data used in benchmarking objective QoE models.

11.8 Applications of QoE

The practical applications of QoE can be grouped into two areas based on the main usage.

- **Service QoE monitoring:** Service monitoring allows the support teams (for example, service provider and network operator) to continually monitor the quality experienced by the end users of the service. A service alert message might be sent to the support teams when QoE falls below a certain threshold value, as this will allow the support teams to quickly identify and resolve service outages and other QoE issues.

  Depending on the use case of the service being monitored, such monitoring tools might be located at any one node, or at all nodes, within the content delivery ecosystem. The nodes could be at the headend incoming feeds, distribution networks, and at endpoints locations. This approach might introduce high monitoring overheads for a per-user scenario.

- **QoE-centric network management:** The ability to control and optimize the user experience when QoE degradation issues arise is the holy grail of QoE network management. Given the multidimensional aspect of the overall QoE (such as the network-level conditions of the subnetworks, application-level QoS, device capability, and user demographics), a typical challenge lies in providing actionable QoE information feedback to the network or service provider.

  Two approaches in which QoE-centric network management can be exploited are as

follows:

- In the first approach, a set of QoS measurement values together with the appropriate assumptions, are used in computing the expected QoE for a user.
- In the second approach, which is somewhat the opposite of the first, a target QoE for a user together with the appropriate assumptions is used to produce estimates of the required QoS values.

The first approach can be taken by a service provider, who can provide a range of QoS offerings with an outline of the QoE that the customer might reasonably expect.

The second approach can be taken by a customer who defines the required QoE, and then determines what level of service will meet that need.

Figure 11.5 illustrates a scenario where the user can make a selection from a range of services, including the required level of service (SLA). By contrast to the purely QoS-based management, the SLA here is not expressed in terms of raw network parameters. Instead, the user indicates a QoE target; it is the service provider that maps this QoE target together with the type of service selected, onto QoS demands.

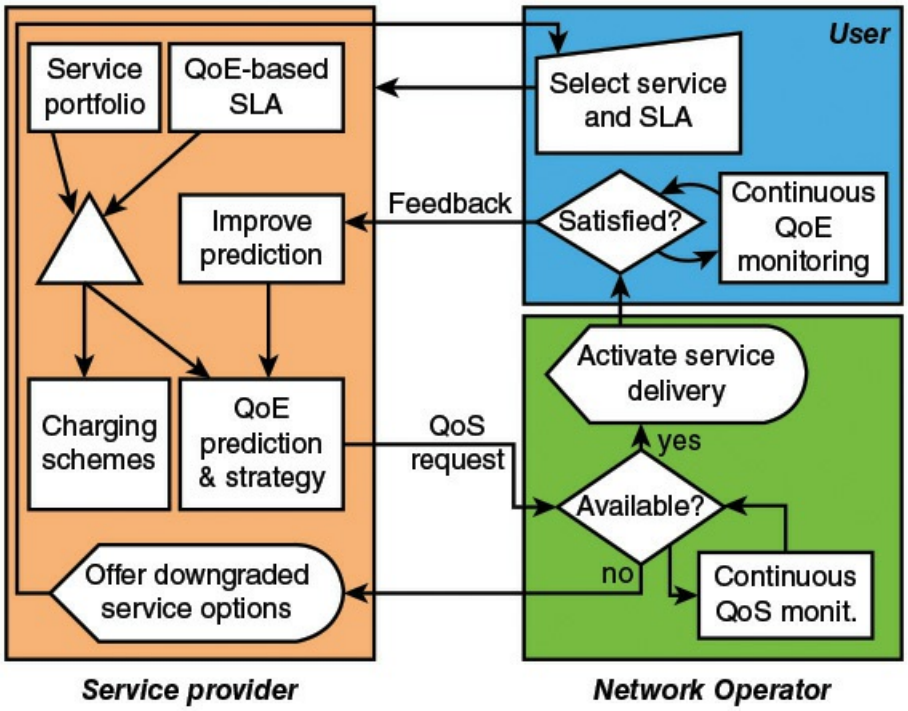


FIGURE 11.5 QoE-Centric Network Management

For instance, in the case of multimedia streaming service, the user may simply choose between two QoE levels (high or low). The service provider selects the appropriate quality prediction model and management strategy (for example, minimize network resource consumption) and forwards a QoS request to the operator. It is possible that the network cannot sustain the required level of QoS, making it impossible to deliver the requested QoE. This situation leads to a signal back to the user, prompting a reduced set of services/QoE values.

Assuming that the network can support the service, delivery can be activated. During service

operation, two monitoring and control loops run concurrently: one at network level and the other at service level. The latter allows the user to switch to a different level of QoE (for example, to get a cheaper service or to request higher quality). If the user generates no explicit feedback, this means that the user is satisfied, which confirms that the quality prediction model is working. In this way, the quality prediction model continues to be redefined during service delivery, allowing it to evolve as user needs and devices change over time.

11.9 Key Terms

After completing this chapter, you should be able to define the following terms.

trick mode

cognitive

QoE measurement

contextual

perception

subjective assessment

quality of experience

Event

Objective assessment

11.10 References

ETSI14: ETSI TS 103 294 V1.1.1 Speech and Multimedia Transmission Quality (STQ); Quality of Experience; A Monitoring Architecture (2014-12).

MOLL12: Moller, S., Callet, P., and Perkis, A. “Qualinet White Paper on Definitions on Quality of Experienced,” European Network on Quality of Experience in Multimedia Systems and Services (COST Action IC 1003) (2012).

QUIN12: M.R.Quintero, M., and Raake, A. “Is Taking into Account the Subjects’ Degree of Knowledge and Expertise Enough When Rating Quality?” Fourth International Workshop on Quality of Multimedia Experience (QoMEX), pp.194,199, 5[nd]7 July 2012.

Chapter 12. Network Design Implications of QoS and QoE

By Sofiene Jelassi

Assistant Professor, University of Monastir, Tunisia

> But some amazing experience had disturbed his native composure and left its traces in his bristling hair, his flushed, angry cheeks, and his flurried, excited manner.
>
> —*The Adventure of Wisteria Lodge*, Sir Arthur Conan Doyle

***Chapter Objectives*: After studying this chapter, you should be able to**

- Translate metrics from QoS to QoE domain.
- Select the appropriate QoE/QoS mapping model for a given operational situation.
- Deploy QoE-centric monitoring solutions over a given infrastructure.
- Deploy QoE-aware applications over QoE-centric infrastructure.

This chapter concludes Part Four by bringing together the concepts of quality of service (QoS) and quality of experience (QoE) and discussing the practical implications of employing these two concepts.

The chapter is organized as follows; section 12.1 classifies existing **QoS/QoE mapping models** from practical perspectives. Section 12.2 enumerates few IP-oriented QoE/QoS mapping models used for video services. Section 12.3 discusses approaches that could be used to add QoE capability to networks and services. Sections 12.4 and 12.5 describe respectively QoE-centric monitoring and management solutions.

12.1 Classification of QoE/QoS Mapping Models

Typically, mathematical models are used to define the empirical relationship between QoS and QoE. These models will be referred to hereafter as either *QoE/QoS mapping models* or *quality models*. They are derived using classical approaches that fit a model to a dataset, such as regression, artificial neural network, and Bayesian network. Today, a wide range of QoE/QoS mapping models are reported in the literature. They differ in term of their inputs, working modes, accuracy, and application areas. The application area of QoE/QoS mapping models depends mostly on their inputs. QoE/QoS mapping models can be classified according to their inputs into three categories:

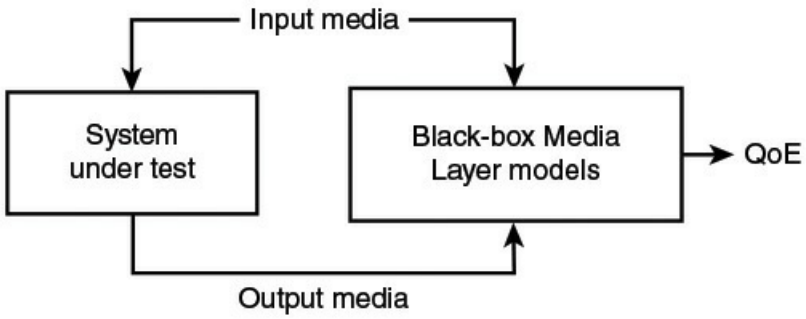
- Black-box media-based models
- Glass-box parameter-based models
- Gray-box parameter-based models

The sections that follow describe these models.

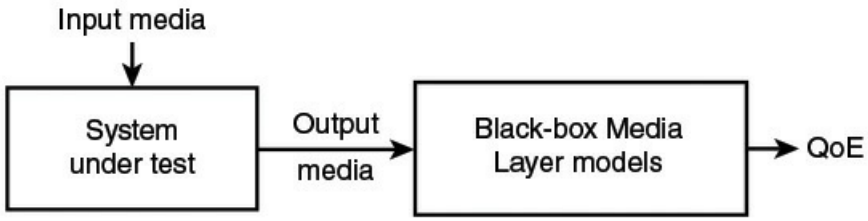
Black-Box Media-Based QoS/QoE Mapping Models

Black-box media-based quality models rely on the analysis of media gathered at system entrance and exit. Hence, they account implicitly for the characteristics of examined media processing system. They are classified into two categories:

- **Double-sided or full-reference quality models:** They use as inputs the clean stimulus and the corresponding degraded stimulus (see part a of Figure 12-1). They compare the clean and degraded stimulus in a *perceptual domain* that accounts for psychophysics capability of human sensory system. The perceptual domain is a transformation of traditional physical temporal and frequency domains performed according to characteristics of users perceptions. Basically, the larger the perceptual distance, the greater the degradation level. This model needs to align clean and degraded stimulus because the comparison is made on per-block basis. The stimulus alignment should be realized autonomously, that is, without adding extra control information describing stimulus structure.



(a): Double-sided or full-reference quality models.



(b): One-sided or no-reference mapping models.

FIGURE 12.1 Black-Box Media-Based QoS/QoE Mapping Models

- **One-sided or no-reference quality models**: They rely solely on the degraded stimulus to estimate the final QoE values. They parse the degraded stimulus to extract the observed distortions, which are dependent on the media type, for example, audio, image and video. As an example, artifacts extracted from audio stimulus include whistle, circuit noises, echoes, level saturation, clapping, interruptions, and pauses (see part b of Figure 11-1). The gathered distortions are adequately combined and transformed to compute the QoE values.

The main advantage of black-box quality models resides in their ability to measure QoE values using information gathered at the periphery of a given media processing system. Hence, they may be used in a generic fashion over different infrastructures and technologies. This sidesteps a complex and cumbersome measuring process of the underlying systems. Moreover, it enables enhancing unconditionally quality models, that is, independently of technical and ethical constraints related to the measurement processes. Furthermore, black-box quality models may easily operate on either per-user or per-content basis.

The main shortcoming of black-box quality models resides in the requirements to access the final representation of stimulus, which is often inaccessible in practice for privacy reasons. Moreover, full-reference quality models use clean stimulus as inputs that is often unavailable or hardly accessible at the system output. This issue may be sidestepped using no-reference quality models, but their unproved and instable performance confines their effectiveness.

The full-reference black-box quality models are widely used for onsite benchmarking, diagnosis, and tuning of network equipments, where clean stimulus is available. The no-reference quality models may be used for the same purposes, but their limited accuracy reduces their results credibility. The black-box quality models are used offline for the evaluation of application-layer components, such as codec, packet loss concealment (PLC), and buffering schemes. In addition, the no-reference black-box quality models may be used online for QoE monitoring.

Glass-Box Parameter-Based QoS/QoE Mapping Models

The glass-box parameter-based quality models quantify the QoE of a given service through the full characterization of the underlying transport network and edge devices. The set of considered characterization parameters and their combination rules are derived based on extensive subjective experiments and thorough statistical analysis. The glass-box parameter-based models may operate off line or on line according to the availability of characterization parameters at a given measurement instant. The characterization parameters include noise, packet loss, coding scheme, one-way delay, and delay jitter. The glass-box parameter-based models are generally less accurate and coarser than black-box media-based ones.

A well-known offline glass-box parameter-based model, named E-Model, has been defined by the ITU-T in Rec. G.107. E-Model aims at estimating QoE of voice calls transmitted over a planned transport infrastructure (*The E-Model, A Computational Model for Use in Transmission Planning*, 2007). The prevailing version of E-Model includes 21 basic characterization parameters. E-Model provides a single scalar value referred to as rating factor R that lies between 0 (worst quality) and 100 (excellent quality). In practice, any designed transport configurations that result in a rating factor below 60 should be avoided. In such a case, adequate actions should be undertaken to enhance QoE of voice calls. The basic characterization parameters are classified into simultaneous, equipment, and delay impairment factors, denoted respectively as I_s, I_e, and I_d. I_s quantifies the impairments that depends on characteristics of voice signals, such as quantification and compression. I_e quantifies the impairments caused by equipments, such as packet loss or interruption. I_d quantifies the impairment caused by delays and echoes. ITU-T Rec. G.107 gives the range values of each basic parameter and the mathematical expressions, which enable computing the value of each impairment factor. For simplification reasons, E-Model assumes that the perceived effects of impairment factors are

additive on a psychological scale. Thus, the final rating score R is given by, $R = R_0 - I_s - I_e - I_d$, where R_0 refers to user satisfaction under no-distortion condition.

The offline glass-box parameter-based quality models are suitable for planning purposes. They enable a general overview pf QoE values of a voice transmission system at an early phase. However, for service monitoring and management, online models are needed. In such a case, the variable model parameters should be acquired at run time. This is especially suitable for IP-based services where control data, such as sequence number and time stamp, are included in each packet header. In such an environment, it is possible to extract static characterization parameters from signaling messages and variable ones from the received packets captured at the destination port. This means that parameters are acquired without acceding to the media content, which is preferable for privacy reasons. This class of models will be considered in more detail in Section 12.2.

Gray-Box QoS/QoE Mapping Models

The gray-box quality models combine advantages of black- and glass-box mapping models. They sample basic characterization parameters at system output in addition to some control data describing the structure of clean stimulus (see Figure 12.2). The control data may be sent in separate control packets or piggybacked inside transmitted media packets. Hence, perceptually important information about a given content can be considered by the quality models. Therefore, they can measure QoE value on per-content basis. Given its simplicity to deploy and its reasonable accuracy, this class of QoS/QoE mapping models is quickly proliferating.

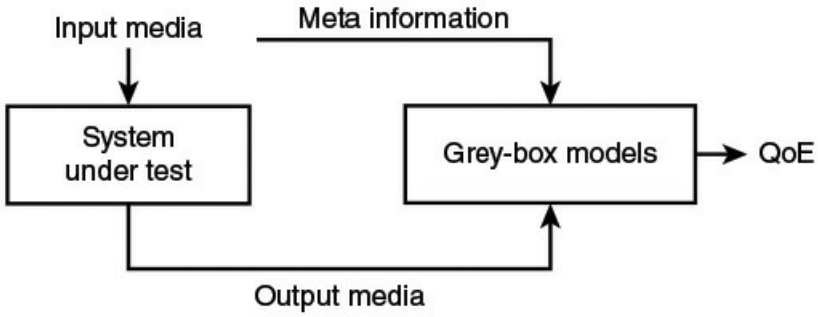


FIGURE 12.2 Gray-Box QoS/QoE Mapping Models

Typically, large telecom operators, such as Ericsson, Deutsch Telekom, and British Telecom, develop their propriety implementations of QoS/QoE mapping models and their companion software tools to acquire, record, and analyze measures that satisfy their specific needs. However, the majority of telecom operators delegate the task of assessment of their transport infrastructure, services, and equipments to specialized corporations, such as GL, OPTICOM, Telchemy, and HEAD Acoustics. In reality, QoS/QoE mapping models should be maintained and evolved to account for a new technology or usage context.

Tips for QoS/QoE Mapping Model Selection

The following checklist of five items can aid in the selection of a QoS/QoE mapping model:

- Which types of operations am I considering?
- Which parameters do I have? Can I access the signals, the contents, the packet payload or the header?
- Do I expect specifications and usage conditions to use a given mapping model?
- How much precision do I need?
- Do I have all inputs available for selected mapping models?

12.2 IP-Oriented Parameter-Based QoS/QoE Mapping Models

The area of measuring QoE of IP networks and applications is still in its infancy. However, the popularity of multimedia and user-friendly IP-based services puts QoE at the center of interest of today's ecosystem. In contrast to legacy content-oriented telecom systems (for example, public switched telephone network [PSTN], radio, and TV), IP networks carry clean media content from a server to a destination using a flow of media packets composed of a header and a payload. Therefore, parameters gathered at network layer, in addition to application layer, are easily accessible at run time on user devices. This enables measuring QoE at run time using online glass- or gray-box parameter-based quality models. The QoE over IP-based networks are time-varying in contrast to telecom networks, which are roughly time-invariant. This characteristic leads to considering instantaneous and overall QoE. The former refers to the observed QoE over a short time interval, on the order of 8 to 20 seconds. The latter refers to the overall QoE observed throughout a whole session in order of 1 to 3 minutes. The next sections give examples of online glass-box parameter-based quality models of IP-based video streaming applications.

Network Layer QoE/QoS Mapping Models for Video Services

The network layer QoS/QoE mapping models rely solely on NQoS metrics gathered from the TCP/IP stack except for the application layer (that is, transport, network, link, and physical layers). In a 2010 paper, Ketyko et al. proposed the following parameter-based quality model for estimating video streaming quality in 3G environment [KETY10]:

$$\overline{QoE} = 8.49 - 0.02 \cdot AL - 0.01 \cdot VL - 1.12 \cdot AJ + 0.04 \cdot RSSI \qquad \text{(Eqtn. 12.1)}$$

where AL and VL refer respectively to audio and video packet loss rates, AJ and VJ represent respectively audio and video packet jitter (VJ), and RSSI is the received signal strength indicator. A 2014 paper by Kim and Choi presented a two-stage QoE/QoS mapping model for IPTV over 3G networks [KIM14]. The first stage consists of combining a set of basic QoS parameters into one metric as follows:

$$QoS(L, U, J, D, B) = K\{W_L \cdot L + W_U \cdot U + W_J \cdot J + W_d \cdot D + W_b \cdot B\} \qquad \text{(Eqtn. 12.2)}$$

where L, U, J, D, and B refer, respectively, to packet loss, burst level, packet jitter, packet delay, and bandwidth. The constants K, W_I, W_u, W_J, W_d and W_b are predefined weighting coefficients, which depend on the type of the access network (that is, wired or wireless). The second stage consists of computing QoE value as following:

$$QoE(QoS(X)) = Q_r\,(1 - QoS\,(X))^{\,QoS(X)\times A/R} \qquad \text{(Eqtn. 12.3)}$$

where, X is a vector of parameters {L, U, J, D, B} and Q_r is a scalar limiting the range of the IPTV QoE obtained as a function of the display size/resolution of the screen. The constant A expresses the subscribed service class and R is a constant reflecting the structure of the video frames.

Application Layer QoE/QoS Mapping Models for Video Services

Besides NQoS parameters, application layer QoE/QoS mapping models use metrics gathered at application layers (AQoS). Moreover, they can account for the user behavior while interacting with a given video content. In a 2014 paper by Ma et al., the following parameter-based quality model is presented for video streaming application [MA14]:

$$QoE = 4023 - 0.0672\,L_x - 0.742\,(N_{QS} + N_{RE}) - 0.106\,T_{mr} \qquad \text{(Eqtn. 12.4)}$$

where Lx refers to the start-up latency, that is, the waiting time before playing a video sequence, N_{QS} is the number of quality switches that count the number of times the video bit rate is changed during a session, N_{RE} is the number of rebuffering events, and T_{MR} is the mean rebuffering time. The following parameter-based quality models, reported in a 2009 paper by Khan et al., estimate QoE of a generic streamed content video over wireless networks using MPEG4 codec [KHAN14]:

$$QoE(FR, SBR, PER) = \frac{a_1 + a_2\ FR + a_3 \cdot \ln(SBR)}{1 + a_4 \cdot PER + a_5 \cdot (PER)^2} \qquad \text{(Eqtn. 12.5)}$$

where FR, SBR, and PER refer, respectively, to the frame rate sampled at the application level, sent bit rate, and packet error rate sampled at the network level. The coefficients a_1 to a_5 are used to calibrate the quality model. This model has been updated to account for three types of video content: slight movement, gentle walking, and rapid movement. The quality model is given by the following:

$$QoE(FR, SBR, BLER, CT) = a + \frac{b \cdot e^{FR} + c \cdot \ln(SBR) + CT \cdot (d + e \cdot \ln(SBR))}{1 + f \cdot (BLER) + g \cdot (BLER)^2} \qquad \text{(Eqtn. 12.6)}$$

where a, b, c, d, e, f, g represent constants; CT is the content type of the video; and SBR and BLER refer respectively to the sent bit rate and the bit loss error rate. This model was developed for video streaming service transmitted over UMTS networks using H.264 video codec.

A QoE /QoS mapping model for IPTV was developed by Kuipers et al. [KUIP10], which accounts for the start-up latency and zapping time. The latter is defined as the change frequency of TV channels. The quality model in given by the following

$$QoE_{zapping} = a \cdot \ln(ZT) + b \qquad \text{(Eqtn. 12.7)}$$

where $QoE_{zapping}$ is a one-dimension QoE component considering zapping behaviour, ZT is the zapping time expressed in seconds, and a and b are numeric constants that might be positive or negative. Finally, the following parameter-based quality models proposed by Hossfeld et al. [HOSS13] accounts for stalling events, which are defined as involuntary pauses in the rendered

video streams. The quality model is given by the following:

$$QoE = 3.5e^{-(0.15L+0.19)N} + 1.5 \quad \text{(Eqtn. 12.8)}$$

where L refers to average stalling duration and N is the number of stalling events.

