## CHAPTER CONTENTS

CHAPTER 1

# Introduction

We begin this book with an introduction to the analysis, architecture, and design processes. Many concepts and terms that are used throughout this book are introduced and defined in this chapter. Some of these concepts may be new to you while others are presented in a different light. Glossaries of terms and acronyms are provided at the end of this book for easy reference.

## 1.1 Objectives

The objectives of this chapter are to:

1. Introduce the fundamental concepts of this book; the processes of network analysis, architecture, and design; systems and services; as well as their characteristics
2. Define several terms used throughout the book
3. Prepare the reader for the analysis process

## 1.2 Preparation

To be able to understand and apply the concepts in this chapter, you should be familiar with basic networking concepts, including functions and features of the TCP/IP protocol suite; technologies such as Ethernet, Fast Ethernet, Gigabit Ethernet, asynchronous transfer mode (ATM), synchronous optical network (SONET), and frame relay; and the basics of network performance. Some recommended sources of information include:

- *TCP/IP Illustrated,* volume 1, by W. Richard Stevens, Addison-Wesley Publishing, January 1994
- *Essentials of ATM Networks and Services,* by Oliver Chukwudi Ibe, Addison-Wesley Publishing, January 1997
- *ATM and SONET Basics,* by George Dobrowski and Donald Grise, APDG Publishing, January 2001
- *Switched, Fast, and Gigabit Ethernet,* by Sean Riley and Robert A. Breyer, New Riders Publishing, January 1999
- *Frame Relay: Technology and Practice,* by Jeff T. Buckwalter, Addison-Wesley Publishing, December 1999

## 1.3 Background

Network analysis, architecture, and design have traditionally been considered art, combining an individual's particular rules on evaluating and choosing network technologies; knowledge about how technologies, services, and protocols can be meaningfully combined; experience in what works and what doesn't; along with (often arbitrary) selections of network architectures. However, as in other types of art, success of a particular network architecture or design often depends primarily on who is doing the work, with results that are rarely reproducible. This may have been acceptable in the early days of networking, when networks were more of a hobby than a critical resource and did not support revenue generation. Today, however, networks are embedded within our work and home lives. They are considered "mission-critical" to corporate success and provide near real-time access to information throughout the world. As such, a network's architecture and design must be logical, reproducible, and defensible. This premise is the foundation for this book.

Traditionally, network analysis, architecture, and design have been based on developing and applying a set of rules for the network. In developing a set of rules, an individual may draw from experience or from general rules such as the 80/20 rule (where 80% of a network's traffic is local, and 20% is remote) or the adage "bridge when you can, route when you must" (bridging being simpler, easier, and cheaper at the time). Both of these rules have been modified, as we will see later in the book. Such rules were useful when there weren't many choices in network technologies, services, and interconnection strategies and when the differences between choices were clearly understood. But times have changed, and our notion of architecting and designing networks must adapt to the variety of options now available to us, the many different types of services we can offer to our end customers (users), and the subtleties between technologies.

Network analysis, architecture, and design have traditionally focused on *capacity planning,* which is estimating capacity (also known as bandwidth) required in the network to accommodate most short- and long-term traffic fluctuations. As network traffic grows, this bandwidth "buffer" provided by capacity planning reduces and customers experience problems related to congestion. Some common solutions are to "throw bandwidth at the problem," increasing the bandwidth buffer to (hopefully) reduce congestion, or to overengineer the network, intentionally adding capacity to accommodate network fluctuations. This approach often does not work, as network bandwidth is only one component of network resources that must be considered. We now also need to consider how delay performance can be optimized. And, in many cases, network reliability, maintainability, and availability, also known as RMA (more on this in Section 1.9), are more important than overall throughput.

In this book we will explore how the analysis, architecture, and design processes have changed and how they continue to change. We will discuss how these processes work together in engineering a new or existing network. We will approach networks from a different perspective—as a system providing services to customers—and we discuss how networks can be architected and designed to provide many different types of services to its customers (users). In taking this approach, we will emphasize network analysis, which helps us understand what will be required of a network in supporting its customers and their applications and devices. As we will see, these processes require an investment of time and effort, but the return on investment is significant. They are powerful tools that can help you build better networks.

This book begins by applying a systems methodology to networking. This methodology is relatively new, and you will learn a number of useful definitions for network analysis, architecture, and design. The rest of the book is logically divided into three sections. The first section covers the analysis process, consisting of how to develop requirements, flow, and risk analyses. The analysis process prepares you for network architecture and design, discussed in the following two sections. The second section discusses how to understand the relationships between functions within your network and how to develop an architecture describing these relationships. In the final section the network architecture is used as input for the network design process, where technology selection and connectivity are determined. The information flows between network analysis, architecture, and design are presented in Figure 1.1.

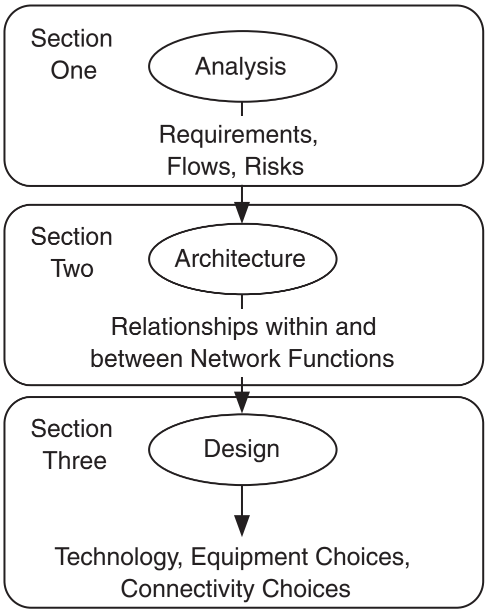
Section One
Analysis
Requirements, Flows, Risks
Section Two
Architecture
Relationships within and between Network Functions
Section Three
Design
Technology, Equipment Choices, Connectivity Choices

**FIGURE 1.1** Information flows between network analysis, architecture, and design.

Network analysis, architecture, and design will help you identify and apply the services and performance levels that your network must satisfy. Through these processes, you will be able to architect and design your network to provide the desired services and performance levels and will be able to choose the appropriate network technologies and interconnection strategies to meet your architectural and design goals.

## 1.4 Overview of Analysis, Architecture, and Design Processes

What are network analysis, architecture, and design? *Network analysis* is about studying network components (from network devices such as switches and routers, to requirements and performance levels) and their inputs and outputs to understand network behavior under various situations (Figure 1.2). Network analysis defines, determines, and describes relationships between network components for many conditions. In analyzing a network, we will examine the state of the existing network,

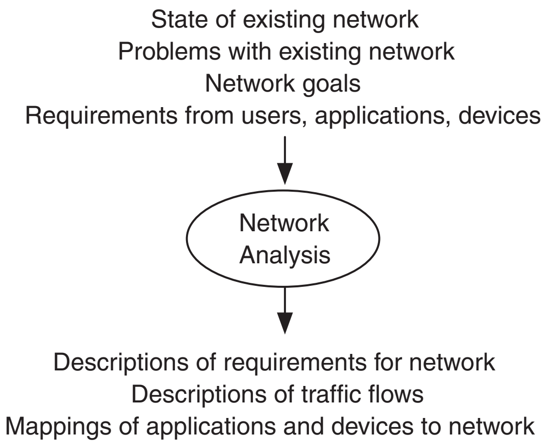
State of existing network
Problems with existing network
Network goals
Requirements from users, applications, devices
Network Analysis
Descriptions of requirements for network
Descriptions of traffic flows
Mappings of applications and devices to network

**FIGURE 1.2** Inputs and outputs to network analysis process.

including whatever problems it is having. We will also examine network goals and requirements, traffic flows, and user and application behavior. Analysis results include descriptions of requirements and traffic flows, as well as mappings of users, applications, and devices within the *network environment* (everything that is external to the network, for example, users, applications, devices, buildings, business environment).

Network analysis helps us understand what problems we are trying to solve, and in the process, we compile information that will be used in developing an architecture and design.

*Network architecture* uses this information to develop a high-level, end-to-end structure for the network. A network architecture develops the major network functions (e.g., addressing/routing, network management, performance, security) as architectural components that will be brought together to form the network; goals for the architecture; and sets of interactions, trade-offs, dependencies, and constraints (these will be described in detail later) that will be used to optimize the architecture. There usually is not a single "right" architecture or design for a network; instead there are several that will work, some better than others. Architecture and design focus on finding the best network (optimized across several parameters) for your customer. Network architecture also provides a set of guidelines that can be used to formulate the technical design of a network.

Results of the network architecture process include a *reference network architecture* (or just reference architecture), which is a description of the complete network, considering all of its architectural components. It is a compilation of the relationships developed during the network architecture process. Results also include descriptions

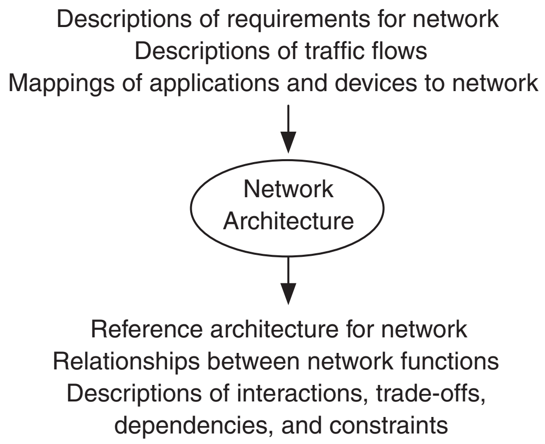
Descriptions of requirements for network
Descriptions of traffic flows
Mappings of applications and devices to network
Network Architecture
Reference architecture for network
Relationships between network functions
Descriptions of interactions, trade-offs, dependencies, and constraints

**FIGURE 1.3** Inputs and outputs to network architecture process.

of the interactions, trade-offs, dependencies, and constraints used to arrive at that architecture.