12.3 Actionable QoE over IP-Based Networks

This section introduces **actionable QoE,** which refers to all techniques and mechanisms enabling to concretely measure and utilize QoE metrics. Actionable QoE goes beyond QoE definition and measurement toward QoE exploitation. An actionable QoE solution strongly depends on the underlying system and services characteristics. Moreover, actionable QoE solution works over multiplane architectures that integrate data, control, and management planes. Basically, two solutions may be used to achieve actionable QoE:

- System-oriented actionable QoE solution
- Service-oriented actionable QoE solution

The System-Oriented Actionable QoE Solution

The system-oriented actionable QoE solutions account for QoE measures within the delivery infrastructure. In such a condition, services are engineered while assuming that underlying system is perfect; that is, no degradations are inserted. Figure 12.3 illustrates a nominal environment where system-oriented actionable QoE solution may be provided. As can be seen, actionable QoE solution requires (1) a QoS measurement module that gathers basic **key performance indicators** (KPIs) from the underlying system, (2) a QoE/QoS mapping model, and (3) a resource management module of controlled devices. Each service provider specifies a target QoE level that should be offered for its customers. The QoE/QoS mapping model should be selected in a way that guarantees (a) the availability of quality model input parameters and (b) conformity with service specifications and conditions. A signaling procedure may be executed to do that. The management procedure may be executed either before starting a service or during its delivery. It involves specifications of all configurable parameters provided by a given infrastructure, such as priority, marking threshold, traffic shaping, and so on. This should be realized using an autonomous decision system, including a policy that maps observed QoE measures to a course of actions executed by managed devices.

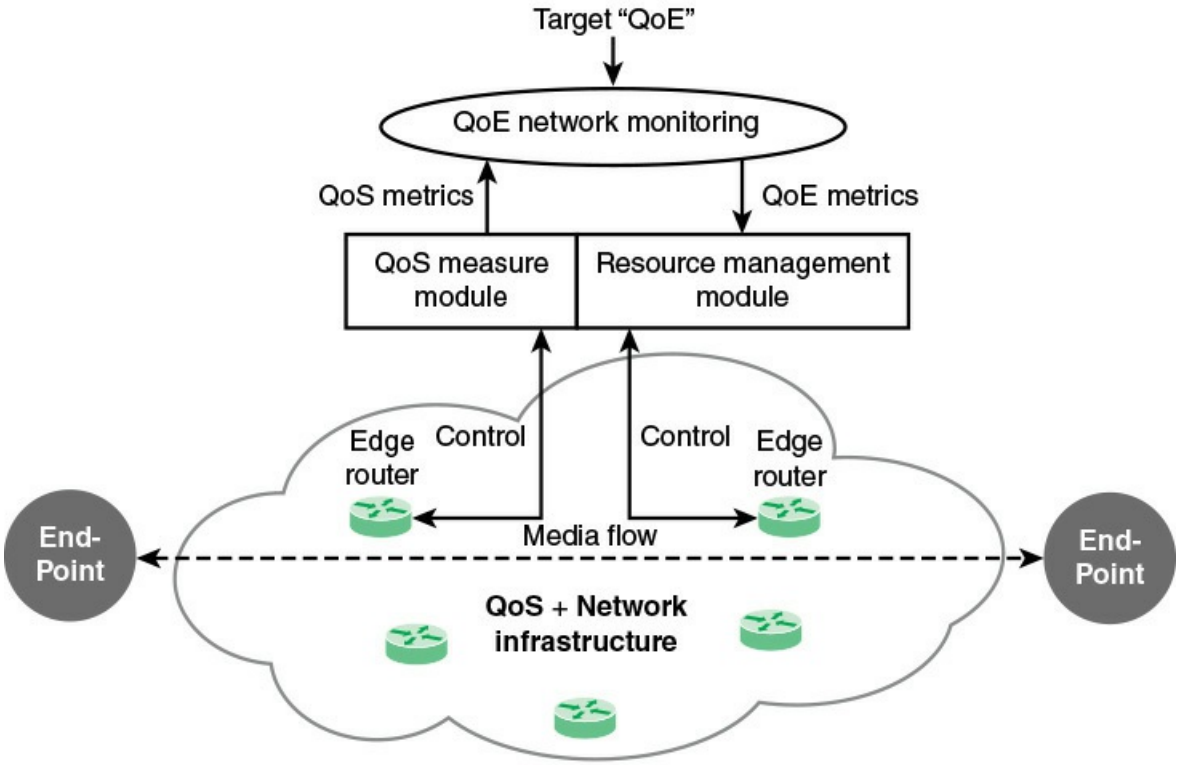


FIGURE 12.3 A Nominal Environment for Providing QoE-Centric Services

This operational mode applies well for software-defined networking (SDN) where the network paths are managed by an SDN controller. In such a case, the measured QoE values are reported to the SDN controller, which uses them to define the behavior of SDN switches. The SDN controller should include a QoE policy and rules module that (1) checks whether the contracted QoE level is respected on a per-user/per-flow basis and (2) specifies SDN paths that should be used to forward users flows. The QoE policy and rules module should consider situations where services cross SDN-supported and -unsupported realms.

The Service-Oriented Actionable QoE Solution

The service-oriented actionable QoE solutions account for QoE values measured at endpoints and service level (see Figure 12.4). In such a situation, services are engineered to deal with the underlying system flaws to reach a specified QoE level. The services may change their behavior as a function of the current context and condition. The measurement module of KPI is installed on endpoints. The QoE/QoS mapping models may be deployed either on endpoints or specialized devices. The measured QoE values are sent to endpoints to configure different application modules at sender, proxy, and receiver entities.

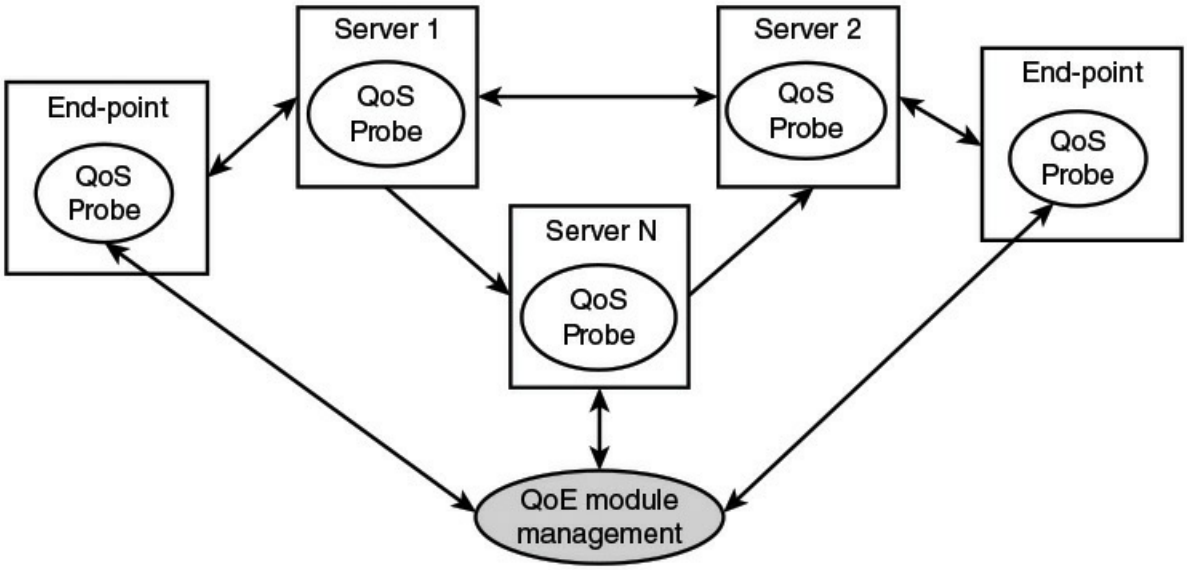


FIGURE 12.4 Service-Aware QoE Deployment Scheme

The service-oriented actionable QoE solution involves multiple advantages. First, per-service, per-user, and per-content QoE monitoring and management solutions are performed to provide a given QoE level. Second, it provides more adaptation possibility because it precisely discerns capability and the role of each service component. Third, it reduces the communication overhead and balances computing loads. Finally, it enables component-level granularity treatment of QoE in addition to stream- and packet-level granularities. However, this solution cannot be applied to already running services and results in higher complexity in service design and engineering.

12.4 QoE Versus QoS Service Monitoring

Monitoring is a strategic function that should be supported by today's IT system. It returns indicators and provides clues regarding the system performance and its workload. Moreover, it enables detecting system dysfunction and defects as well as underperforming devices and applications so that the best course of actions may be undertaken. The monitoring solutions of current IT systems may be classified into the following four categories (see Figure 12-5):

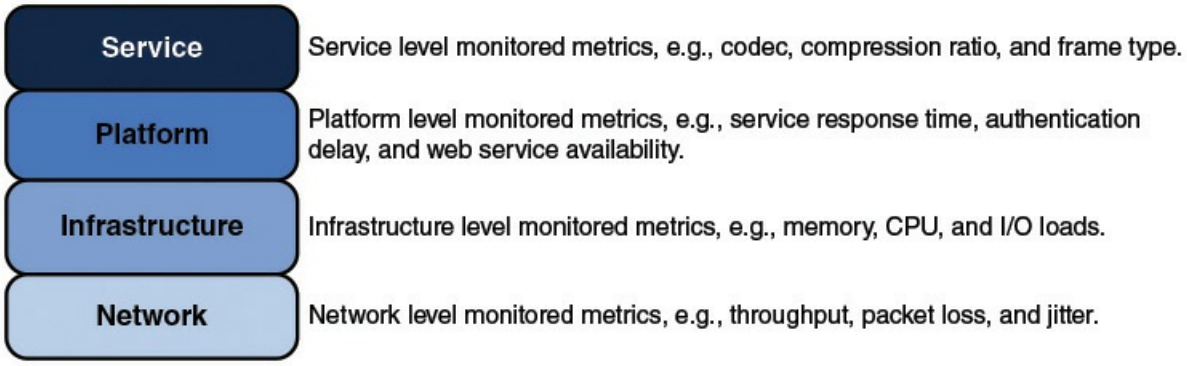


FIGURE 12.5 A Classification of Monitoring Solutions

- **Network monitoring:** Provides measures about performance of paths and links used to deliver media units. They are collected at packet processing devices (router and switch) and may operate on per-flow or per-packet bases. The path characterization metrics, such as throughput, packet loss, reorder and duplication, delay, and jitter, are calculated using atomic metrics extracted from packet header, such as sequence number and time stamps.
- **Infrastructure monitoring:** Provides measures about devices performance and resources state, such as memory, CPU, IO, load, and so on.
- **Platform monitoring:** Provides performance indicators about the computing center where back-end servers are running. They may work over a virtualized infrastructure where business application logics are deployed using virtual machines.
- **Service monitoring:** Provides measures about services performance. The metrics are dependent on each application, and may be realized from technical or perceptual perspectives.

Typically, the monitoring solution in a distributed system involves a variety of **probes** that measure the performance of a given element participating in the service delivery chain. They are distributed and deployed over the system following a particular policy. Moreover, it includes a reliable and scalable manager that remotely configures probe behavior, especially in terms of the frequency and content of measurement reports. Often, probes are built in a given managed device or component (for example, SNMP agent). They may be configured by the network or system administrators to fit specific environment requirements. Typically, probes send atomic and elementary metrics that are transformed by the manager into human-friendly metrics. The manager logs all measures following a specific representation and saves them at a specific location.

The monitoring solution should provide communication facility between the manager and the managed devices. The interaction between the manager and managed entities is conventionally realized using the connectionless User Datagram Protocol (UDP) through a couple of reserved ports. This has been evolved to work as a short-lived HTTP connection. The probes are defined as a RESTful service that may be called by the manager. Moreover, exchanged reports can be realized either in band or out of band. The first strategy shares the resource used for data delivery, but the second one uses a dedicated and autonomous devices and channels to perform monitoring tasks.

Figure 12.6 presents a typical configuration of an on-demand monitoring solution. The manager receives monitoring requests from customers expressed using specific syntax. Upon the receipt of a new monitoring request, the manager inquires with a Universal Description, Discovery, and Integration (UDDI) directory to get more information about the monitored service, such as location and properties. The monitoring solution, including a set of probes, is deployed over a given infrastructure. They register themselves automatically once a monitored component is activated in a preconfigured registrar. The registrar keeps traces and features of all active probes. They should be configured off line by the administrator to report metrics according to a given behavior during a service. The metrics generated by the probes may be aggregated and processed before sending them to the manager that performs data analytic procedure.

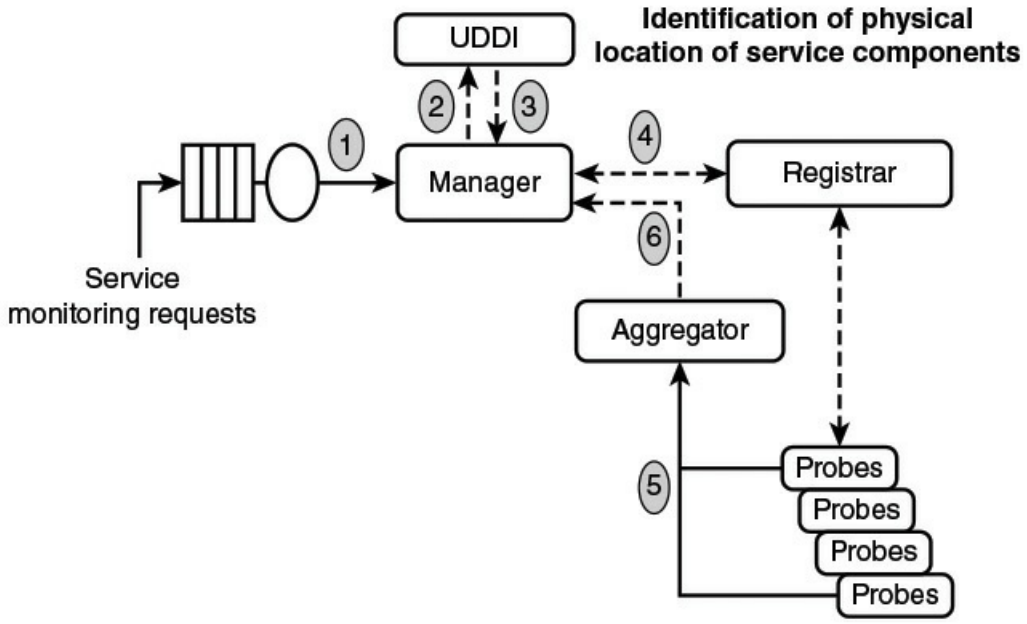


FIGURE 12.6 A Baseline and Generic Monitoring Solution

The traditional QoS metrics may be measured at network, infrastructure, platform, and service layers. However, QoE metrics may be only measured at the service layer, where it is possible to interact with final users. The next sections present recent technologies and trends in QoS and QoE monitoring solutions.

QoS Monitoring Solutions

The emerging QoS monitoring solutions are basically developed for data centers and clouds where virtualization technology is supported. Figure 12.7 shows a network- and infrastructure-level monitoring solution built for cloud-based IPTV service. The audiovisual content servers are placed on a cloud. The traffic sent from the content servers to IPTV devices is permanently monitored through a set of Vprobes deployed across the network. A Vprobe is an open-ended investigatory tool that is used in the cloud environment to inspect, record, and compute the state of the hypervisor as well as each virtual machine running service business logics. The flows of video packets are parsed at different measurement points. The information collected by Vprobes is used next to reconstruct service-level detailed records (SDRs). Each record contains the most relevant information of the complete session between an origin (server) and a destination (user). The critical parameters of the messages associated with an IPTV session are stored inside the SDRs.

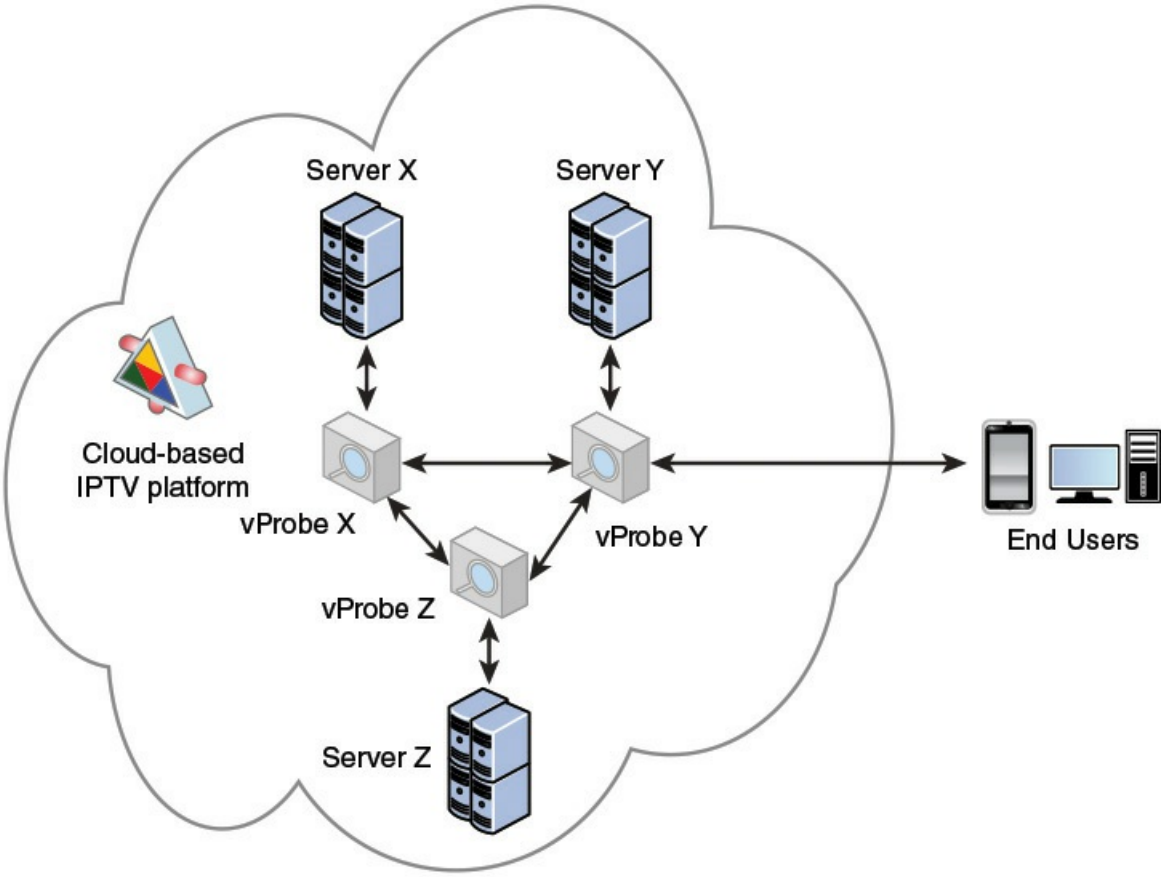


FIGURE 12.7 vProbes Approach in Cloud-Based IPTV Network

Amazon developed CloudWatch, which is a tied monitoring solution over Amazon cloud (http://aws.amazon.com/cloudwatch/). It can monitor cloud resources, such as CPU and memory utilization, in addition to client applications and services. The administrators can collect and track customized metrics that are dependent on each application. Amazon CloudWatch retrieves the monitored data, displays graphs, and sets alarms to help in resolving troubleshooting, performing spot trends, and taking automated courses of action based on the state of the cloud.

QoE Monitoring Solutions

The emerging QoE monitoring solutions extend and adapt QoS ones. As discussed before, the QoE monitoring solution strongly depends on QoE/QoS mapping models. Moreover, there is no one-size-fits-all solution for QoE monitoring, in opposition with QoS monitoring solutions.

The diagrams in Figure 12.8 show four configurations that can be used to monitor at run time the QoE values of IP-based video streaming services. The configurations differ in term of the measurement and mapping model locations. Each configuration is denoted using XY expression, where X refers to the measurement location and Y refers to the quality model location. They may take one of these values: N for network, C for client, and B for both:

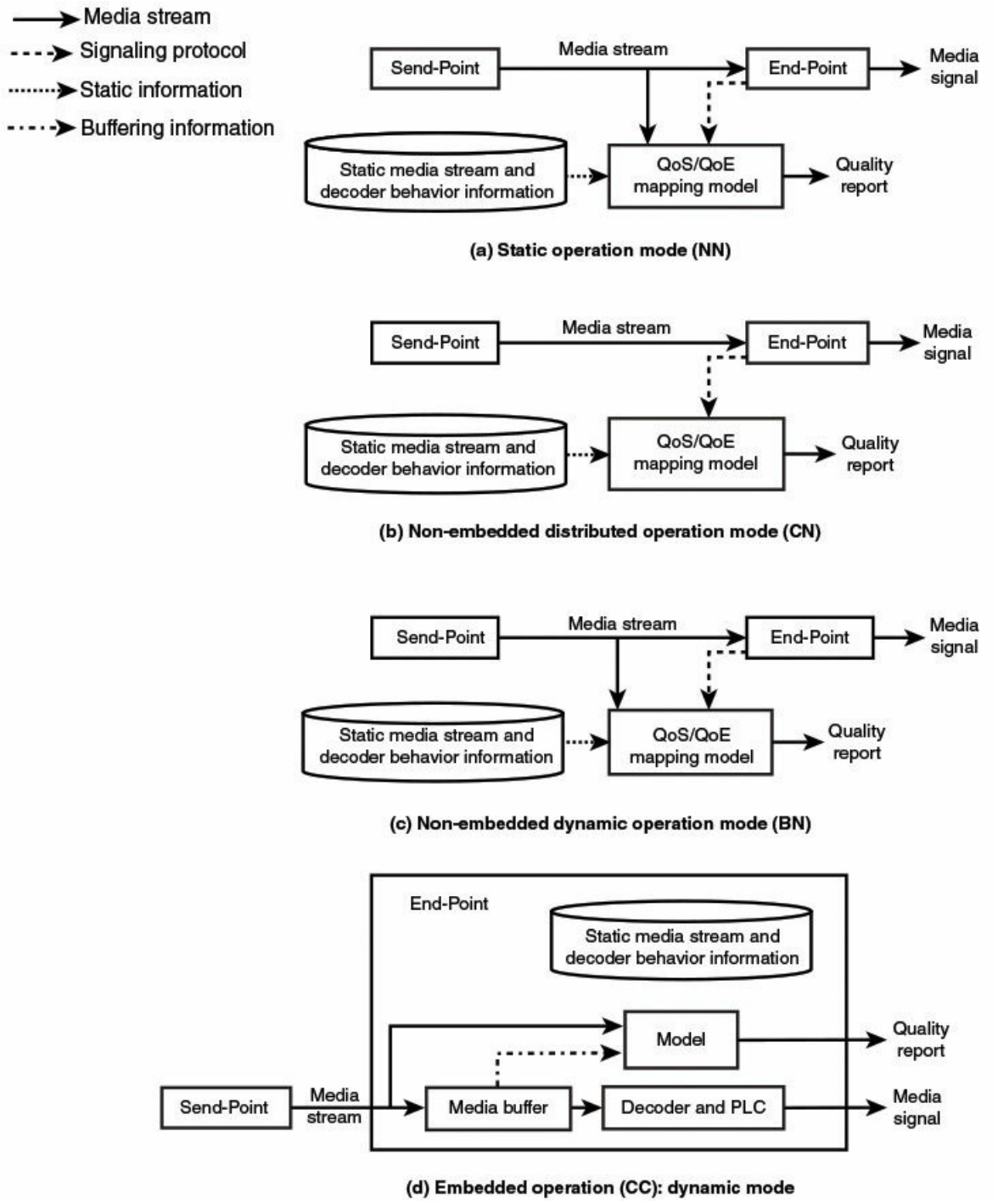


FIGURE 12.8 The Operational Working Modes of Quality Models in Networks

A. Static operation mode (NN): Both the measurements of KPIs and QoE are performed inside the network. The QoE/QoS mapping model is installed on a device listening to the service delivery path (see part a of Figure 12.8). The quality model uses collected KPIs, prior knowledge about video coding schemes, and endpoint characteristics. The parameters of QoE/QoS mapping models are extracted from decrypted information included in the transmitted media packets. The characteristics of endpoints can be

acquired either by polling them or by inspecting exchanged SDP (Session Description Protocol) messages. The QoE measurement points may include an endpoint emulator enabling a realistic reconstruction of received streams.

B. Nonembedded dynamic operation (BN): The measurement of KPIs is performed at both the network and the client, whereas QoE values are measured inside the network (see part c of Figure 12.8). The quality model uses gathered KPIs, prior knowledge about coding schemes, and information about the client obtained using a customized signaling protocols.

C. Nonembedded distributed operation (CN): The measurement of KPIs is performed at the client side, and these are sent periodically to the QoE/QoS mapping model located inside the network (see part b of Figure 12.8).

D. Operation embedded (CC): The measurement of KPIs and QoE is performed at the client side. The QoS/QoE mapping model is embedded inside the client (see part d of Figure 12.8). The measured QoE metrics may be reported to a centralized monitoring entity.

A standardized multidimensional QoE monitoring solution is defined in ETSI Technical Specification TS 103 294 (Speech and Multimedia Transmission Quality (STQ); Quality of Experience; A Monitoring Architecture, 2014). This solution used QoE agents deployed over devices that communicate with each other and with data-acquisition objects (or probes). The architecture of QoE-agent is based on a layered definition of APIs that enable convenient grouping of different factors that influence QoE. The six layers are defined as follows:

- **Resource:** Composed of dimensions representing the characteristics and performance of the technical system(s) and network resources used to deliver the service. Examples of such factors include network QoS in terms of delay, jitter, loss, error rate, and throughput. Furthermore, system resources such as server processing capabilities and end user device capabilities (e.g. computational power, memory, screen resolution, user interface, battery lifetime, etc.) are included.
- **Application:** Composed of dimensions representing application/service configuration factors. Examples of such factors include media encoding, resolution, sample rate, frame rate, buffer sizes, SNR, etc. Content-related factors (e.g., specific temporal or spatial requirements, 2D/3D content, and color depth) also belong to this space.
- **Interface:** Represents the physical equipment and interface through which the user is interacting with the application (type of device, screen size, mouse, etc.).
- **Context:** Related to the physical context (e. g. geographical aspects, ambient light and noise, time of the day), the usage context (e.g. mobility/no-mobility or stress/no-stress), and the economic context (e.g. the cost that a user is paying for a service).
- **Human:** Represents all factors related to the perceptual characteristics of users (e.g. sensitivity to audio-visual stimulus, perception of durations, etc.).
- **User:** Users' factors that are not represented in the Human layer. These factors encompass all aspects of humans as users of services or applications (e.g., history and social characteristics, motivation, expectation, and level of expertise).

The strength of the QoE monitoring solution using these layers resides in its ability to monitor

any service using customized QoS/QoE mapping models. The QoE agent is composed of the six major objects illustrated in Figure 12.9 and described in the list that follows.

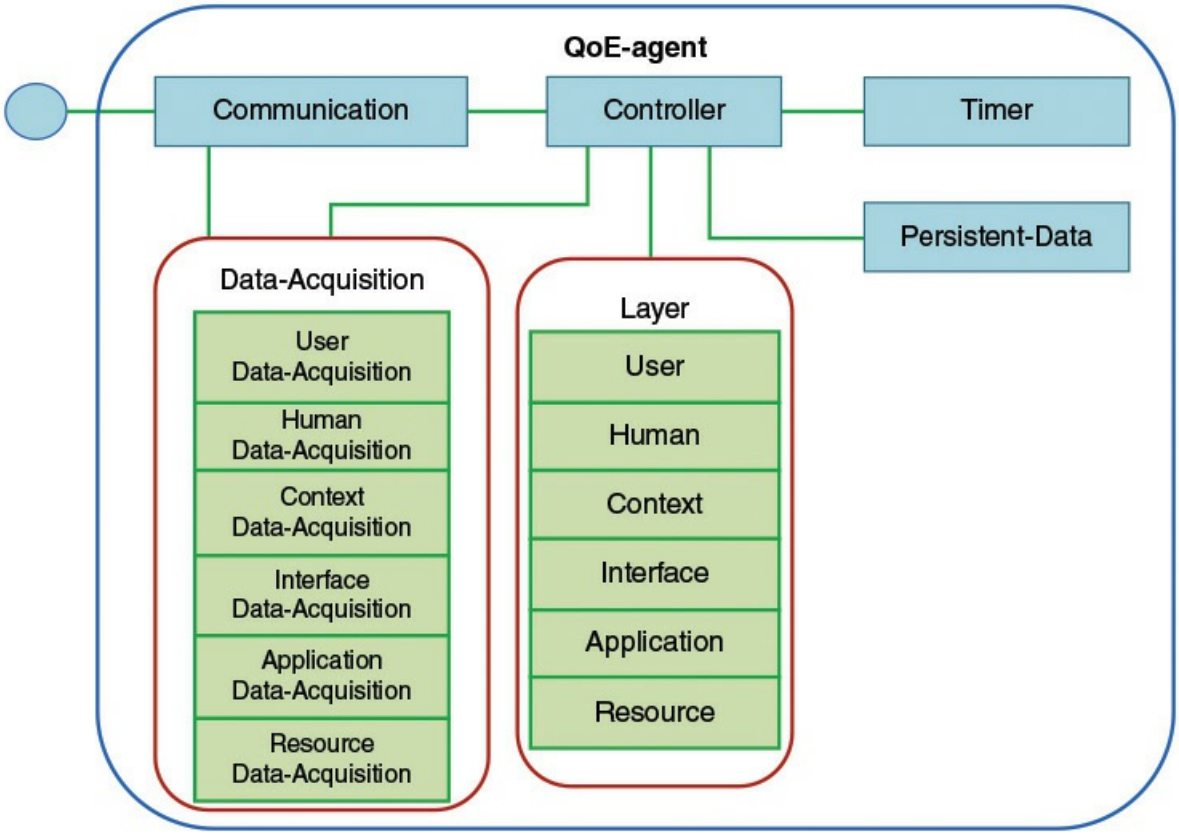


FIGURE 12.9 A Maximum Co-Located QoE Agent (Generic QoE Agent)

- The **Communication** object manages inter-QoE agent communications.
- The **Data-Acquisition** object implements all data-acquisition sublayers. It acquires atomic information necessary for the calculation of the internal parameters of a given QoE/QoS mapping model.
- The **Controller** object implements global QoE/QoS mapping models and handles external requests and commands, such as get and set operations.
- The **Layer** object is an interface object that implements different model layers, such as Application model, Context model, and User model.
- The **Persistent-Data** object stores the quality parameters for all layers.
- The **Timer** object is used as the internal time for the QoE agent.

The QoE agent must implement all layers of the ARCU model. However, they may be distributed over many physical devices. To do that, two types of QoE agents are introduced:

- A **master QoE agent** is a nondistributed entity implementing at least the User model sublayer, which contains at least a Layer object of type User. It must implement a Communication object, a Controller object, a Timer object, and a Persistent-Data object. It must also implement a Data-Acquisition object (see Figure 12.10).

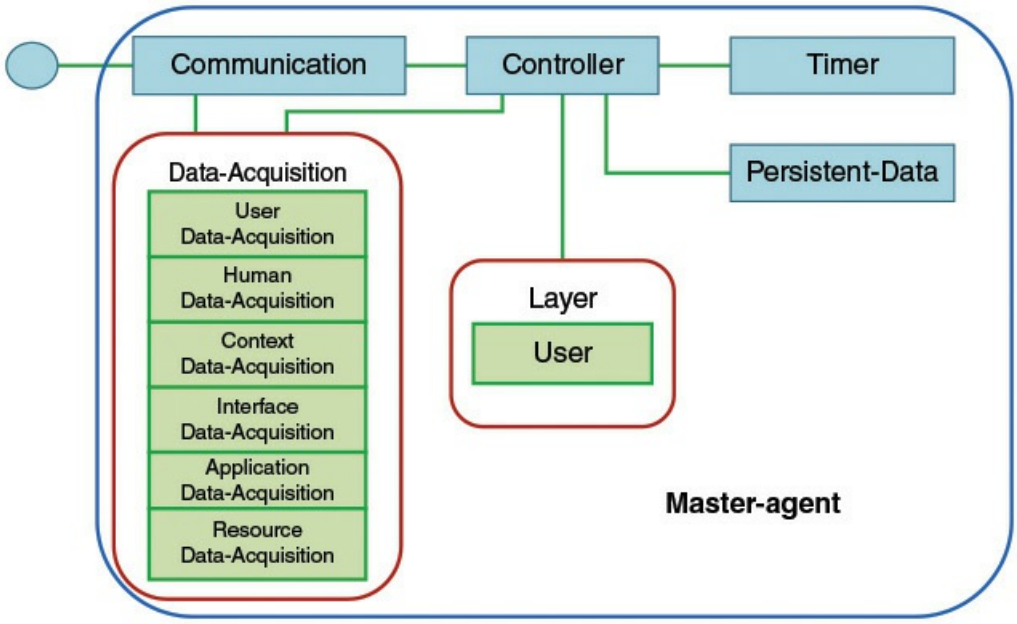


FIGURE 12.10 A Minimum Master Agent (With Only User Model)

- A **slave QoE Agent** is a nondistributed entity implementing a Data-Acquisition object / some Layer object, but no Layer object of type User. It must also implement a Communication object, a Controller object, and a Timer object.

The components of maximum and minimum master QoE agents are, respectively, illustrated in Figure 12.9 and Figure 12.10. The maximum master QoE agent includes all ACRU layers, whereas the minimum master QoE agent implements only the user layer. Figure 12.11 illustrates a slave QoE agent that implements only one layer of ACRU model other than the user layer.

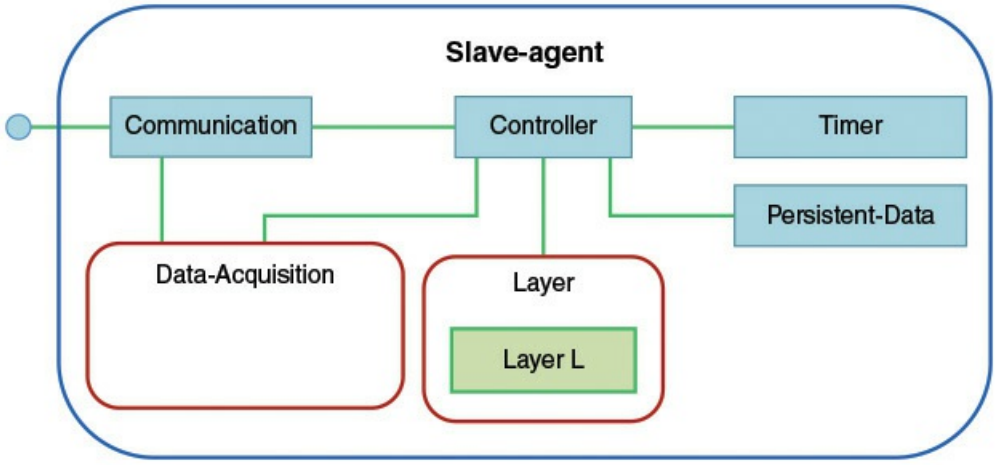


FIGURE 12.11 A Slave QoE Agent Implementing Only One Layer Model (Besides the User Layer)

The data-acquisition module is encapsulated into a probe agent specified as follows:

- **A probe agent of type L:** A nondistributed entity implementing the data-acquisition

sublayer of type L and no Layer objects (see Figure 12.12). It must also implement a Communication object, a Controller object, and a Timer object.

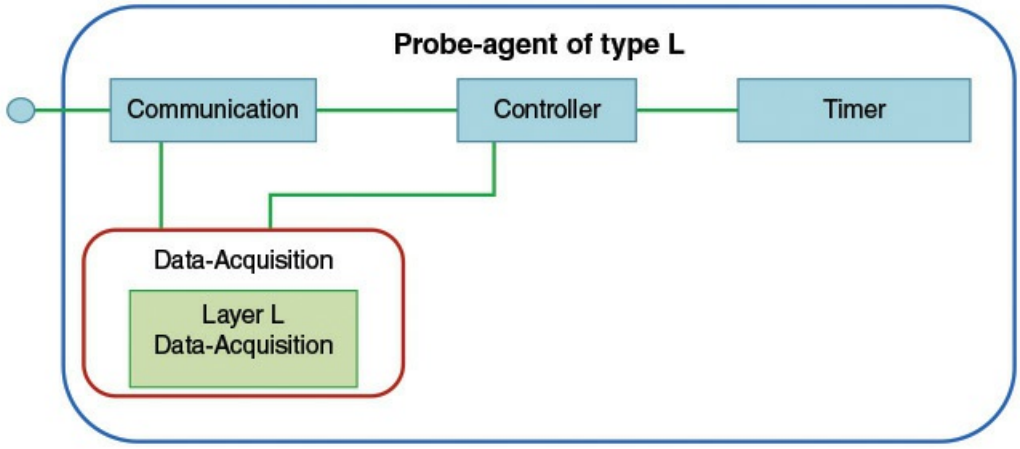


FIGURE 12.12 A Probe Agent of Type L

- **A probe agent** is used when the type of the probe agent of type L is irrelevant or when the entity implements several sublayers of different types. It must also implement a Communication object, a Controller object, and a Timer object.

12.5 QoE-Based Network and Service Management

The quantified QoE values may be considered in networks and services management. This enables getting an optimal trade-off that maximizes QoE and minimizes consumption of resources. The major challenge resides in the translation of QoE metrics into a course of actions that definitely enhance encountered QoE and reduce resources consumptions. Unfortunately, there is no systematic approach for reaching such a goal. The following sections describe a number of applications that seek to undertake a set of actions as a function of measured QoE.

QoE-Based Management of VoIP Calls

The management of Voice over IP (VoIP) based on QoE has been extensively investigated in the literature. The goal is to maintain a constant QoE level during a whole packet voice session transmitted over time-varying quality IP networks. Typically, QoE measurement probes following one glass-box parameter-based model are installed on VoIP endpoints. They collect at run time atomic KPIs, which are transformed and given as inputs to a QoE/QoS mapping model. After a new measure of QoE values is received, a QoS controller adjusts the reconfigurable network parameters within a delivery path, such as queuing allocation and congestion thresholds. A simple policy consists of allocating more (respectively less) network resource if the QoE value is less (greater) than a targeted QoE value.

QoE-Based Host-Centric Vertical Handover

Mobile consumers over next-generation networks could be served at one moment by several overlapping heterogeneous wireless networks. In such a case, mobile users should choose the access network that will likely achieve good quality. The network selection/switching procedure can be performed either at the start or during the service. An internetwork hard handover occurs when users switch from one network to another because of specific reasons related to both consumers and providers. A handover could be managed in network- or host-centric way. In a traditional network-centric approach, the infrastructure monitored by providers decides when a handover is required through a set of control algorithms. In a host-centric approach, however, end nodes can perform a handover when quality of service becomes unsteady and unsatisfactory.

Figure 12.13 illustrates a likely envisaged scenario where the client could be served either by WiMAX or Wi-Fi systems. Appropriate equipment should be deployed and configured, such as outdoor and indoor units, server, router, and Wi-Fi and WiMAX access points to enable network handover. Throughout a vocal call, the client may switch from WiMAX system to Wi-Fi system, and vice versa.

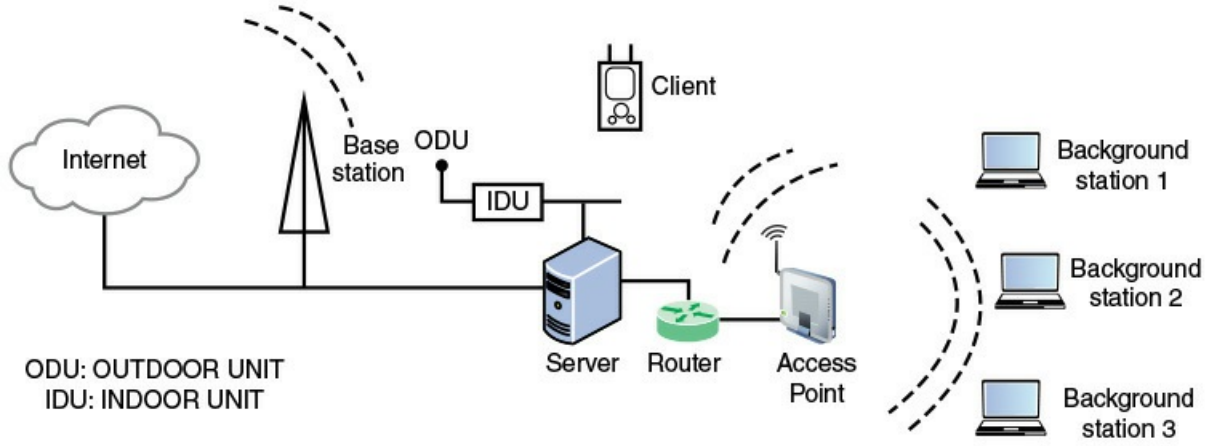


FIGURE 12.13 Network Selection Between Wi-Fi and WiMAX Based on Client and Link Quality [MURP007]

Murphy et al. argue that a host-centric network selection approach is more suitable to support delay-sensitive services [MURP07]. Indeed, in such a case, internetwork handover may be performed according to customized requirements specified by on a per-service and per-user basis. In general, for delay-sensitive services, a seamless network switching that reduces/cancels service interruptions network selection controller may be used.

To do that, it is possible to use message-based, multistreamed, multihomed, and reliable Stream Control Transmission Protocol (SCTP) transport protocol. In contrast to TCP, SCTP allows delivering out-of-order packets to applications, which is more suitable for delay-sensitive applications. The multihoming feature of SCTP enables a transparent handover over several heterogeneous overlapping wireless networks. One path with specified destination and source addresses plays the role of primary path. The remaining ones play the role of secondary paths. SCTP can monitor at run time delay and jitter on all active paths, and makes them available to the application. Heartbeat messages are sent over secondary paths to collect required measurements. The collected KPIs are mapped to QoE values using a suitable QoE/QoS mapping model. The path quality is compared at a regular interval, and the client decides whether a network switch is necessary according to its customized and internal policy.

QoE-Based Network-Centric Vertical Handover

This section covers a QoE-based network-centric internetwork handover scheme. The goal is to perform a handover between overlapping WLAN and GSM networks. This allows, on one hand, relatively exploiting the high capacity of a WLAN, and on the other hand, reducing the GSM network load and cost. Figure 12.14 shows a scenario where a mobile subscriber initiates a voice call to a landline PSTN subscriber using a WLAN as a last-wireless hop. Next, when the QoE of the voice call goes below a given critical threshold because of mobility or congestion, a handover is performed. In such a case, the mobile subscriber is linked to the landline subscriber using the GSM infrastructure. The hands-free terminal is equipped with two wireless card interfaces to allow connection to WLAN and GSM networks. The mobile terminal sends adequate "quality reports" to a PBX that analyzes received feedbacks. Once an unsatisfied score is detected, the PBX instructs the mobile terminal to perform a handover. To do that seamlessly, a voice channel is opened using GSM infrastructure between the mobile terminal and PBX, which is responsible to relay received voice information toward the fixed subscriber.

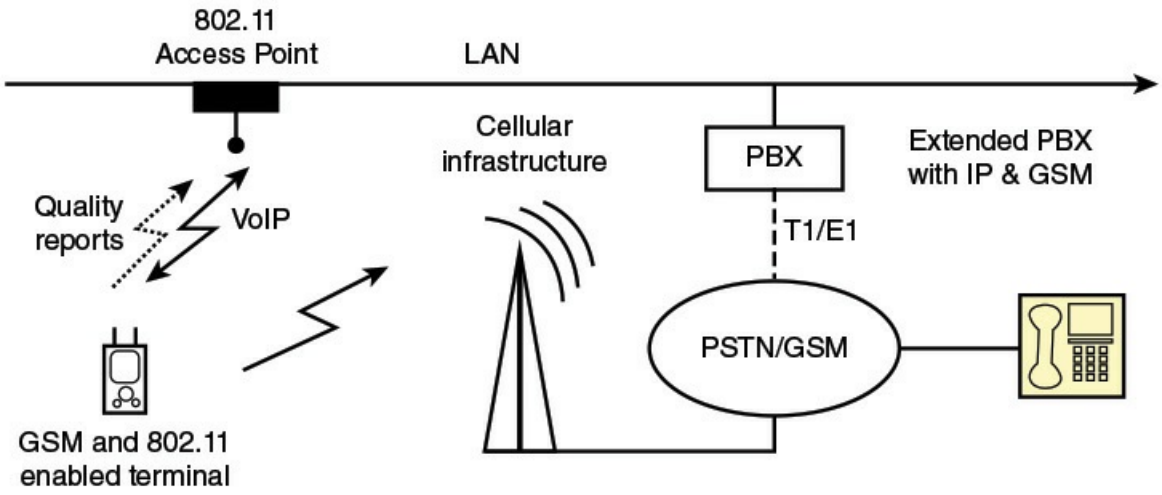


FIGURE 12.14 Handover Scenario Between WLAN and GSM Networks [MAES06]

The handovers are controlled using the following simplified additive quality models, installed at the private branch exchange (PBX), as reported by Marsh et al. [MARS06]:

$$\text{Handover score} = \text{Signal} + \text{Loss} + \text{Jitter} + \text{Report loss} \qquad \text{(Eqtn. 12.9)}$$

where, the handover scores vary from –100 to 100. The remaining variables are defined as follows:

- **Received signal strength indicator:** The signal-to-noise ratio is a good indicator about quality of service, especially over wireless Telecom networks. This metric may entail inaccurate estimates about quality over wireless data networks. In fact, over wireless telecom networks, high signal strength indicates that users sustain potentially a good QoE. This rule is inaccurate over wireless data networks where QoE could be poor in spite of high measured signal strength because of, for example, congestion-induced packet losses. The mobile terminal periodically records the received signal strength. The obtained value is scaled according to handover score defined by authors. Specifically, the measured

received signal strengths are mapped to values varying from 0 to +90.