*Network design* provides *physical* detail to the reference architecture. During network design, you evaluate and choose technologies for each area of the network and develop strategies to connect these technologies across the network. You will learn how to set design goals, such as minimizing network costs or maximizing performance, as well as how to achieve these goals, through mapping network performance and function to your design goals and evaluating your design against its goals to recognize when the design varies significantly from these goals. Network design is also about applying the trade-offs, dependencies, and constraints developed as part of the network architecture. Trade-offs, such as cost versus performance or simplicity versus function, occur throughout the design process, and a large part of network design concerns recognizing such trade-offs (as well as interactions, dependencies, and constraints) and optimizing the design between them. As part of the design process, you will also learn how to develop evaluation criteria for your designs.

As we will see throughout the remainder of this book, network analysis, architecture, and design combine several things—requirements, traffic flows, architectural and design goals, interactions, trade-offs, dependencies, constraints, and evaluation criteria—to optimize a network's architecture and design across several parameters. These parameters are chosen and analyzed during the analysis process and prioritized and evaluated during the architecture and design processes. Upon completion of these processes, you should have a thorough understanding of the network and plenty of documentation to take you forward to implementation, testing, and integration.

**Example 1.1.** A network's architecture and design are analogous to the architecture and design of a home. Both the network and home architecture describe the major functional components of each (for the network, network management, addressing and routing, security and privacy, and performance; for the home, plumbing, electrical, HVAC [heating, vacuum, air conditioning], framing) and the relationships between them (for the network, interactions, dependencies, trade-offs, and constraints; for the home, where each component is placed relative to one another). The network and home designs are also similar in that they both provide physical detail to the architecture. For the network, this means where major network devices are located and, for the home, where ducts, outlets, faucets, drains, and so forth are located.

### 1.4.1 Hierarchy and Interconnectivity

All of these processes center around two important characteristics of networks: their levels of hierarchy and interconnectivity. *Hierarchy* is the degree of concentration of networks or traffic flows at interconnection points within the network, as well as the number of tiers of interconnection points within the network. In general, as networks grow in size and numbers of users, applications, and devices increase, hierarchies provide separation and structure within the network. Hierarchies are important because they help us in determining the sizes of networks, including routing and addressing configurations, and the scaling of network technologies, performance, and service levels. A key concept of this book is understanding these hierarchies, learning how and where they will occur, and learning how to take advantage of them.

Along with hierarchy, there must be some consideration for the degree of *interconnectivity* (or redundancy) in the network design. As hierarchy provides structure in the network, interconnectivity balances this structure by interconnecting the network at different levels in the design to provide greater performance through parts of the network. Interconnectivity is important in that it provides a mechanism to achieve performance within a hierarchical structure. The dynamic between hierarchy and interconnectivity is perhaps one of the most fundamental trade-offs in network architecture and design, and it shows up several times in the analysis, architecture, and design processes.

Hierarchy and interconnectivity may be a bit confusing at this point, but they will become clearer as we progress through the book. Hierarchy is fundamental to networking (as it is throughout nature): It provides a separation of the network into segments. These segments may be separate, smaller networks, or subnets, or broadcast domains. This is necessary when the amount of traffic on the network grows beyond the capacity of the network or when interactions between devices on the network result in congestion (e.g., broadcast storms).

Figure 1.4 illustrates levels of hierarchy and interconnectivity in a network. This is a typical tree structure for a network, with circles representing networks or routers