- **Delay jitter:** An increasing delay jitter is a good indicator of poor quality. According to a preliminary empirical study, a score of +10 and 0 is assigned to good and negligible jitter conditions, respectively. A score of –10 and –20 is assigned to poor and very poor jitter conditions, respectively.
- **Packet loss:** High packet loss rate indicates that users sustain undoubtedly a very poor quality. A decreasing score step of –10 is assigned to encountered packet loss ratio with an increasing step of 8 percent. A long bad period is accounted for by increasing properly the contribution of packet loss.
- **RTCP losses:** The mobile terminal will likely sustain reception problems when the monitoring node does not receive RTCP quality reports. Three or more consecutive losses of RTCP feedback are generally quite significant to reduce aggressively the overall handover score. A decreasing score step of –10 is assigned to each consecutively lost RTCP report.

A large positive score indicates a good QoE. The mobile users are allowed to specify the lower acceptable threshold score. As a consequence, a handover is performed only when the calculated handover score falls below the defined threshold. An increasing threshold results in the improvement of average quality at the expense of a higher communication cost, because the system will switch the voice session to GSM system earlier. Conversely, a decreasing threshold results in the reduction of communication cost at the expense of longer periods of degraded quality.

12.6 Key Terms

After completing this chapter, you should be able to define the following terms.

QoE/QoS model mapping

black-box mapping model

glass-box mapping model

gray-box mapping model

QoE-aware services

QoE-based monitoring

QoE-based management

12.7 References

HOSS13: Hossfeld, T., et al. “Internet Video Delivery in YouTube: From Traffic Measurements to Quality of Experience.” Book chapter in *Data Traffic Monitoring and Analysis: From Measurement, Classification, and Anomaly Detection to Quality of Experience*, Lecture Notes in Computer Science, Volume 7754, 2013.

KETY10: Ketyko, I., De Moor, K., Joseph, W., and Martens, L. "Performing QoE-Measurements in an Actual 3G Network," IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, March 2010.

KHAN09: Khan, A., Sun, L., and Ifeachor, E. "Content Clustering Based Video Quality Prediction Model for MPEG4 Video Streaming over Wireless Networks," *IEEE International Conference on Communications*, 2009.

KIM14: Kim, H., and Choi, S. "QoE Assessment Model for Multimedia Streaming Services Using QoS Parameters," *Multimedia Tools and Applications*, October 2014.

KUIP10: Kuipers, F. et al. "Techniques for Measuring Quality of Experience," 8th International Conference on Wired/Wireless Internet Communications, 2010.

MA14: Ma, H., Seo, B., and Zimmermann, R. "Dynamic Scheduling on Video Transcoding for MPEG DASH in the Cloud Environment," Proceedings of the 5th ACM Multimedia Systems Conference, March 2014.

MARS06: Marsh, I., Grönvall, B., and Hammer, F. "The Design and Implementation of a Quality-Based Handover Trigger," 5th International IFIP-TC6 Networking Conference, Coimbra, Portugal.

MURP07: Murphy, L. et al. "An Application-Quality-Based Mobility Management Scheme," Proceedings of 9th IFIP/IEEE International Conference on Mobile and Wireless Communications Networks, 2007.

Part V: Modern Network Architecture: Clouds and Fog

We have discussed a new large communication system, one markedly different from the present in both concept and equipment, and one which will mean a merging of two different technologies: computers and communications.

—On Distributed Communication: Summary Overview, Rand Report RM-3767-PR, Paul Baran, August 1964

The two dominant modern network architectures are cloud computing and the Internet of Things (IoT), sometimes referred to as fog computing. The technologies and applications discussed in the preceding sections all provide a foundation for cloud computing and IoT. Chapter 13 is a survey of cloud computing. The chapter begins with a definition of basic concepts, and then covers cloud services, deployment models, and architecture. The chapter then discusses the relationship between cloud computing and software-defined networking (SDN) and network functions virtualization (NFV). Chapter 14 and Chapter 15 provide a detailed look at IoT, with Chapter 14 providing coverage of the principal components of IoT-enabled things, while Chapter 15 examines IoT reference architectures and looks at three implementation examples.

Chapter 13. Cloud Computing

> One can now picture a future investigator in his laboratory. His hands are free, and he is not anchored. As he moves about and observes, he photographs and comments. Time is automatically recorded to tie the two records together. If he goes into the field, he may be connected by radio to his recorder. As he ponders over his notes in the evening, he again talks his comments into the record. His typed record, as well as his photographs, may both be in miniature, so that he projects them for examination.
>
> —"As We May Think," Vannevar Bush, *The Atlantic*, July 1945

Chapter Objectives: **After studying this chapter, you should be able to**

- Present an overview of cloud computing concepts.
- List and define the principal cloud services.
- List and define the cloud deployment models.
- Compare and contrast the NIST and ITU-T cloud computing reference architectures.
- Discuss the relevance of SDN and NFV to cloud computing.

Section 1.6 provided a brief overview of the concept of cloud computing, and Section 2.2 included a discussion of the requirements that cloud computing generates with respect to networking. This chapter begins with a more detailed look at the basic concepts of cloud computing. Next is a discussion of the principal types of services typically offered by cloud providers. The chapter then looks at various deployment models for cloud systems, followed by an examination of two cloud computing reference architectures, developed by NIST and ITU-T, respectively. A consideration of these two different models provides insight into the nature of cloud computing. Finally, the chapter discusses how SDN and NFV can support cloud computing deployment and operation.

13.1 Basic Concepts

There is an increasingly prominent trend in many organizations to move a substantial portion or even all information technology (IT) operations to an Internet-connected infrastructure known as enterprise cloud computing. At the same time, individual users of PCs and mobile devices are relying more and more on cloud computing services to backup data, synch devices, and share. NIST defines cloud computing, in NIST SP-800-145, *The NIST Definition of Cloud Computing*, as follows:

← *See Section 1.6, "Cloud Computing"*

> **Cloud computing:** A model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (for example, networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes

availability and is composed of five essential characteristics, three service models, and four deployment models.

← *See Figure 1.7, Cloud Computing Context*

The definition refers to various models and characteristics, whose relationship is illustrated in Figure 13.1. The five essential characteristics were discussed in Chapter 1, "Elements of Modern Networking." This chapter covers the three service models and the four deployment models.

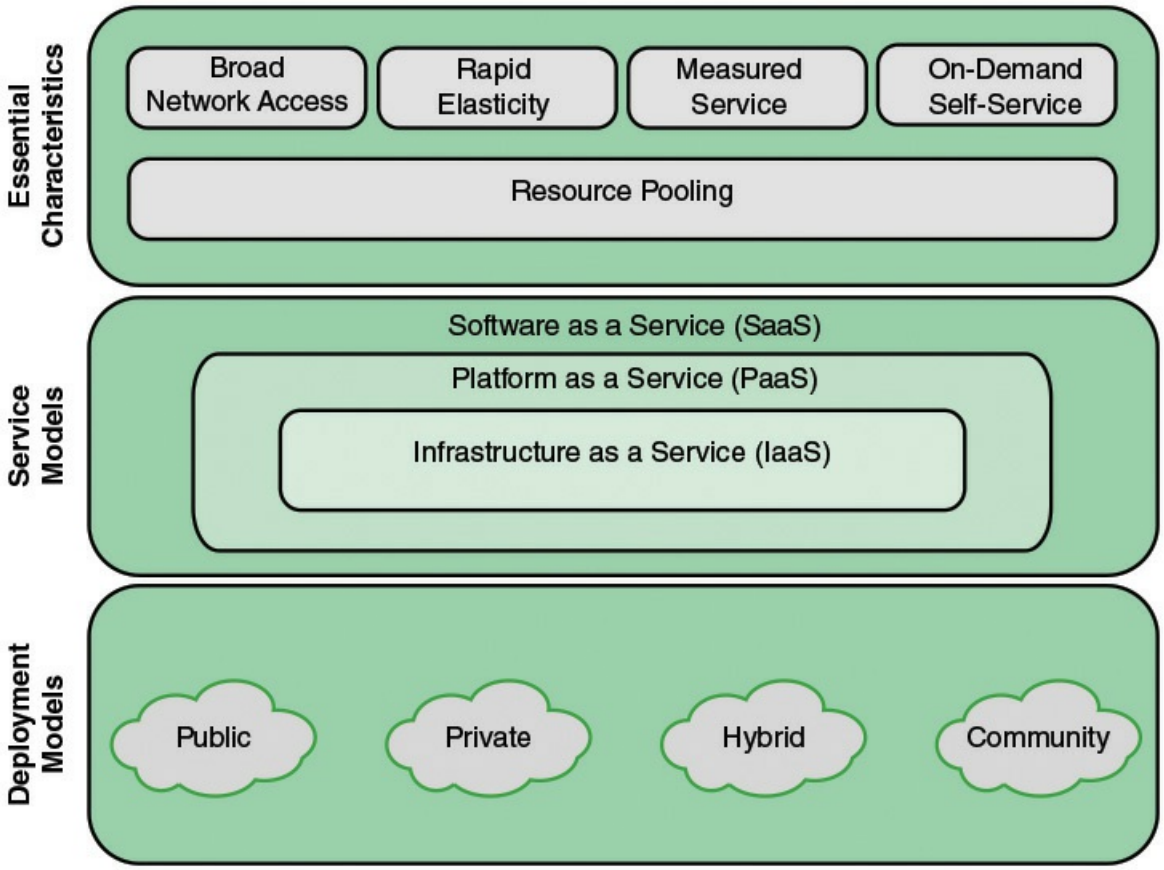


FIGURE 13.1 Cloud Computing Elements

← *See Section 2.2, "Demand: Big Data, Cloud Computing, and Mobile Traffic"*

Basically, cloud computing provides economies of scale, professional network management, and professional security management. These features can be attractive to companies large and small, government agencies, and individual PC and mobile users. The individual or company only needs to pay for the storage capacity and services they use. The user, be it company or individual, does not have the hassle of setting up a database system, acquiring the hardware they need, doing maintenance, and backing up the data—all these are part of the cloud service.

← *See Figure 2.4, Cloud Network Model*

In theory, another big advantage of using cloud computing to store your data and share it with others is that the cloud provider takes care of security. Alas, the customer is not always protected. There have been a number of security failures among cloud providers. Evernote made headlines in early 2013 when it told all of its users to reset their passwords after an intrusion was

discovered.

Cloud networking refers to the networks and network management functionality that must be in place to enable cloud computing. Most cloud computing solutions rely on the Internet, but that is only a piece of the networking infrastructure. One example of cloud networking is the provisioning of high-performance/high-reliability networking between the provider and subscriber. In this case, some or all of the traffic between an enterprise and the cloud bypasses the Internet and uses dedicated private network facilities owned or leased by the cloud service provider. More generally, cloud networking refers to the collection of network capabilities required to access a cloud, including making use of specialized services over the Internet, linking enterprise data centers to a cloud, and using firewalls and other network security devices at critical points to enforce access security policies.

We can think of **cloud storage** as a subset of cloud computing. In essence, cloud storage consists of database storage and database applications hosted remotely on cloud servers. Cloud storage enables small businesses and individual users to take advantage of data storage that scales with their needs and to take advantage of a variety of database applications without having to buy, maintain, and manage the storage assets.

13.2 Cloud Services

This section looks at commonly defined cloud services, beginning with three service models defined by NIST:

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

These can be viewed as nested service alternatives (see Figure 13.2) and are universally accepted as the basic service models for cloud computing. The section then examines other popular cloud service models.

FIGURE 13.2 Cloud Service Models

Software as a Service

As the name implies, an **SaaS** cloud provides service to customers in the form of software, specifically application software running on and accessible in the cloud. SaaS follows the familiar model of web services, in this case applied to cloud resources. SaaS enables the customer to use the cloud provider's applications running on the provider's cloud infrastructure. The applications are accessible from various client devices through a simple interface such as a web browser. Instead of obtaining desktop and server licenses for software products it uses, an enterprise obtains the same functions from the cloud service. The use of SaaS avoids the complexity of software installation, maintenance, upgrades, and patches. Examples of services at this level are Google Gmail, Microsoft 365, Salesforce, Citrix GoToMeeting, and Cisco WebEx.

Common subscribers to SaaS are organizations that want to provide their employees with access to typical office productivity software, such as document management and e-mail. Individuals also commonly use the SaaS model to acquire cloud resources. Typically, subscribers use specific applications on demand. The cloud provider also usually offers data-related features such as automatic backup and data sharing between subscribers.

The following list, derived from an ongoing industry survey by OpenCrowd (http://cloudtaxonomy.opencrowd.com/taxonomy), describes example SaaS services. The numbers in parentheses refer to the number of vendors currently offering each service.

Cloud Taxonomy

- **Billing** (3)**:** Application services to manage customer billing based on usage and subscriptions to products and services.
- **Collaboration** (18)**:** Platforms providing tools that allow users to collaborate in workgroups, within enterprises, and across enterprises.
- **Content management** (7)**:** Services for managing the production and access to content for Web-based applications.
- **Customer relationship management** (13)**:** Platforms for CRM application that range from call center applications to sales force automation.
- **Document management** (6)**:** Platforms of managing documents, document production workflows, and providing workspaces for groups or enterprises to find and access documents.
- **Education** (4)**:** Provides online services to Educators and Educational institutions.
- **Enterprise resource planning** (8)**:** ERP is an integrated computer-based system used to manage internal and external resources, including tangible assets, financial resources, materials, and human resources.
- **Financials** (11)**:** Applications for managing financial processes for companies that range from expense processing and invoicing to tax management.
- **Healthcare** (10)**:** Services for improving and managing people's health and healthcare management.
- **Human resources** (10)**:** Software for managing human resources functions within companies.
- **IT services management** (5)**:** Software that helps enterprises manage IT services delivery to services consumers and manage performance improvement.
- **Personal productivity** (5)**:** Software that business users use on a daily basis in the normal course of business. The typical suite includes applications for word processing, spreadsheets, and presentations.

- **Project management** (12)**:** Software packages for managing projects. Features of packages may specialize the offering for specific types of projects such as software development, construction, and so on.
- **Sales** (7)**:** Applications that are specifically designed for sales functions such as pricing, commission tracking, and so on.
- **Security** (10)**:** Hosted products for security services such as malware and virus scanning, single sign-on, and so on.
- **Social networks** (4)**:** Platforms for creating and customizing social networking applications.

Platform as a Service

A **PaaS** cloud provides service to customers in the form of a platform on which the customer's applications can run. PaaS enables the customer to deploy onto the cloud infrastructure customer-created or -acquired applications. A PaaS cloud provides useful software building blocks, plus a number of development tools, such as programming language tools, runtime environments, and other tools that assist in deploying new applications. In effect, PaaS is an operating system in the cloud. PaaS is useful for an organization that wants to develop new or tailored applications while paying for the needed computing resources only as needed and only for as long as needed. AppEngine, Engine Yard, Heroku, Microsoft Azure, Force.com, and Apache Stratos are examples of PaaS.

The following list describes example PaaS services. The numbers in parentheses refer to the number of vendors currently offering each service:

- **Big data as a service** (19)**:** These are cloud-based services for the analysis of large or complex data sets that require high scalability.
- **Business intelligence** (18)**:** Platforms for the creation of business intelligence applications such as dashboards, reporting systems, and big data analysis.
- **Database** (18)**:** These services offer scalable database systems ranging from relational database solutions to massively scalable non-SQL datastores.
- **Development and testing** (18) **:** These platforms are only for the development and testing cycles of application development, which expand and contract as needed.
- **General purpose** (22)**:** Platforms suited for general-purpose application development. These services provide a database, a web application runtime environment, and typically support web services for integration.
- **Integration** (14)**:** Services for integrating applications ranging from cloud-to-cloud integration to custom application integration.

Infrastructure as a Service

With **IaaS**, the customer has access to the resources of the underlying cloud infrastructure. IaaS

provides virtual machines and other abstracted hardware and operating systems. IaaS offers the customer processing, storage, networks, and other fundamental computing resources so that the customer can deploy and run arbitrary software, which can include operating systems and applications. IaaS enables customers to combine basic computing services, such as number crunching and data storage, to build highly adaptable computer systems.

Typically, customers are able to self-provision this infrastructure, using a web-based graphical user interface that serves as an IT operations management console for the overall environment. API access to the infrastructure may also be offered as an option. Examples of IaaS are Amazon Elastic Compute Cloud (Amazon EC2), Microsoft Windows Azure, Google Compute Engine (GCE), and Rackspace.

The following list describes example IaaS services. The numbers in parentheses refer to the number of vendors currently offering each service:

- **Backup and recovery** (14)**:** Platforms providing services to backup and recover file systems and raw data stores on servers and desktop systems.
- **Cloud broker** (7)**:** Tools that manage services on more than one cloud infrastructure platform. Some tools support private-public cloud configurations.
- **Compute** (31)**:** Provides server resources for running cloud-based systems that can be dynamically provisioned and configured as needed.
- **Content delivery networks** (2)**:** CDNs store content and files to improve the performance and cost of delivering content for web based systems.
- **Services management** (7)**:** Services that manage cloud infrastructure platforms. These tools often provide features that cloud providers do not provide or specialize in managing certain application technologies.
- **Storage** (12)**:** Provides massively scalable storage capacity that can be used for applications, backups, archiving, file storage, and more.

Figure 13.3 compares the functions implemented by the cloud service provider for the three principal cloud service models.

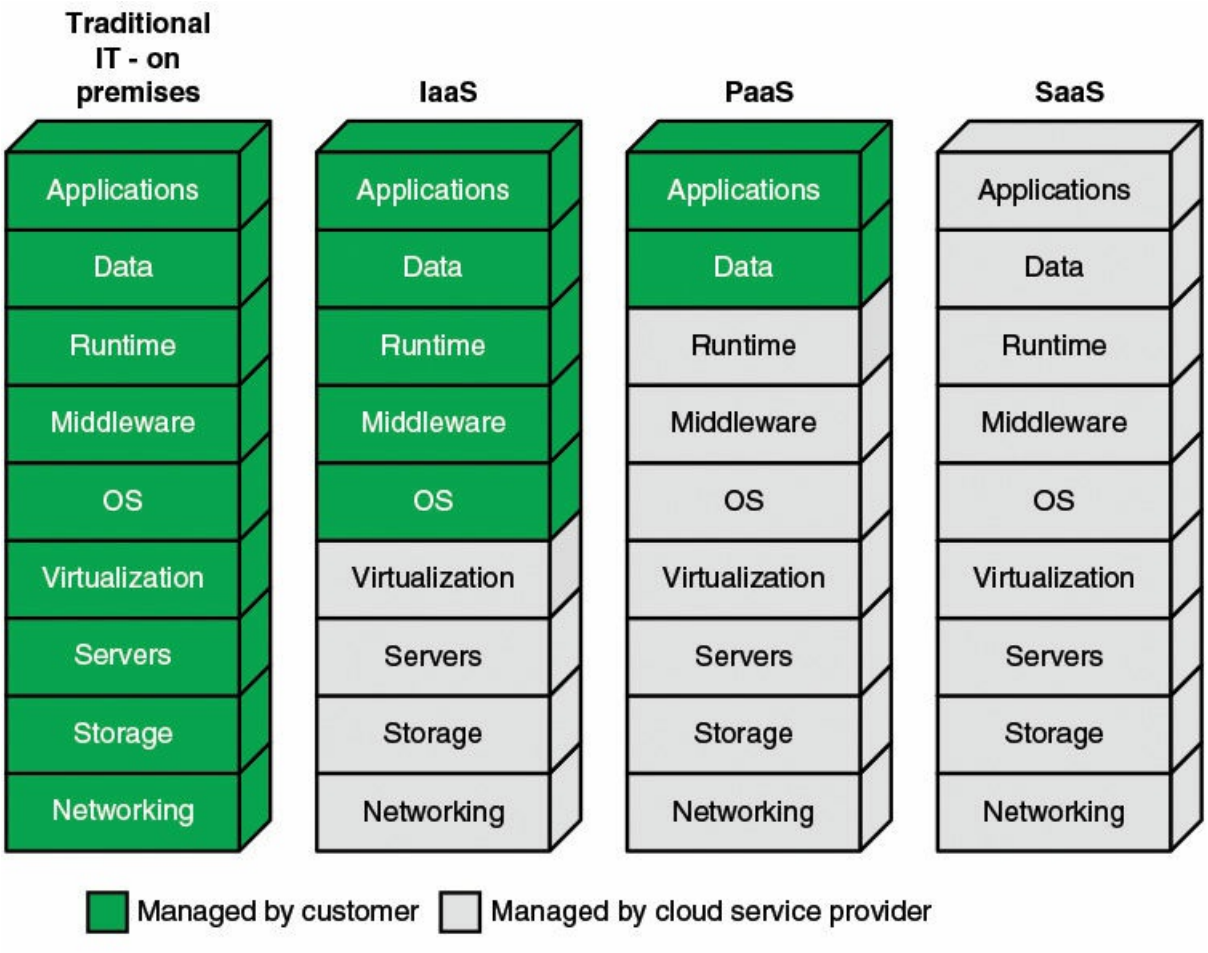


FIGURE 13.3 Separation of Responsibilities in Cloud Operation

Other Cloud Services

A number of other cloud services have been proposed, with some available as vendor offerings. A useful list of these additional services is provided by ITU-T Y.3500 (*Cloud Computing — Overview and Vocabulary*, August 2014).

In addition to SaaS, PaaS, and IaaS, Y.3500 lists the following as representative cloud service categories:

- **Communications as a Service (CaaS):** The integration of real-time interaction and collaboration services to optimize business processes. This service provides a unified interface and consistent user experience across multiple devices. Examples of services included are video teleconferencing, web conferencing, instant messaging, and voice over IP.
- **Compute as a Service (CompaaS):** The provision and use of processing resources needed to deploy and run software. CompaaS may be thought of as a simplified IaaS, with the focus on providing compute capacity.
- **Data Storage as a Service (DSaaS):** The provision and use of data storage and related

capabilities. DSaaS describes a storage model where the client leases storage space from a third-party provider. Data is transferred from the client to the service provider via the Internet, and the client then accesses the stored data using software provided by the storage provider. The software is used to perform common tasks related to storage, such as data backups and data transfers.

- **Network as a Service (NaaS):** Transport connectivity services / intercloud network connectivity services. NaaS involves the optimization of resource allocations by considering network and computing resources as a unified whole. NaaS can include flexible and extended virtual private network (VPN), bandwidth on demand, custom routing, multicast protocols, security firewall, intrusion detection and prevention, wide-area network (WAN), content monitoring and filtering, and antivirus.

Y.3500 distinguishes between cloud capabilities and cloud services. The three capabilities types are application, platform, and infrastructure, corresponding to the basic service types of SaaS, PaaS, and IaaS. A cloud service category can include capabilities from one or more cloud capability types. Table 13.1 shows the relationship of the seven cloud service categories and the three cloud capabilities types.

Cloud Service Categories	Cloud Capabilities Types		
	Infrastructure	Platform	Application
Compute as a Service	X		
Communications as a Service		X	X
Data Storage as a Service	X	X	X
Network as a Service	X	X	
Infrastructure as a Service	X		
Platform as a Service		X	
Software as a Service			X

TABLE 13.1 Cloud Service Categories and Cloud Capabilities Types

Y.3500 also lists examples of emerging cloud service categories:

- **Database as a Service:** Database functionalities on demand where the installation and maintenance of the databases are performed by the cloud service provider.
- **Desktop as a Service:** The ability to build, configure, manage, store, execute, and deliver user desktop functions remotely. In essence, Desktop as a Service offloads common desktop apps plus data from the user's desktop or laptop computer into the cloud. Designed to provide a reliable, consistent experience for the remote use of programs, applications, processes, and files.
- **E-mail as a Service:** A complete e-mail service, including related support services such as storage, receipt, transmission, backup, and recovery of e-mail.
- **Identity as a Service:** Identity and access management (IAM) that can be extended and centralized into existing operating environments. This includes provisioning, directory management, and the operation of a single sign-on service.
- **Management as a Service:** Includes application management, asset and change

management, capacity management, problem management (service desk), project portfolio management, service catalog, and service level management.

- **Security as a Service:** The integration of a suite of security services with the existing operating environment by the cloud service provider. This may include authentication, antivirus, antimalware/spyware, intrusion detection, and security event management, among others.

XaaS

XaaS is the latest development in the provisioning of cloud services. The acronym has three generally accepted interpretations, all of which mean pretty much the same thing:

- **Anything as a Service:** Where *anything* refers to any service other than the three traditional services.
- **Everything as a Service:** Although this version is sometimes spelled out, it is somewhat misleading, because no vendor offers every possible cloud service. This version is meant to suggest that the cloud service provider is providing a wide range of service offerings.
- **X as a Service:** Where *X* can represent any possible cloud service option.

XaaS providers go beyond the traditional "big three" services in three ways.

- Some providers package together SaaS, PaaS, and IaaS so that the customer can do one-stop shopping for the basic cloud services that enterprises are coming to rely on.
- XaaS providers can increasingly displace a wider range of services that IT departments typically offer internal customers. This strategy reduces the burden on the IT department to acquire, maintain, patch, and upgrade a variety of common applications and services.
- The XaaS model typically involves an ongoing relationship between customer and provider, in which there are regular status updates and a genuine two-way, real-time exchange of information. In effect, this is a managed service offering, enabling the customer to commit to only the amount of service needed at any time, and to expand both the amount and types of service as the customers needs evolve and as the offerings available expand.

XaaS is becoming increasingly attractive to customers because it offers these benefits:

- Total costs are controlled and lowered. By outsourcing the maximum range of IT services to a qualified expert partner, an enterprise sees both immediate and long-term cost reductions. Capital expenditures are drastically reduced because of the need to acquire far less hardware and software locally. Operating expenses are lower because the resources used are tailored to immediate needs and change only as needs change.
- Risks are lowered. XaaS providers offer agreed service levels. This eliminates the risks of cost overruns so common with internal projects. The use of a single provider for a wide range of services provides a single point of contact for resolving problems.
- Innovation is accelerated. IT departments constantly run the risk of installing new hardware and software only to find that later versions that are more capable, less expensive, or both are available by the time installation is complete. With XaaS, the latest

offerings are more quickly available. Further, providers can react quickly to customer feedback.

13.3 Cloud Deployment Models

An increasingly prominent trend in many organizations is to move a substantial portion or even all information technology (IT) operations to enterprise cloud computing. The organization is faced with a range of choices as to cloud ownership and management. This section looks at the four most prominent deployment models for cloud computing.

Public Cloud

A public cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services. The cloud provider is responsible both for the cloud infrastructure and for the control of data and operations within the cloud. A public cloud may be owned, managed, and operated by a business, academic, or government organization, or some combination of them. It exists on the premises of the cloud service provider.

In a public cloud model, all major components are outside the enterprise firewall, located in a multitenant infrastructure. Applications and storage are made available over the Internet via secured IP, and can be free or offered at a pay-per-usage fee. This type of cloud supplies easy-to-use consumer-type services, such as: Amazon and Google on-demand web applications or capacity, Yahoo! Mail, and Facebook or LinkedIn social media providing free storage for photographs. Although public clouds are inexpensive and scale to meet needs, they typically provide no or lower service level agreements (SLAs) and may not offer the guarantees against data loss or corruption found with private or hybrid cloud offerings. The public cloud is appropriate for consumers and entities not requiring the same levels of service that are expected within the firewall. Also, the public IaaS clouds do not necessarily provide for restrictions and compliance with privacy laws, which remain the responsibility of the subscriber or corporate end user. In many public clouds, the focus is on the consumer and small and medium-size businesses where pay-per-use pricing is available, often equating to pennies per gigabyte. Examples of services here might be picture and music sharing, laptop backup, or file sharing.

The major advantage of the public cloud is cost. A subscribing organization pays only for the services and resources it needs and can adjust these as needed. Further, the subscriber has greatly reduced management overhead. The principal concern is security; however, a number of public cloud providers have demonstrated strong security controls and, in fact, such providers may have more resources and expertise to devote to security that would be available in a private cloud.

Private Cloud

A private cloud is implemented within the internal IT environment of the organization. The organization may choose to manage the cloud in house or contract the management function to a third party. In addition, the cloud servers and storage devices may exist on premises or off premises.

Private clouds can deliver IaaS internally to employees or business units through an intranet or the Internet via a virtual private network (VPN), as well as software (applications) or storage as services to its branch offices. In both cases, private clouds are a way to leverage existing infrastructure, and deliver and chargeback for bundled or complete services from the privacy of the organization's network. Examples of services delivered through the private cloud include database on demand, e-mail on demand, and storage on demand.

A key motivation for opting for a private cloud is security. A private cloud infrastructure offers tighter controls over the geographic location of data storage and other aspects of security. Other benefits include easy resource sharing and rapid deployment to organizational entities.

Community Cloud

A community cloud shares characteristics of private and public clouds. Like a private cloud, a community cloud has restricted access. Like a public cloud, the cloud resources are shared among a number of independent organizations. The organizations that share the community cloud have similar requirements and, typically, a need to exchange data with each other. One example of an industry that is using the community cloud concept is the healthcare industry. A community cloud can be implemented to comply with government privacy and other regulations. The community participants can exchange data in a controlled fashion.

The cloud infrastructure may be managed by the participating organizations or a third party and may exist on premises or off premises. In this deployment model, the costs are spread over fewer users than a public cloud (but more than a private cloud), so only some of the cost savings potential of cloud computing are realized.

Hybrid Cloud

The hybrid cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (for example, cloud bursting for load balancing between clouds). With a hybrid cloud solution, sensitive information can be placed in a private area of the cloud, and less sensitive data can take advantage of the benefits of the public cloud.

A hybrid public/private cloud solution can be particularly attractive for smaller businesses. Many applications for which security concerns are less can be offloaded at considerable cost savings without committing the organization to moving more sensitive data and applications to the public cloud.

Table 13.2 lists some of the relative strengths and weaknesses of the four cloud deployment models.

	Private	Community	Public	Hybrid
Scalability	Limited	Limited	Very high	Very high
Security	Most secure option	Very secure	Moderately secure	Very secure
Performance	Very good	Very good	Low to medium	Good
Reliability	Very high	Very high	Medium	Medium to high
Cost	High	Medium	Low	Medium

TABLE 13.2 Comparison of Cloud Deployment Models

13.4 Cloud Architecture

To gain a better understanding of the elements of a cloud system, this section examines two reference architectures.

NIST Cloud Computing Reference Architecture

NIST SP 500-292, *NIST Cloud Computing Reference Architecture*, September 2011, establishes a reference architecture, described as follows:

> The NIST cloud computing reference architecture focuses on the requirements of "what" cloud services provide, not a "how to" design solution and implementation. The reference architecture is intended to facilitate the understanding of the operational intricacies in cloud computing. It does not represent the system architecture of a specific cloud computing system; instead it is a tool for describing, discussing, and developing a system-specific architecture using a common framework of reference.

NIST developed the reference architecture with the following objectives in mind:

- To illustrate and understand the various cloud services in the context of an overall cloud computing conceptual model.
- To provide a technical reference for consumers to understand, discuss, categorize, and compare cloud services.
- To facilitate the analysis of candidate standards for security, interoperability, and portability and reference implementations.

Cloud Computing Actors

The reference architecture depicted in Figure 13.4 defines five major actors in terms of the roles and responsibilities, as defined in the list that follows.

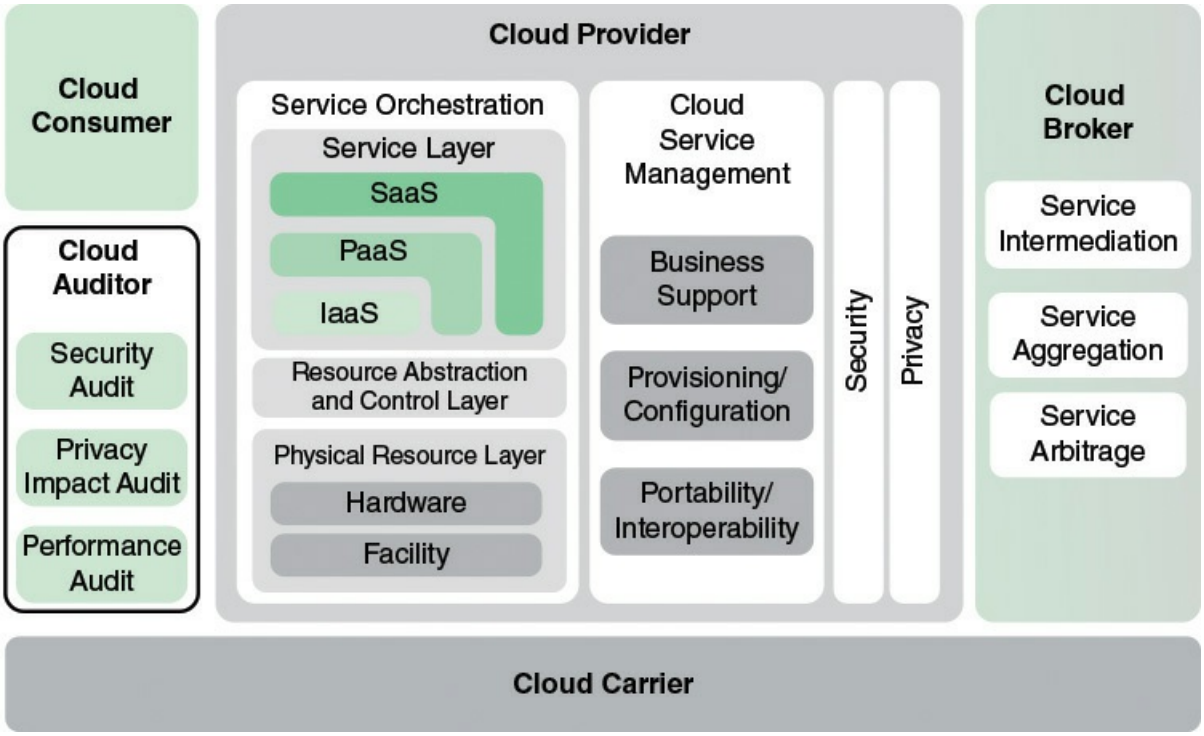


FIGURE 13.4 NIST Cloud Computing Reference Architecture

- **Cloud consumer**: A person or organization that maintains a business relationship with and uses services from cloud providers.
- **Cloud provider (CP)**: A person, organization, or entity responsible for making a service available to interested parties.
- **Cloud auditor**: A party that can conduct independent assessment of cloud services, information system operations, performance, and security of the cloud implementation.
- **Cloud broker**: An entity that manages the use, performance and delivery of cloud services and negotiates relationships between CPs and cloud consumers.
- **Cloud carrier**: An intermediary that provides connectivity and transport of cloud services from CPs to cloud consumers.

The roles of the cloud consumer and provider have already been discussed. To summarize, a **cloud provider** can provide one or more of the cloud services to meet IT and business requirements of **cloud consumers**. For each of the three service models (SaaS, PaaS, IaaS), the CP provides the storage and processing facilities needed to support that service model, together with a cloud interface for cloud service consumers. For SaaS, the CP deploys, configures, maintains, and updates the operation of the software applications on a cloud infrastructure so that the services are provisioned at the expected service levels to cloud consumers. The consumers of SaaS can be organizations that provide their members with access to software applications, end users who directly use software applications, or software application administrators who configure applications for end users.

For PaaS, the CP manages the computing infrastructure for the platform and runs the cloud software that provides the components of the platform, such as runtime software execution stack,

databases, and other middleware components. Cloud consumers of PaaS can employ the tools and execution resources provided by CPs to develop, test, deploy, and manage the applications hosted in a cloud environment.

For IaaS, the CP acquires the physical computing resources underlying the service, including the servers, networks, storage, and hosting infrastructure. The IaaS cloud consumer in turn uses these computing resources, such as a virtual computer, for their fundamental computing needs.

The **cloud carrier** is a networking facility that provides connectivity and transport of cloud services between cloud consumers and CPs. Typically, a CP will set up SLAs with a cloud carrier to provide services consistent with the level of SLAs offered to cloud consumers, and may require the cloud carrier to provide dedicated and secure connections between cloud consumers and CPs.

A **cloud broker** is useful when cloud services are too complex for a cloud consumer to easily manage. Three areas of support can be offered by a cloud broker:

- **Service intermediation:** These are value-added services, such as identity management, performance reporting, and enhanced security.
- **Service aggregation:** The broker combines multiple cloud services to meet consumer needs not specifically addressed by a single CP, or to optimize performance or minimize cost.
- **Service arbitrage**: This is similar to service aggregation except that the services being aggregated are not fixed. Service arbitrage means a broker has the flexibility to choose services from multiple agencies. The cloud broker, for example, can use a credit-scoring service to measure and select an agency with the best score.

A **cloud auditor** can evaluate the services provided by a CP in terms of security controls, privacy impact, performance, and so on. The auditor is an independent entity that can assure that the CP conforms to a set of standards.

Figure 13.5 illustrates the interactions between the actors. A cloud consumer may request cloud services from a cloud provider directly or via a cloud broker. A cloud auditor conducts independent audits and may contact the others to collect necessary information. This figure shows that cloud networking issues in fact involve three separate types of networks. For a cloud producer, the network architecture is that of a typical large data center, which consists of racks of high-performance servers and storage devices, interconnected with high-speed top-of-rack Ethernet switches. The concerns in this context focus on virtual machine placement and movement, load balancing, and availability issues. The enterprise network is likely to have a quite different architecture, typically including a number of LANs, servers, workstations, PCs, and mobile devices, with a broad range of network performance, security, and management issues. The concern of both producer and consumer with respect to the cloud carrier, which is shared with many users, is the ability to create virtual networks, with appropriate SLA and security guarantees.

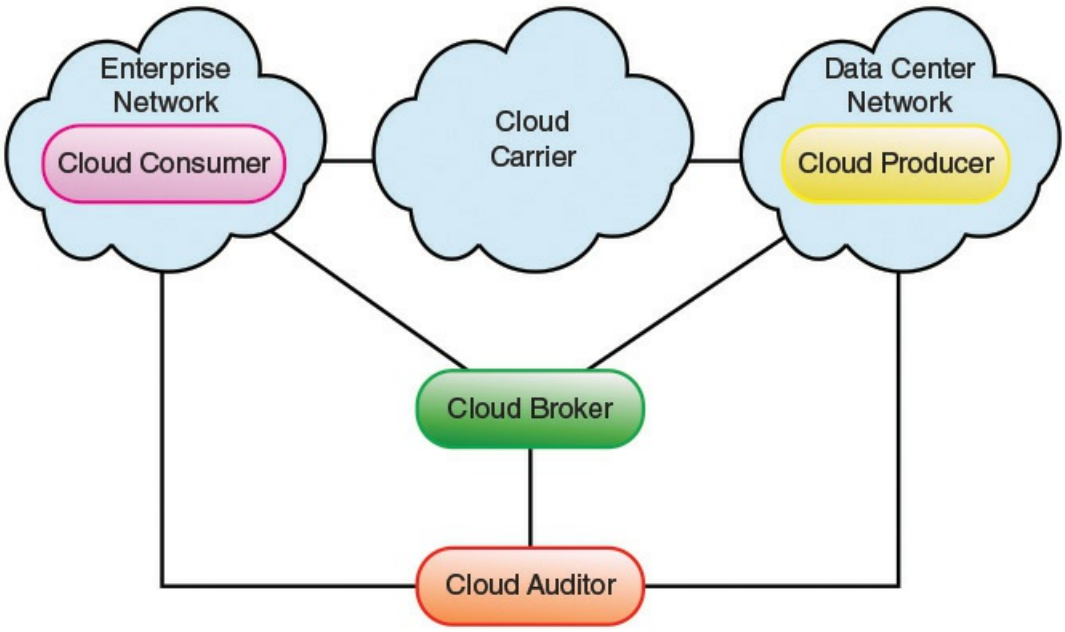


FIGURE 13.5 Interactions Between Actors in Cloud Computing

Cloud Provider Architectural Components

Figure 13.4 shows four main architectural components of the cloud provider. **Service orchestration** refers to the composition of system components to support the cloud provider activities in arrangement, coordination, and management of computing resources to provide cloud services to cloud consumers. Orchestration is shown as a three-layer architecture. We see here the familiar mapping of physical resources to consumer-visible services by a resource abstraction layer. Examples of resource abstraction components include software elements such as hypervisors, virtual machines, virtual data storage, and other computing resource abstractions.

Cloud service management includes all the service-related functions necessary for the management and operation of those services required by or proposed to cloud consumers. It covers three main areas:

- **Business support:** This consists of business-related services dealing with customers, such as accounting, billing, reporting, and auditing.
- **Provisioning/configuration:** This includes automated tools for rapid deployment of cloud systems for consumers, adjusting configuration and resource assignment, and monitoring and reporting on resource usage.
- **Portability/interoperability:** Consumers are interested in cloud offering that support data and system portability and service interoperability. This is particularly useful in a hybrid cloud environment, in which the consumer may want to change the allocation of data and applications between on-premises and off-premises sites.

Security and **privacy** are concerns that encompass all layers and elements of the cloud provider's architecture.

ITU-T Cloud Computing Reference Architecture

It is useful to look at an alternative reference architecture, published in ITU-T Y.3502, *Cloud Computing Architecture*, August 2014. This architecture is somewhat broader in scope than the NIST architecture and views the architecture as a layered functional architecture.

Cloud Computing Actors

Before looking at the four-layer reference architecture, we need to note the differences between NIST and ITU-T in defining cloud actors. The ITU-T document defines three actors:

- **Cloud service customer or user:** A party that is in a business relationship for the purpose of using cloud services. The business relationship is with a cloud service provider or a cloud service partner. Key activities for a cloud service customer include, but are not limited to, using cloud services, performing business administration, and administering use of cloud services.
- **Cloud service provider:** A party that makes cloud services available. The cloud service provider focuses on activities necessary to provide a cloud service and activities necessary to ensure its delivery to the cloud service customer as well as cloud service maintenance. The cloud service provider includes an extensive set of activities (for example, provide service, deploy and monitor service, manage business plan, provide audit data) as well as numerous subroles (for example, business manager, service manager, network provider, security and risk manager).
- **Cloud service partner:** A party which is engaged in support of, or auxiliary to, activities of either the cloud service provider or the cloud service customer, or both. A cloud service partner's activities vary depending on the type of partner and their relationship with the cloud service provider and the cloud service customer. Examples of cloud service partners include cloud auditor and cloud service broker.

Figure 13.6 depicts the actors with some of their possible roles in a cloud ecosystem.

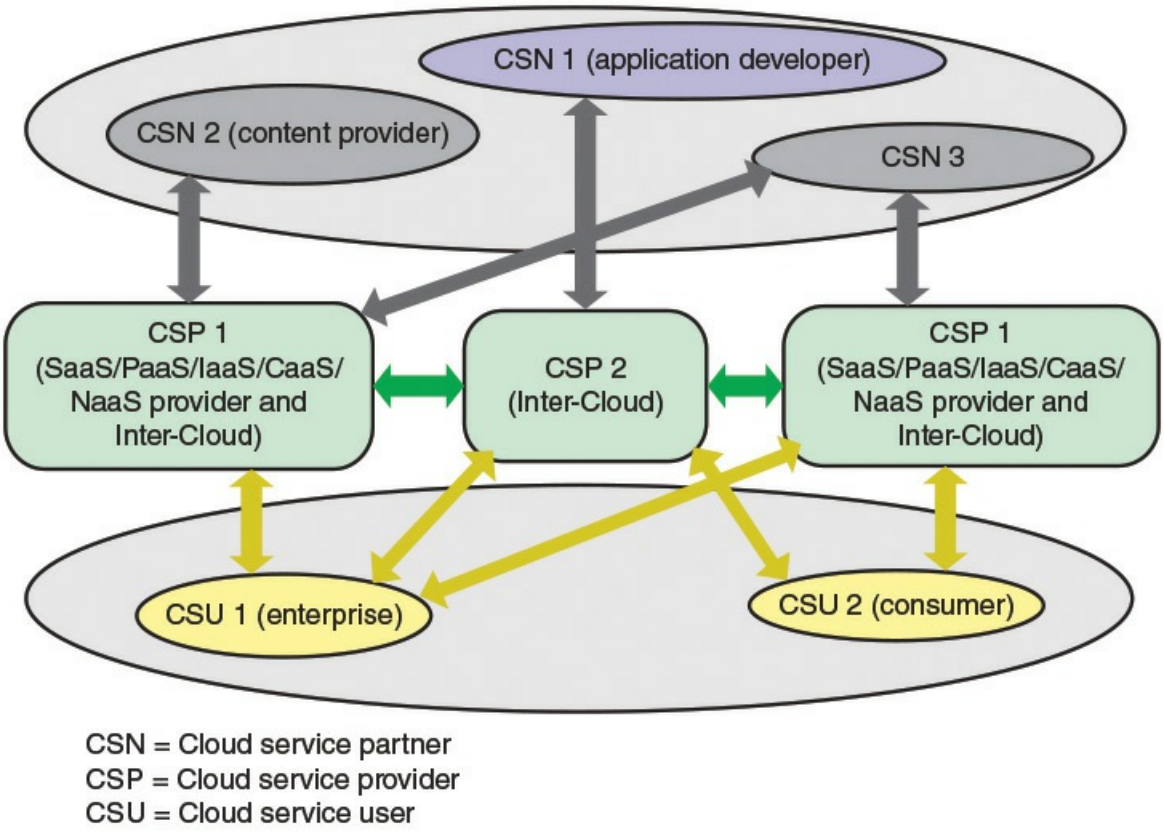


FIGURE 13.6 Actors with Some of Their Possible Roles in a Cloud Ecosystem

Layered Architecture

Figure 13.7 shows the four-layer ITU-T cloud computing reference architecture. The user layer is the user interface through which a cloud service customer interacts with a cloud service provider and with cloud services, performs customer related administrative activities, and monitors cloud services. It can also offer the output of cloud services to another resource layer instance. When the cloud receives service requests, it orchestrates its own resources and/or other clouds' resources (if other clouds' resources are received via the intercloud function) and provides back cloud services through the user layer. The user layer is where the CSU resides.

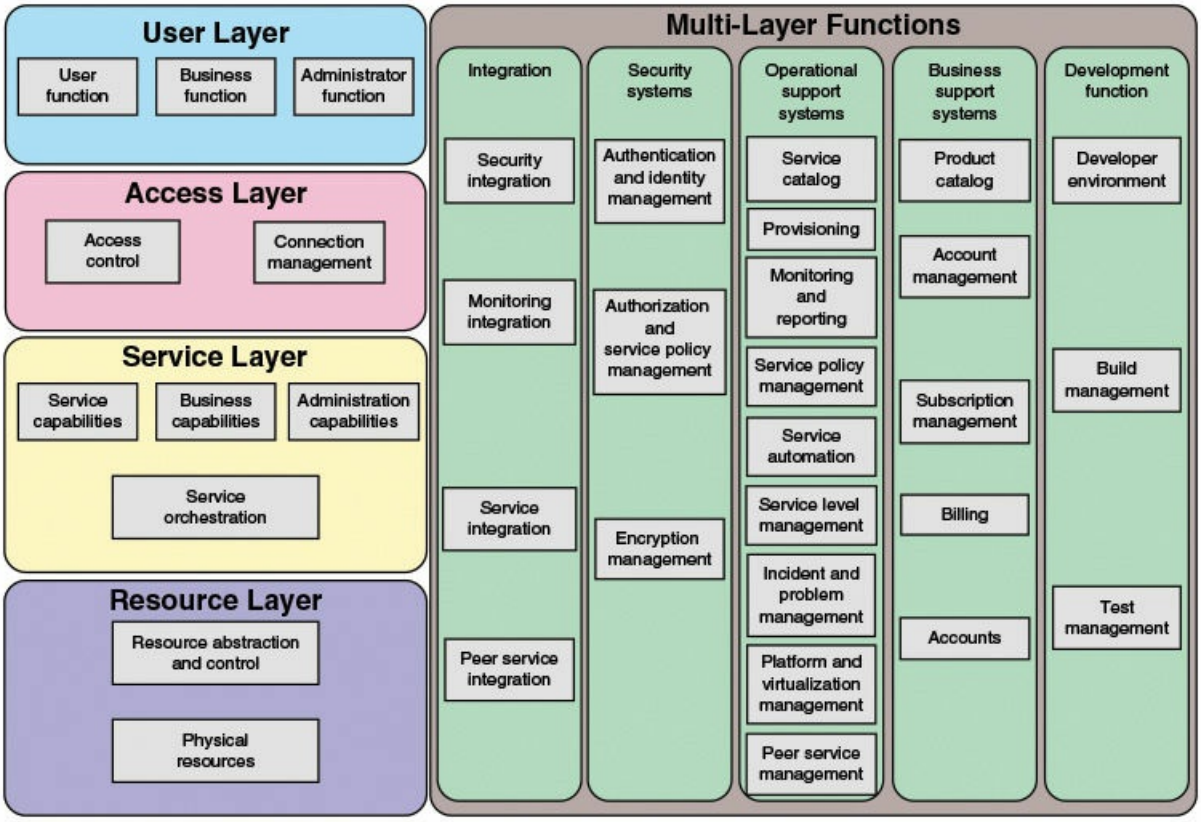


FIGURE 13.7 ITU-T Cloud Computing Reference Architecture

The access layer provides a common interface for both manual and automated access to the capabilities available in the services layer. These capabilities include both the capabilities of the services and also the administration and business capabilities. The access layer accepts user/partner/other provider cloud service consumption requests using cloud application programming interfaces (APIs) to access the provider's services and resources.

The access layer is responsible for presenting cloud service capabilities over one or more access mechanisms—for example, as a set of web pages accessed via a browser, or as a set of web services that can be accessed programmatically. The access layer also deals with security and QoS.

The service layer contains the implementation of the services provided by a cloud service provider (for example, SaaS, PaaS, IaaS). The service layer contains and controls the software components that implement the services (but not the underlying hypervisors, host operating systems, device drivers, and so on), and arranges to offer the cloud services to users via the access layer.

The resource layer consists of physical resources available to the provider and the appropriate abstraction and control mechanisms. For example, hypervisor software can provide virtual network, virtual storage, and virtual machine capabilities. It also houses the cloud core transport network functionality that is required to provide underlying network connectivity between the provider and users.

The multilayer functions include a series of functional components that interact with functional

components of the four other layers to provide supporting capabilities. It includes five categories of functional components:

- **Integration:** Responsible for connecting functional components in the architecture to create a unified architecture. The integration functional components provide message routing and message exchange mechanisms within the cloud architecture and its functional components as well as with external functional components.
- **Security systems:** Responsible for applying security related controls to mitigate the security threats in cloud computing environments. The security systems functional components encompass all the security facilities required to support cloud services.
- **Operational support systems (OSS):** Encompass the set of operational related management capabilities that are required to manage and control the cloud services offered to customers. OSS is also involved in system monitoring, including the use of alarms and events.
- **Business support systems (BSS):** Encompass the set of business-related management capabilities dealing with customers and supporting processes, such as billing and accounts.
- **Development function:** Supports the cloud computing activities of the cloud service developer. This includes support of the development/composition of service implementations, build management and test management.

13.5 SDN and NFV

Cloud computing predates software-defined networking (SDN) and network functions virtualization (NFV). While cloud computing can be, and has been, deployed and managed without SDN and NFV, both of these technologies are compelling for both private cloud operators and public cloud service providers.

In simplified and generalized terms, what SDN offers is centralized command and control of network resources and traffic patterns. A single central controller, or a few distributed cooperating controllers, can configure and manage virtual networks and provide QoS and security services. This relieves network management of the need to individually configure and program each networking device.

What NFV offers is automated provisioning of devices. NFV virtualizes network devices, such as switches and firewalls, as well as compute and storage devices, and provides tools for scaling out and automatically deploying devices as needed. Therefore, each project or cloud customer does not require separate equipment or reprogramming of existing equipment. Relevant devices can be centrally deployed via a hypervisor management platform and configured with rules and policies.

Service Provider Perspective

A large cloud service provider will deal with thousands of customers, with dynamic needs for capacity, both in terms of traffic-carrying capacity and in terms of compute and storage resources. The provider needs to be able to rapidly manage the entire network to handle traffic

bottlenecks, manage numerous traffic flows with differing QoS requirements, and deal with outages and other problems. All of this must be done in a secure manner. SDN can provide the needed overall view of the entire network and secure, centralized management of the network. The provider needs to be able to deploy and scale in/out and up/down virtual switches, servers, and storage rapidly and transparently for the customer. NFV provides the automated tools for managing this process.

Private Cloud Perspective

Large and medium-size enterprises see a number of advantages to moving much of their network-based operations to a private cloud or a hybrid cloud. Their customers are end users, IT managers, and developers. Individual departments may have substantial, dynamic IT resource needs. The enterprise typically will need to develop one or multiple server farms / data centers. As the overall resource demand grows, the ability to deploy and manage all of the equipment becomes more challenging. In addition, there are security requirements, such as firewalls, and antivirus deployments. Further complicating the scenario is the need for load balancing as projects grow and consume more resources, thus the need for rapid scalability and provisioning of devices becomes more pronounced. The need for automated provisioning of virtual networking equipment almost becomes a requirement, and with all the new virtual devices (especially in conjunction with the existing physical devices), centralized command and control becomes a must. SDN and NFV provide the enterprise with the tools to successfully develop and manage private cloud resources for internal use.

ITU-T Cloud Computing Functional Reference Architecture

Figure 13.7 showed the four-layer cloud computing reference architecture defined in Y.3502. For our discussion of the relationship between cloud networking and NFV, it is instructive to look at an earlier version of this architecture, defined in *ITU-T Focus Group on Cloud Computing Technical Report, Part 2: Functional Requirements and Reference Architecture*, February 2012 and shown in Figure 13.8. This architecture has the same four-layer structure as that of Y.3502, but provides more detail of the lowest layer, called the resources and network layer. This layer consists of three sublayers as defined in the list that follows.

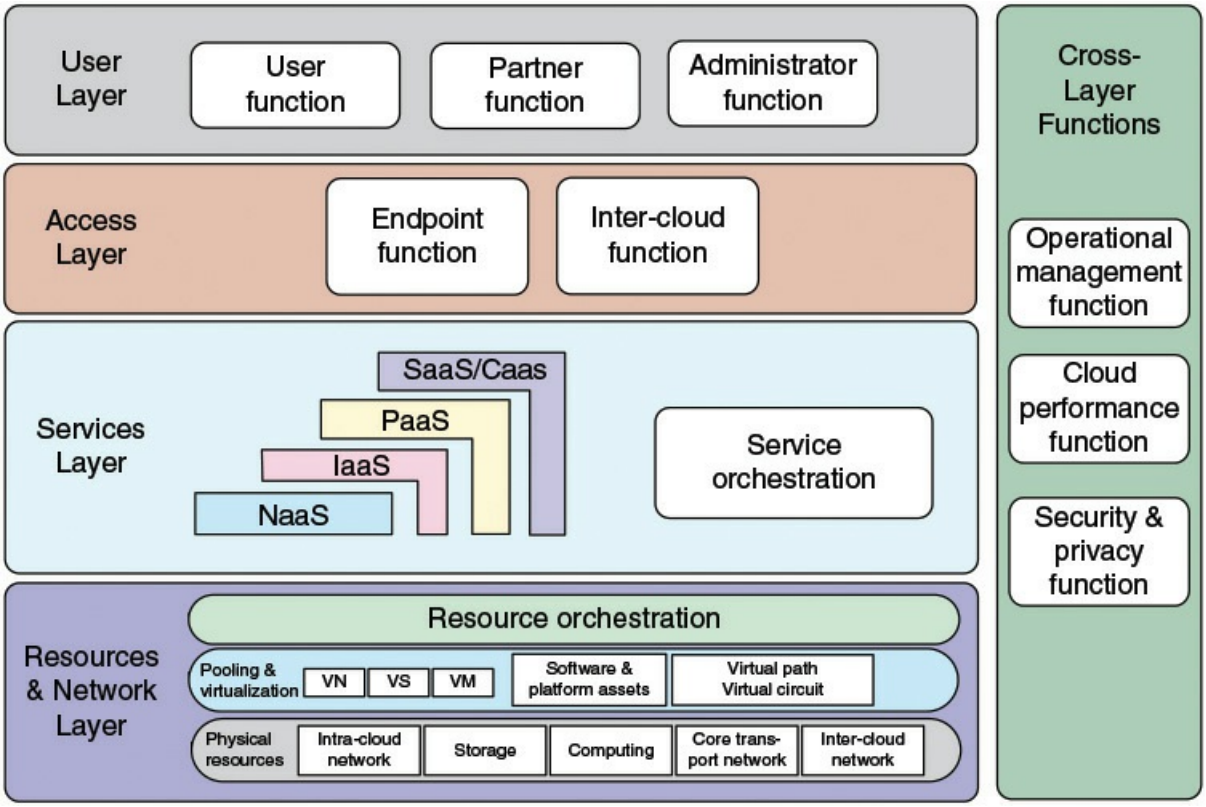


FIGURE 13.8 ITU-T Cloud Computing Functional Reference Architecture

- **Resource orchestration:** The management, monitoring, and scheduling of computing, storage, and network resources into consumable services by the upper layers and users. It controls the creation, modification, customization and release of virtualized resources.
- **Pooling and virtualization:** The virtualization function turns physical resources into virtual machines, virtual storage, and virtual networks. These virtual resources are in turn managed and controlled by the resource orchestration, based on user demand. Software and platform assets in the pooling and virtualization layer are the runtime environment, applications, and other software assets used to orchestrate and implement cloud services.
- **Physical resources:** The computing, storage, and network resources that are fundamental to providing cloud services. These resources may include those that reside inside cloud-data centers (for example, computing servers, storage servers, and intracloud networks), and those that reside outside of data centers, typically networking resources, such as intercloud networks and core transport networks.

A comparison of the resources and network layer of the ITU-T architecture to the NFV architectural framework (refer to Figure 7.7 in Chapter 7, "Network Functions Virtualization: Concepts and Architecture") suggests that the resources and network layer can be implemented using the network functions virtualization infrastructure (NFVI) for the lower two sublayers and virtualized infrastructure manager (VIM) for the resource orchestration sublayer. Thus, the general-purpose tools, often in the form of open software, plus commercial off-the-shelf physical resources, enable the cloud provider to effectively deploy and manage cloud services and resources. It should also be an effective strategy to map many of the upper layer functions in the cloud architecture to either virtual network functions or SDN control and application layer

functions. Thus, both NFV and SDN contribute to the deployment of cloud services.

Similar reasoning applies to the NIST reference architecture shown previously in Figure 13.4. The service orchestration component consists of three layers: physical resource, resource abstraction and control, and service layers. The lower two layers correspond quite well to the NFVI portion of the NFV architecture.

13.6 Key Terms

After completing this chapter, you should be able to define the following terms.

Anything as a Service (XaaS)

cloud auditor

cloud broker

cloud carrier

cloud computing

cloud consumer

cloud networking

cloud provider

cloud service customer

cloud service management

cloud service partner

cloud service provider

cloud storage

Communications as a Service (CaaS)

community cloud

Compute as a Service (CompaaS)

Data Storage as a Service (DSaaS)

hybrid cloud

Infrastructure as a Service (IaaS)

Network as a service (NaaS)

Platform as a Service (PaaS)

private cloud

public cloud

service orchestration

Software as a Service (SaaS)

Chapter 14. The Internet of Things: Components

> Within our grasp is the leisure of the Greek citizen, made possible by our mechanical slaves, which far outnumber his twelve to fifteen per free man. These mechanical slaves jump to our aid. As we step into a room, at the touch of a button a dozen light our way. Another slave sits twenty-four hours a day at our thermostat, regulating the heat of our home. Another sits night and day at our automatic refrigerator. They start our car; run our motors; shine our shoes, and cut our hair. They practically eliminate time and space by their very fleetness.
>
> —*Spectatoritis*, Jay B. Nash, 1932

Chapter Objectives: **After studying this chapter, you should be able to**

- Explain the scope of the Internet of Things.
- List and discuss the five principal components of IoT-enabled things.

Section 1.7 provided a brief overview of the concept of the Internet of Things (IoT). This chapter and the next provide a more detailed treatment. This chapter begins with a discussion of the basic concepts and scope of IoT. Then, Section 14.3 lists and discusses the main components of IoT-enabled things. Chapter 15, "The Internet of Things: Architecture and Implementation," discusses IoT architecture and implementation.

14.1 The IoT Era Begins

The future Internet will involve large numbers of objects that use standard communications architectures to provide services to end users. It is envisioned that tens of billions of such devices will be interconnected in a few years. This will provide new interactions between the physical world and computing, digital content, analysis, applications, and services. This resulting networking paradigm is being called the Internet of Things (IoT). This will provide unprecedented opportunities for users, manufacturers, and service providers in a wide variety of sectors. Areas that will benefit from IoT data collection, analysis, and automation capabilities include health and fitness, healthcare, home monitoring and automation, energy savings and smart grid, farming, transportation, environmental monitoring, inventory and product management, security, surveillance, education, and many others.

Technology development is occurring in many areas. Not surprisingly, wireless networking research is being conducted and actually has been conducted for quite a while now, but under previous titles such as mobile computing, pervasive computing, wireless sensor networks, and cyber-physical systems. Many proposals and products have been developed for low power protocols, security and privacy, addressing, low cost radios, energy efficient schemes for long battery life, and reliability for networks of unreliable and intermittently sleeping nodes. These wireless developments are crucial for the growth of IoT. In addition, areas of development have also involved giving IoT devices social networking capabilities, taking advantage of machine-to-

machine communications, storing and processing large amounts of real-time data, and application programming to provide end users with intelligent and useful interfaces to these devices and data.

Many have provided a vision for the IoT. In a 2014 paper in the *Internet of Things Journal* [STAN14], the author suggests personal benefits such as digitizing daily life activities, patches of bionic skin to communicate with surrounding smart spaces for improved comfort, health, and safety, and smart watches and body nodes that optimize access to city services. Citywide benefits could include efficient, delay-free transportation with no traffic lights. Smart buildings could not only control energy and security, but also support health and wellness activities. In the same ways people have been provided new ways of accessing the world through smartphones, the IoT will create a new paradigm in the ways we have continuous access to needed information and services. Regardless of the level of positivity in one's view of the IoT or predictions about how soon this will be realized, it is certainly exciting to consider this future.

14.2 The Scope of the Internet of Things

ITU-T Y.2060, *Overview of the Internet of Things*, June 2012, provides the following definitions that suggest the scope of IoT:

- **Internet of Things (IoT):** A global infrastructure for the information society, enabling advanced services by interconnecting (physical and virtual) things based on existing and evolving interoperable information and communication technologies.
- **Thing:** With regard to the IoT, this is an object of the physical world (physical things) or the information world (virtual things), which is capable of being identified and integrated into communication networks.
- **Device:** With regard to the IoT, this is a piece of equipment with the mandatory capabilities of communication and the optional capabilities of sensing, actuation, data capture, data storage, and data processing.

Most of the literature views the IoT as involving intercommunicating smart objects. Y.2060 extends this to include virtual things, a topic examined in Section 14.4.

Y.2060 characterizes the IoT as adding the dimension "Any THING communication" to the information and communication technologies which already provide "any TIME" and "any PLACE" communication (see Figure 14.1).

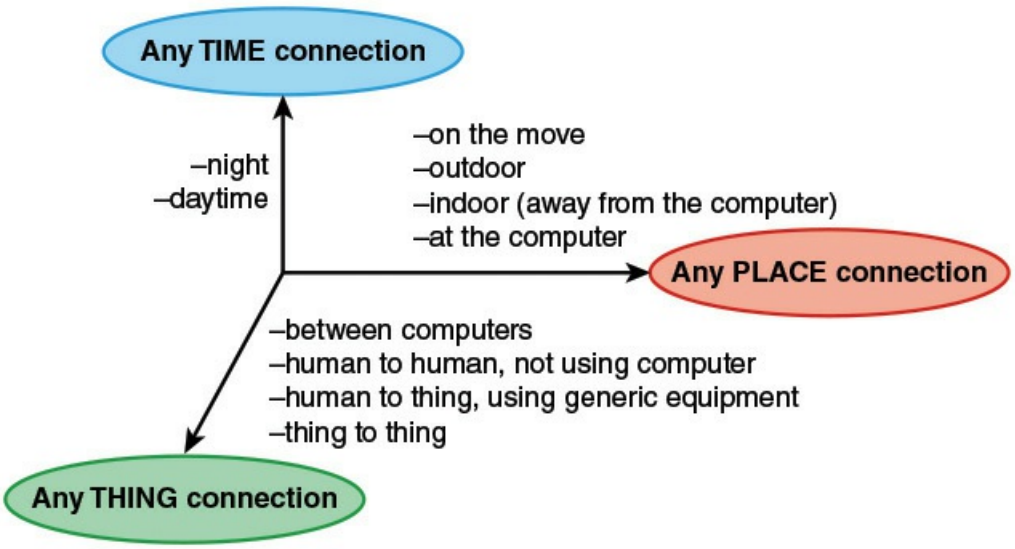


FIGURE 14.1 The New Dimension Introduced in the Internet of Things

In *Designing the Internet of Things* [MCEW13], the author condenses the elements of the IoT into a simple equation:

Physical objects + Controllers, Sensors, Actuators + Internet = IoT

This equation neatly captures the essence of the Internet of Things. An instance of the IoT consists of a collection of physical objects, each of which

- Contains a microcontroller that provides intelligence
- Contains a sensor that measures some physical parameter/actuator that acts on some physical parameter
- Provides a means of communicating via the Internet or some other network

One item not covered in the equation, and referred to in the Y.2060 definition, is a means of identification of an individual thing, usually referred to as a *tag*. We discuss tags in Section 14.3.

Note that although the phrase *the Internet of Things* is always used in the literature, a more accurate description would be *an Internet of Things*, or *a network of things*. A smart home installation, for example, consists of a number of things in the home that are interconnected via Wi-Fi or Bluetooth with some central controller. In a factory or farm setting, there may be a network of things enabling enterprise applications to interact with the environment and run applications to exploit the network of things. In these examples, it is usually but not invariably the case that remote access over the Internet is available. Whether such Internet connection is available, the collection of smart objects at a site, plus any other local compute and storage device, can be characterized as a network or an Internet of Things.

Table 14.1, based on a graphic from Beechem Research, gives an idea of the scope of IoT.

Service Sectors	Application Groups	Locations	Device Examples
IT and networks	Public	Services, e-commerce, data centers, mobile carriers, fixed carriers	Servers, storage, PCs, routers, switches, PBXs
		ISPs	
	Enterprise	IT/data center, office, private nets	
Security/public safety	Surveillance equipment, tracking	Radar/satellite, military security, unmanned, weapons, vehicles, ships, aircraft, gear	Tanks, fighter jets, battlefield communications, jeeps
	Public infrastructure	Human, animal, postal, food/health, packaging, baggage, water treatment, building environmental, general environmental	Cars, breakdown lane worker, homeland security, fire, environmental monitor
	Emergency services	Equipment and personnel, police, fire, regulatory	Ambulances, public security vehicles
Retail	Specialty	Fuel stations, gaming, bowling, cinema, discos, special events	POS terminals, tags, cash registers, vending machines, signs
	Hospitality	Hotels, restaurants, bars, cafes, clubs	
	Stores	Supermarkets, shopping centers, single site, distribution center	

Transportation	Nonvehicular	Air, rail, marine	Vehicles, lights, ships, planes, signage, tolls
	Vehicles	Consumer, commercial, construction, off-road	
	Transportation systems	Tolls, traffic management, navigation	
Industrial	Distribution	Pipelines, materials handling, conveyance	Pumps, valves, vats, conveyers, pipelines, motors, drives, converting, fabrication, assembly/packing, vessels, tanks
	Converting, discrete	Metals, paper, rubber, plastic, metalworking, electronics assembly, test	
	Fluid/processes	Petro-chemical, hydrocarbon, food, beverage	
	Resource automation	Mining, irrigation, agricultural, woodland	
Healthcare and life science	Care	Hospital, ER, mobile PoC, clinic, labs, doctor office	MRIs, PDAs, implants, surgical equipment, pumps, monitors, telemedicine
	In-vivo, home	Implants, home monitoring systems	
	Research	Drug discovery, diagnostics, labs	
Consumer and home	Infrastructure	Wiring, network access, energy mgt	Digital camera, power systems, dishwashers, eReaders, desktop computers, washer/dryer, meters, lights, TVs, MP3, games console, lighting, alarms
	Awareness and safety	Security/alert, fire safety, environmental safety, elderly, children, power protection	
	Convenience and entertainment	HVAC/climate, lighting, appliance, entertainment	
Energy	Supply/demand	Power generation, transport and distribution, low voltage, power quality, energy management	Turbines, windmills, uninterruptible power supply (UPS), batteries, generators, meters, drills, fuel cells
	Alternative	Solar, wind, co-generation, electro-chemical	
	Oil/gas	Rigs, derricks, well heads, pumps, pipelines	
Buildings	Commercial, institutional	Office, education, retail, hospitality, healthcare, airports, stadiums	HVAC, transport, fire and safety, lighting, security, access
	Industrial	Process, clean room, campus	

Source: Beecham Research

TABLE 14.1 The Internet of Things

14.3 Components of IoT-Enabled Things

The key ingredients of an IoT-enabled thing are sensors, actuators, a microcontroller, a means of communication (transceiver), and a means of identification (radio-frequency identification [RFID]). A means of communication is an essential ingredient; otherwise, the device cannot participate in a network. Nearly all IoT-enabled things have some sort of computing capability, no matter how rudimentary. And a device may have one or more of the other ingredients. We examine each of these ingredients in this section.

Sensors

A **sensor** measures some parameter of a physical, chemical, or biological entity and delivers an electronic signal proportional to the observed characteristic, either in the form of an analog voltage level or a digital signal. In both cases, the sensor output is typically input to a microcontroller or other management element.

The left side of Figure 14.2, adapted from a figure in *Middleware Architecture with Patterns and Frameworks* [KRAK09], shows the interface between a sensor and the controller for that sensor. A sensor may take the initiative in sending sensor data to the controller, either periodically or when a defined threshold is crossed; this is the active mode. Alternatively, or in addition, the sensor may operate in the passive mode, providing data when requested by the controller.

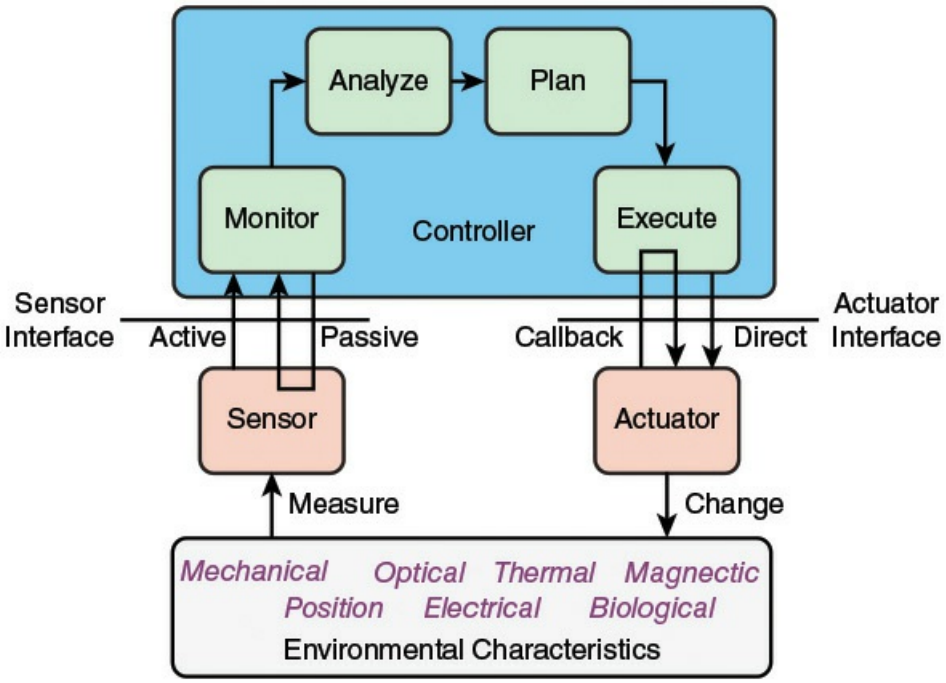


FIGURE 14.2 Interfaces for Sensors and Actuators

Types of Sensors

The variety of sensors used in IoT deployments is huge. Sensors may be extremely tiny, using nanotechnology, or quite substantial, such as a surveillance camera. Sensors may be deployed

individually or in very small numbers on the one hand, or in large numbers on the other. Table 14.2, from *Practical Electronics for Inventors* [SCHE13], lists various types of sensors, with examples of each type.

Category	What It Does	Device Examples
Position measuring devices	Designed to detect and respond to changes in angular position or in linear position of the device	Potentiometer, linear position sensor, hall effect position sensor, magnetoresistive angular, encoders (quadrature, incremental rotary, absolute rotary, optical)
Proximity, motion sensors	Designed to detect and respond to movement outside of the component but within the range of the sensor	Ultrasonic proximity, optical reflective, optical slotted, PIR (passive infrared), inductive proximity, capacitive proximity, reed switch, tactile switch
Inertial devices	Designed to detect and respond to changes in the physical movement of the sensor	Accelerometer, potentiometer, inclinometer, gyroscope, vibration sensor/switch, tilt sensor, Piezo shock sensor, LVDT/RVDT
Pressure/force	Designed to detect a force being exerted against it	IC barometer, strain gauge, pressure potentiometer, LVDT, silicon transducer, Piezoresistive sensor, capacitive transducer
Optical devices	Designed to detect the presence of light or a change in the amount of light on the sensor	LDR, photodiodes, phototransistors, photo interrupters, reflective sensors, IrDA transceiver, solar cells, LTV (light voltage) sensors

Image, camera devices	Designed to detect and change a viewable image into a digital signal	CMOS image sensor
Magnetic devices	Designed to detect and respond to the presence of a magnetic field	Hall effect sensor, magnetic switch, linear compass IC, Reed sensor
Media devices	Designed to detect and respond to the presence or the amount of a physical substance on the sensor	Gas, smoke, humidity, moisture, dust, float level, fluid flow
Current and voltage devices	Designed to detect and respond to changes in the flow of electricity in a wire or circuit	Hall effect current sensor, DC current sensor, AC current sensor, voltage transducer
Temperature	Designed to detect the amount of heat using different techniques and in different mediums	Thermistor NTC, thermistor PTC, resistance temp detectors (RTD)s, thermocouple, thermopile, digital IC, analog IC, infrared thermometer/pyrometer
Specialized	Designed to provide detection, measurement, or response in specialized situations, which also may include multiple functions	Audio Microphone, Geiger-Müller tube, chemical

TABLE 14.2 Types of Sensors

Precision, Accuracy, and Resolution

Two key concepts need to be distinguished in discussing sensors: precision and accuracy. **Accuracy** refers to how close a measurement comes to the truth, represented as a bull's eye in Figure 14.3. **Precision** refers to how close multiple measurements of the same physical quantity are to each other. If a sensor has low accuracy, this produces a systematic error. If a sensor has low precision, it produces a reproducibility error.

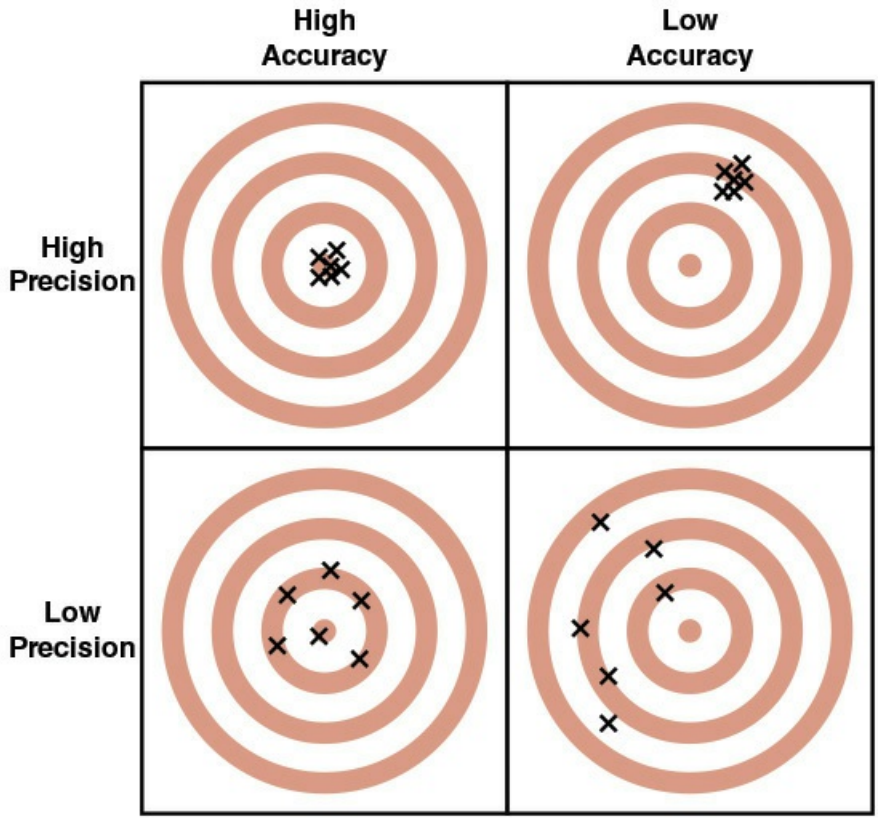


FIGURE 14.3 Precision and Accuracy

Related to precision is the concept of **resolution**. If a sensor has high precision, a very small change in the value of a physical quantity results in a very small change in the value of the sensor measurement. If the sensor output is digital, more bits are needed to represent the measurement to capture these small changes in the underlying physical parameter.

Actuators

An **actuator** receives an electronic signal from a controller and responds by interacting with its environment to produce an effect on some parameter of a physical, chemical, or biological entity. The right side of Figure 14.2 shows the interface between an actuator and the controller for that actuator. In the direct mode of operation, the controller sends a signal that activates the actuator. In callback mode, the actuator responds to the controller to report completion or a problem, and requests further instructions.

Actuators are generally classified as follows:

- **Hydraulic:** Hydraulic actuators consist of a cylinder or fluid motor that utilizes hydraulic power to facilitate mechanical process. The mechanical motion gives an output in terms of linear, rotary, or oscillatory motion.
- **Pneumatic:** Pneumatic actuators work on the same concept as hydraulic actuators except compressed gas is used instead of liquid. Energy, in the form of compressed gas, is converted into linear or rotary motion, depending on the type of actuator.

- **Electric:** Electric actuators are devices powered by motors that convert electrical energy to mechanical torque.
- **Mechanical:** Function through converting rotary motion to linear motion. Devices such as gears, rails, pulley, chain, and others are used to help convert the motion.

Microcontrollers

The “smart” in a smart device is provided by a deeply embedded microcontroller. This section defines some key terms and explains the concept of a microcontroller.

Embedded System

The term **embedded system** refers to the use of electronics and software within a product that has a specific function or set of functions, as opposed to a general-purpose computer, such as a laptop or desktop system. Hundreds of millions of computers are sold every year, including laptops, personal computers, workstations, servers, mainframes, and supercomputers. In contrast, tens of billions of microcrontrollers are produced each year that are embedded within larger devices. Today, many, perhaps most, devices that use electric power have an embedded computing system. It is likely that in the near future nearly all such devices will have embedded computing systems.

Types of devices with embedded systems are almost too numerous to list. Examples include cell phones, digital cameras, video cameras, calculators, microwave ovens, home security systems, washing machines, lighting systems, thermostats, printers, various automotive systems (for example, transmission control, cruise control, fuel injection, anti-lock brakes, and suspension systems), tennis rackets, toothbrushes, and numerous types of sensors and actuators in automated systems.

Often, embedded systems are tightly coupled to their environment. This can give rise to real-time constraints imposed by the need to interact with the environment. Constraints, such as required speeds of motion, required precision of measurement, and required time durations, dictate the timing of software operations. If multiple activities must be managed simultaneously, this imposes more complex real-time constraints.

Figure 14.4 shows in general terms an embedded system organization. In addition to the processor and memory, there are a number of elements that differ from the typical desktop or laptop computer:

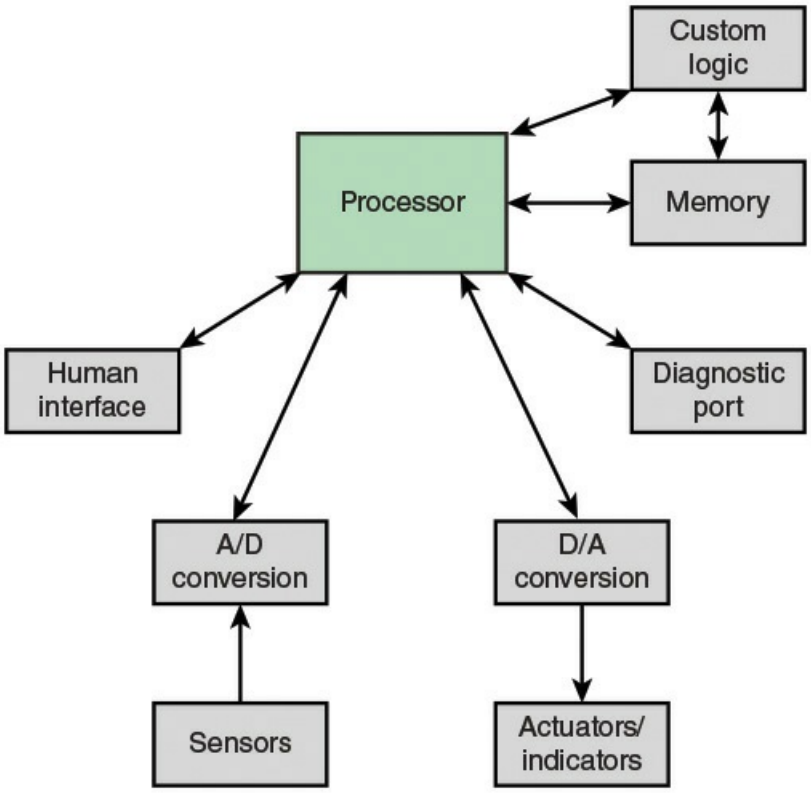


FIGURE 14.4 Possible Organization of an Embedded System

- There may be a variety of interfaces that enable the system to measure, manipulate, and otherwise interact with the external environment. Embedded systems often interact (sense, manipulate, and communicate) with the external world through sensors and actuators and hence are typically reactive systems; a reactive system is in continual interaction with the environment and executes at a pace determined by that environment.
- The human interface may be as simple as a flashing light or as complicated as real-time robotic vision. In many cases, there is no human interface.
- The diagnostic port may be used for diagnosing the system that is being controlled—not just for diagnosing the computer.
- Special-purpose field programmable (FPGA), application-specific (ASIC), or even nondigital hardware may be used to increase performance or reliability.
- Software often has a fixed function and is specific to the application.
- Efficiency is of paramount importance for embedded systems. They are optimized for energy, code size, execution time, weight and dimensions, and cost.

There are several noteworthy areas of similarity to general-purpose computer systems as well:

- Even with nominally fixed function software, the ability to field upgrades to fix bugs, improve security, and add functionality has become very important for embedded systems, and not just in consumer devices.
- One comparatively recent development has been of embedded system platforms that

support a wide variety of apps. Good examples of this are smartphones and audio/visual devices, such as smart TVs.

Application Processors versus Dedicated Processors

Application processors are defined by the processor's ability to execute complex operating systems, such as Linux, Android, and Chrome. Thus, the application processor is general-purpose in nature. A good example of the use of an embedded application processor is the smartphone. The embedded system is designed to support numerous apps and perform a wide variety of functions.

Most embedded systems employ a **dedicated processor**, which, as the name implies, is dedicated to one or a small number of specific tasks required by the host device. Because such an embedded system is dedicated to a specific task or tasks, the processor and associated components can be engineered to reduce size and cost.

Microprocessors

Early **microprocessor** chips included registers, an arithmetic/logic unit (ALU), and some sort of control unit or instruction processing logic. As transistor density increased, it became possible to increase the complexity of the instruction set architecture, and ultimately to add memory and more than one processor. Contemporary microprocessor chips include multiple processors, called cores, and a substantial amount of cache memory. However, as shown in Figure 14.5, a microprocessor chip includes only some of the elements that make up a computer system.

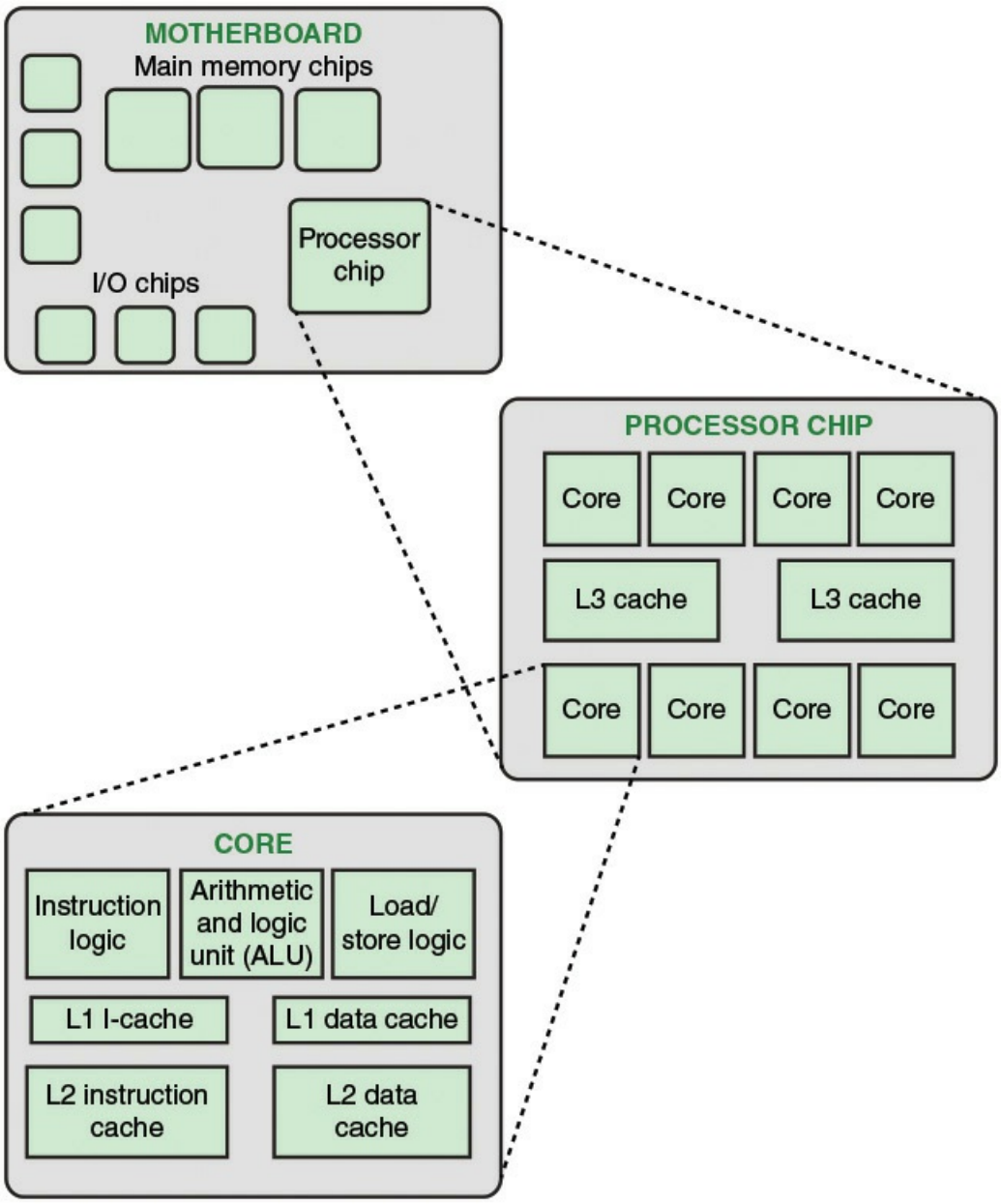


FIGURE 14.5 Simplified View of Major Elements of a Multicore Computer

Most computers, including embedded computers in smartphones and tablets, plus personal computers, laptops, and workstations, are housed on a motherboard. Before describing this arrangement, we need to define some terms. A **printed circuit board** is a rigid, flat board that holds and interconnects chips and other electronic components. The board is made of layers, typically two to ten, that interconnect components via copper pathways that are etched into the board. The main printed circuit board (PCB) in a computer is called a system board or **motherboard**, and smaller ones that plug into the slots in the main board are called expansion boards.

The most prominent elements on the motherboard are the chips. A **chip** is a single piece of semiconducting material, typically silicon, upon which electronic circuits and logic gates are

fabricated. The resulting product is referred to as an **integrated circuit**.

The motherboard contains a slot or socket for the processor chip, which typically contains multiple individual cores, in what is known as a *multicore processor*. There are also slots for memory chips, I/O controller chips, and other key computer components. For desktop computers, expansion slots enable the inclusion of more components on expansion boards. Thus, a modern motherboard connects only a few individual chip components, with each chip containing from a few thousand up to hundreds of millions of transistors.

Microcontrollers

A **microcontroller** chip makes a substantially different use of the logic space available. Figure 14.6 shows in general terms the elements typically found on a microcontroller chip. As shown, a microcontroller is a single chip that contains the core, nonvolatile memory for the program (ROM), volatile memory for input and output (RAM), a clock, and an I/O control unit. The processor portion of the microcontroller has a much lower silicon area than other microprocessors and much higher energy efficiency.

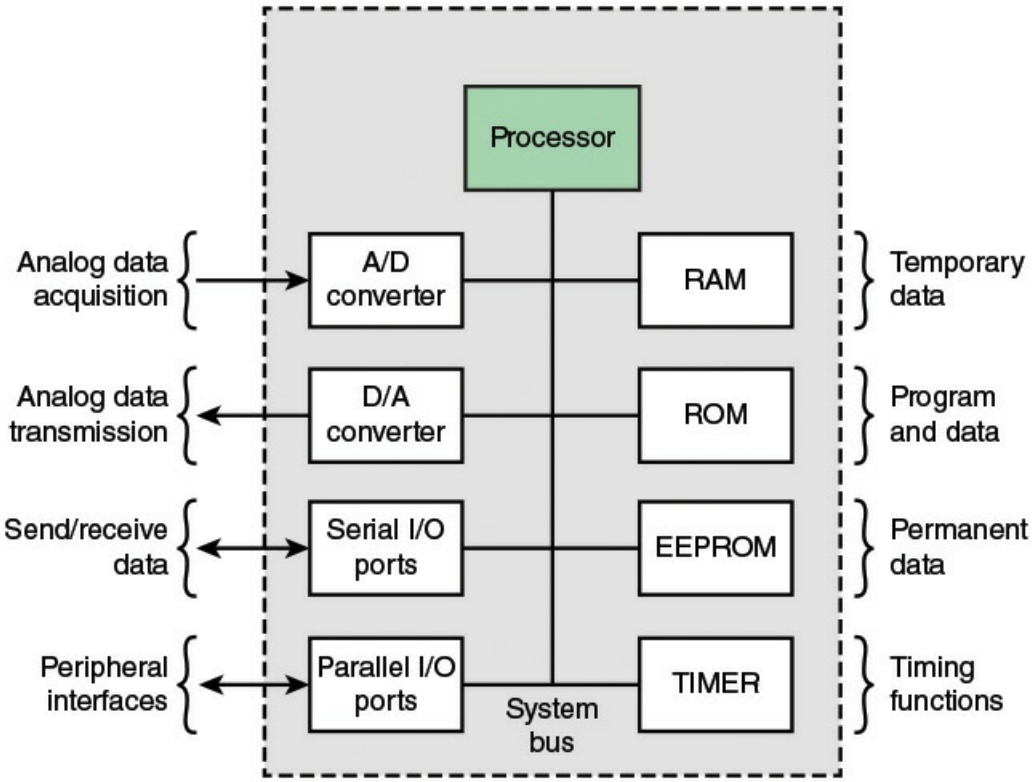


FIGURE 14.6 Typical Microcontroller Chip Elements

Also called a computer on a chip, billions of microcontroller units are embedded each year in myriad products from toys to appliances to automobiles. For example, a single vehicle can use 70 or more microcontrollers. Typically, especially for the smaller, less expensive microcontrollers, they are used as dedicated processors for specific tasks. For example, microcontrollers are heavily utilized in automation processes. By providing simple reactions to

input, they can control machinery, turn fans on and off, open and close valves, and so forth. They are integral parts of modern industrial technology and are among the most inexpensive ways to produce machinery that can handle extremely complex functionalities.

Microcontrollers come in a range of physical sizes and processing power. Processors range from 4-bit to 32-bit architectures. Microcontrollers tend to be much slower than microprocessors, typically operating in the megahertz (MHz) range rather than the gigahertz (GHz) speeds of microprocessors. Another typical feature of a microcontroller is that it does not provide for human interaction. The microcontroller is programmed for a specific task, embedded in its device, and executes as and when required.

Deeply Embedded Systems

A subset of embedded systems, and a quite numerous subset, is referred to as **deeply embedded systems**. Although this term is widely used in the technical and commercial literature, you will search the Internet in vain (at least the writer did) for a straightforward definition. Generally, we can say that a deeply embedded system has a processor whose behavior is difficult to observe both by the programmer and the user. A deeply embedded system uses a microcontroller rather than a microprocessor, is not programmable once the program logic for the device has been burned into ROM (read-only memory), and has no interaction with a user.

Deeply embedded systems are dedicated, single-purpose devices that detect something in the environment, perform a basic level of processing, and then do something with the results. Deeply embedded systems often have wireless capability and appear in networked configurations, such as networks of sensors deployed over a large area (for example, factory, agricultural field). The Internet of Things depends heavily on deeply embedded systems. Typically, deeply embedded systems have extreme resource constraints in terms of memory, processor size, time, and power consumption.

Transceivers

A **transceiver** contains the electronics needed to transmit and receive data. Most IoT devices contain a wireless transceiver, capable of communication using Wi-Fi, ZigBee, or some other wireless scheme.

Figure 14.7 is a simplified block diagram showing the basic elements of a transceiver. The upper part of the figure is the transmitter, which takes some analog or digital input signal as input. This signal is modulated onto a carrier frequency. This is done by a modulator whose input is the source signal plus a carrier wave generated by an oscillator. The resulting signal goes through one or more amplifiers and then is transmitted by an antenna.

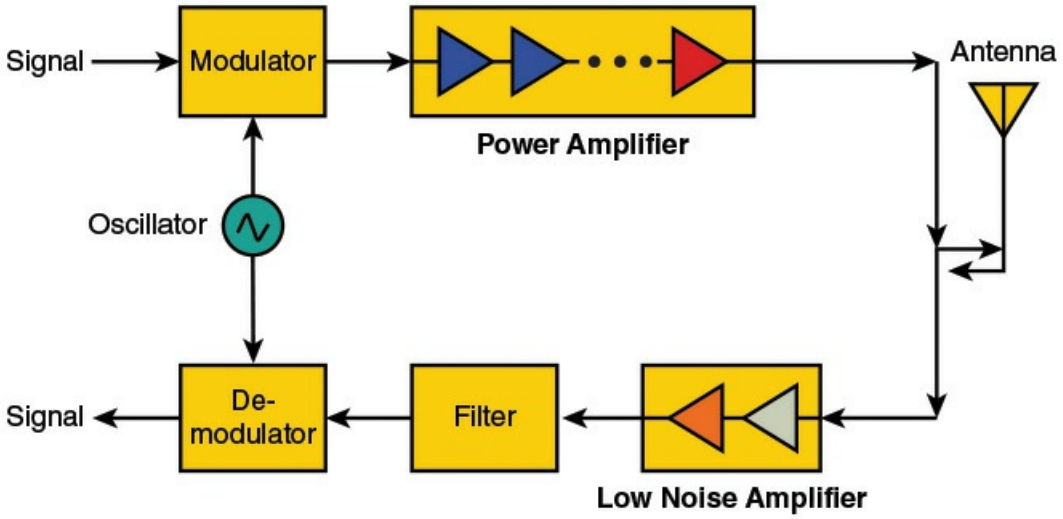


FIGURE 14.7 Simplified Transceiver Block Diagram

The lower part of Figure 14.7 is the receiver. The input to the receiver is the signal captured by the antenna. A low-noise amplifier (LNA) is an electronic amplifier used to amplify very weak signals (for example, captured by an antenna). The LNA is designed to boost the desired signal power while adding as little noise and distortion as possible. Following the LNA, a filter is used to eliminate or reduce unwanted noise and signal components. Then a demodulator converts the filter output to the desired baseband analog or digital signal.

RFID

Radio-frequency identification (RFID) technology, which uses radio waves to identify items, is increasingly becoming an enabling technology for IoT. The main elements of an RFID system are tags and readers. RFID tags are small programmable devices used for object, animal and human tracking. They come in a variety of shapes, sizes, functionalities, and costs. RFID readers acquire and sometimes rewrite information stored on RFID tags that come within operating range (a few inches up to several feet). Readers are usually connected to a computer system that records and formats the acquired information for further uses.

Applications

The range of applications of RFID is wide and ever expanding. Four major categories of application are tracking and identification, payment and stored-value systems, access control, and anticounterfeiting.

The most widespread use of RFID is for tracking and identification. Early use of RFID was for large high-value items such as train cars and shipping containers. As the price has dropped and the technology improved, this application has expanded dramatically. For example, millions of pets have implanted RFID devices allowing lost animals to be identified and returned to their owner. Another example: tracking and managing the billions of consumer items and components that flow through supply changes is a formidable task and there has been widespread adoption of RFID tags to simplify the task. To make this process as inexpensive and interoperable as

possible, standardized identification schemes have been developed, known as **electronic product codes (EPCs)**.

Another key area is payment and stored value systems. Electronic toll systems on highways are one example. Another is the use of electronic key "fobs" for payment at retail stores and entertainment venues.

Access control is another widespread application area. RFID proximity cards control building access at many companies and universities. Ski resorts and other leisure venues are also heavy users of this technology.

RFID also is effective as an anti-counterfeiting tool. Casinos use RFID tags on chips to prevent the use of counterfeit chips. The prescription drug industry uses RFID tags to cope with the counterfeit drug market. The tags are used to ensure the pedigree of drugs as they move through the supply chain and also to detect theft.

Here is a partial list of applications in these four areas:

- **Tracking and identification:**
- Large assets, for example, railway cars and shipping containers
- Livestock with rugged tags
- Pets with implanted tags
- Supply-chain management with EPC
- Inventory control with EPC
- Retail checkout with EPC
- Recycling and waste disposal
- Patient monitoring
- Tagging children at school
- Drivers' licenses and passports
- **Payment and stored-value systems:**
- Electronic toll systems
- Contact-less credit cards (for example, American Express Blue card)
- Stored-valued systems (for example, ExxonMobil Speedpass)
- Subway and bus passes
- Casino tokens and concert tickets
- **Access control:**
- Building access with proximity cards
- Ski lift passes
- Concert tickets
- Automobile ignition systems

- **Anticounterfeiting:**
- Casino tokens (for example, Wynn Casino Las Vegas)
- High-denomination currency notes
- Luxury goods (for example, Prada)
- Prescription drugs

Tags

Figure 14.8 shows the key elements of an RFID system. Primary wireless communication is between a tag and a reader. The reader retrieves identification information and, depending on the application, other information about the tagged item. The reader then communicates this to a computer system which includes an RFID-related database and RFID-related applications.

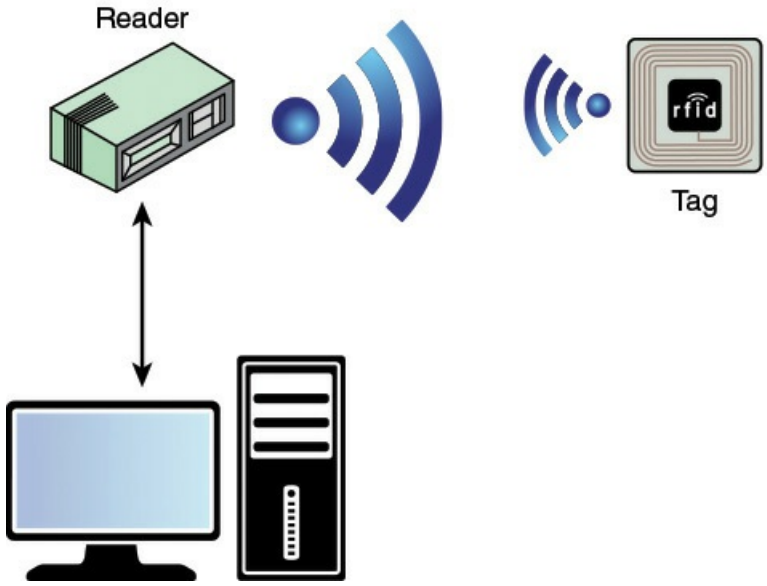


FIGURE 14.8 Elements of an RFID System

Figure 14.9 shows the two key components of a tag. The antenna is a metallic path in the tag whose layout depends on the size and shape of the tag and the operating frequency. Attached to the antenna is a simple microchip with very limited processing and nonvolatile storage.

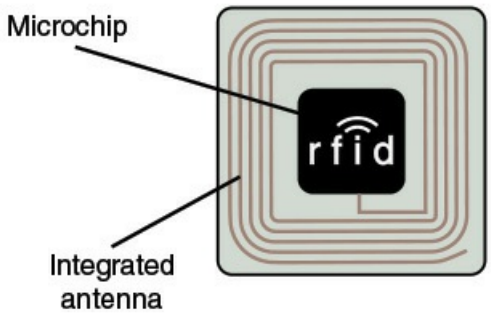


FIGURE 14.9 RFID Tag

RFID tags are classified as active, semi-passive, or passive (see Table 14.3). Active RFID tags produce their own signal from a battery, whereas passive RFID tags obtain their power from an RF signal impinging on the tag. Semi-passive tags do have a battery but behave like passive tags.

	Passive	Semi-Passive	Active
Power source	Harvesting RF energy	Battery	Battery
Required signal strength from reader to tag	High	Low	Low
Communication	Response only	Response only	Respond or initiate
Typical maximum passive read distance	10 m	> 100 m	> 100 m
Relative cost	Least expensive	More expensive	Most expensive
Example applications	EPC Proximity cards	Electronic tolls Pallet tracking	Large-asset tracking Livestock tracking

TABLE 14.3 Types of Tags

Active tags are considerably more expensive than passive tags and typically are physically larger. Active tags can generate a stronger signal and thus have a much further **read range** and can be read at high speed. Active RFIDs are the focus of the IEEE 802.15.4f standards effort.

For auto-ID and electronic key purposes, passive tags are the most common since they can be fabricated thin enough to be labels and are inexpensive. With a passive tag, the reader actually powers the tag, which then sends back its data to the reader.

Readers

RFID readers communicate with tags through an RF channel. The reader may obtain simple identification information or a more complex set of parameters. The dialogue is often a simple ping and response but may involve a more complex multiple exchange of information.

There is a wide variety of different readers in terms of functionality and basic operating style. In general, there are three categories of readers:

- **Fixed:** Fixed readers create portals for automated reading of tags as they pass by. Common applications are to read tags as the associated items enter a room, pass through warehouse dock doors, or travel on a conveyor line.
- **Mobile:** Mobile readers are hand-held devices with an RFID antenna and reader and some computing capability. They are made for manually reading tags on the move. They are useful for inventory applications.
- **Desktop:** This type of reader is typically attached to a PC or point-of-sale terminal and provides easy input.

Operating Frequency

True physical tag maximum read distance is determined by the individual RFID reader and antenna power, the chip used in the RFID tag, the material and thickness of material the tag is coated or covered with, the type of antenna the tag uses, the material the tag is attached to, and so on. The frequency range used by tag and reader is a limiting factor on read range. Table 14.4 lists standard frequencies and their respective passive read distances. Higher frequencies provide greater read range and the ability to transfer greater amounts of data. These frequencies can also be used for active tags. In addition, active tags can use the 433-MHz and 2.4-GHz bands with ranges in the hundreds of meters.

Frequency Range	Frequencies	Passive Read Distance
Low frequency (LF)	120–140 KHz	10–20 cm
High frequency (HF)	13.56 MHz	10–20 cm
Ultra-high frequency (UHF)	868–928 MHz	3 meters
Microwave	2.45 and 5.8 GHz	3 meters
Ultra-wide band (UWB)	3.1–10.6 GHz	10 meters

TABLE 14.4 Common RFID Operating Frequencies

Functionality

As the name suggests, the basic functionality of RFID is identification of tagged items. Tags may offer a number of other functionalities that are compatible with the RFID technology and systems. Table 14.5 lists six general classes defined by the standards group EPCglobal.

Class	Description
Class 0	UHF read-only, preprogrammed passive tag
Class 1	UHF or HF; write once, read many (WORM)
Class 2	Passive read-write tags that can be written to at any point in the supply chain
Class 3	Read-write with onboard sensors capable of recording parameters like temperature, pressure, and motion; can be semi-passive or active
Class 4	Read-write active tags with integrated transmitters; can communicate with other tags and readers
Class 5	Similar to Class 4 tags but with additional functionality; can provide power to other tags and communicate with devices other than readers

TABLE 14.5 Tag Functionality Classes

The Class 0 tags provide the most basic identification functionality, such as a product code or a unique identifier. The identifier is set when the tag is manufactured. These are fairly simple and inexpensive. Class 1 Tags are similar but provide the ability to set the identification information after manufacture time by the end user. Class 2 tags may be used as a logging device, in which the tagged item is logged into some system when first encountered and then provides identification information as needed. Class 3 tags provide two additional capabilities: read-write

memory and onboard sensor capability. A sensor tag may log and store environmental data without the aid of a reader. Many sensor tags may form a “sensor net” that monitors a physical area’s environmental properties. This may include temperature changes, rapid acceleration, changes in orientation, vibrations, the presence of biological or chemical agents, light, sound, and so on. Because they operate without a reader present, sensor tags must necessarily be semi-passive or active.

Class 4 tags, referred to as *motes*, or *smart dust*, are able to initiate communication with peers and form ad hoc networks. This leads to a wide variety of applications for small, inexpensive devices with limited communication range. Motes can be implanted or scattered over a region to collect data and pass it on from one to another to some central collection point. For example, a farmer, vineyard owner, or ecologist could equip motes with sensors that detect temperature, humidity, and so forth, making each mote a mini weather station. Scattered throughout a field, vineyard, or forest, these motes would allow the tracking of microclimates. This goes far beyond basic RFID functionality but is included by EPCglobal as a functional extension. Class 5 extends Class 4 to include the ability of one device to provide power to other tags and communicate with devices other than the reader. This opens up even more possibilities.

14.4 Key Terms

After completing this chapter, you should be able to define the following terms.

accuracy

actuators

application processor

dedicated processor

deeply embedded system

electronic product code (EPC)

embedded systems

fog computing

information technology (IT)

Internet of Things (IoT)

microcontrollers

microprocessor

operational technology (OT)

precision

radio-frequency identification (RFID)

RFID reader

read range

resolution

sensors

RFID tag

transceiver

14.5 References

KRAK09: Krakowiak, S. *Middleware Architecture with Patterns and* Frameworks. 2009. http://sardes.inrialpes.fr/%7Ekrakowia/MW-Book/

MCEW13: McEwen, A., and Cassimally, H. *Designing the Internet of Things*. New York: Wiley, 2013.

SCHE13: Scherz, P., and Monk, S. *Practical Electronics for Inventors*. New York: McGraw-Hill, 2013.

STAN14: Stankovic, J. "Research Directions for the Internet of Things." *Internet of Things Journal*, Vol. 1, No. 1, 2014.

Chapter 15. The Internet of Things: Architecture and Implementation

> Whenever logical processes of thought are employed—that is, whenever thought for a time runs along an accepted groove—there is an opportunity for the machine.
>
> —"As We May Think," Vannevar Bush, *The Atlantic*, July 1945

Chapter Objectives: **After studying this chapter, you should be able to**

- Compare and contrast the ITU-T and IoT World Forum IoT reference models.
- Describe the open source IoTivity IoT implementation.
- Describe the commercial ioBridge IoT implementation.

This chapter concludes the discussion of the Internet of Things (IoT). It begins with a description of two important IoT reference models, which together provide insight into the architecture and functioning of an IoT. The chapter then examines three IoT implementations, one open source and two commercial.

15.1 IoT Architecture

Given the complexity of IoT, it is useful to have an architecture that specifies the main elements and their interrelationship. An IoT architecture can have the following benefits:

- It provides the IT or network manager with a useful checklist with which to evaluate the functionality and completeness of vendor offerings.
- It provides guidance to developers as to which functions are needed in an IoT and how these functions work together.
- It can serve as a framework for standardization, promoting interoperability and cost reduction.

We begin this section with an overview of the IoT architecture developed by ITU-T. We then look at one developed by IoT World Forum. The latter architecture, developed by an industry group, offers a useful alternative framework for understanding the scope and functionality of IoT.

ITU-T IoT Reference Model

The ITU-T IoT reference model is defined in Y.2060, *Overview of the Internet of Things*, June 2012. Unlike most of the other IoT reference models and architectural models in the literature, the ITU-T model goes into detail about the actual physical components of the IoT ecosystem. This is a useful treatment because it makes visible the elements in the IoT ecosystem that must be interconnected, integrated, managed, and made available to applications. This detailed

specification of the ecosystem drives the requirements for the IoT capability.

An important insight provided by the model is that the IoT is in fact not a network of physical things. Rather, it is a network of devices that interact with physical things, together with application platforms, such as computers, tablets, and smartphones, that interact with these devices. So, we begin our overview of the ITU-T model with a discussion of devices.

Table 15.1 lists definitions of key terms used in Y.2060.

Term	Definition
Communication network	An infrastructure network that connects devices and applications, such as an IP-based network or internet.
Thing	An object of the physical world (physical things) or the information world (virtual things) that can be identified and integrated into communication networks.
Device	A piece of equipment with the mandatory capability of communication and the optional capabilities of sensing, actuation, data capture, data storage and data processing.
Data-carrying device	A device attached to a physical thing to indirectly connect the physical thing with the communication networks. Class 3, 4, and 5 radio-frequency identification (RFID) tags are examples.
Data-capturing device	A reader/writer device with the capability to interact with physical things. The interaction can happen indirectly via data-carrying devices, or directly via data carriers attached to the physical things.
Data carrier	A battery-free data carrying object attached to a physical thing that can provide information to a suitable data capturing device. This category includes bar codes and QR codes attached to physical things.
Sensing device	Detects or measures information related to the surrounding environment and convert it into digital electronic signals.
Actuating device	Converts digital electronic signals from the information networks into operations.
General device	A general device has embedded processing and communication capabilities and may communicate with the communication networks via wired or wireless technologies. General devices include equipment and appliances for different IoT application domains, such as industrial machines, home electrical appliances, and smartphones.
Gateway	A unit in the IoT that interconnects the devices with the communication networks. It performs the necessary translation between the protocols used in the communication networks and those used by devices.

TABLE 15.1 Y.2060 IoT Terminology

Devices