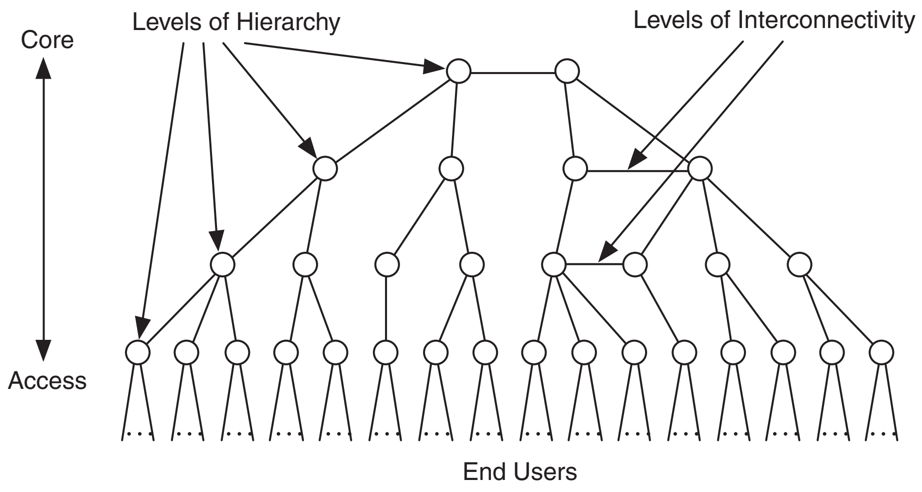

**FIGURE 1.4** Hierarchy and interconnectivity in a network.

and lines representing the communications links between networks and/or routers. In this figure there are four levels of hierarchy, from core (or backbone) networks to access networks closest to users. Note that the end points of this tree (commonly refereed to as leaves; they represent the end networks, devices, or users) all occur at the same level of hierarchy. This does not have to be the case; indeed, in most networks there are leaves at most levels of hierarchy.

An example of adding hierarchy to a network is changing from a flat (bridged or layer 2 switched) structure to a routed structure. This may be done to reduce the size of the broadcast domain or the number of devices that are reached by a broadcast message. Adding routing to the network breaks a broadcast domain into a number of smaller broadcast domains, and traffic flows are concentrated at routers. Figure 1.5 shows this scenario.

A content delivery network (CDN) is an example of adding interconnectivity to a network. A CDN bypasses the core of a network, where congestion is most likely to occur, and directly connects devices or networks lower in the hierarchy (Figure 1.6). This provides better, more predictable performance but can also affect the network hierarchy by modifying its routing behavior.

## 1.4.2 Importance of Network Analysis

The importance of network analysis is emphasized in this book because experience has shown that networking personnel find it extremely valuable–once they are con-

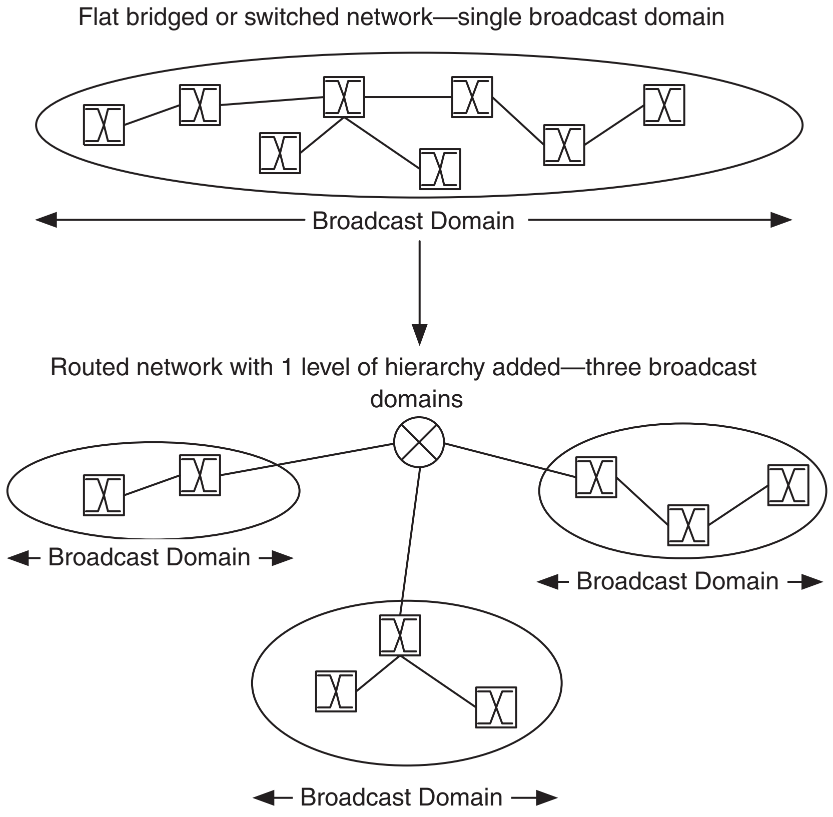
Flat bridged or switched network—single broadcast domain
Broadcast Domain
Routed network with 1 level of hierarchy added—three broadcast domains
Broadcast Domain
Broadcast Domain
Broadcast Domain

**FIGURE 1.5** Hierarchy added to a network.

vinced of its importance. Analysis takes work, and when you know that there will be a payoff, you will be more likely to do that work.

In this book you will learn how to gather and analyze network requirements and traffic flows. *Network requirements* are requests for capabilities in the network, usually in terms of performance and function, which are necessary for the success of that network. Network requirements can be gathered and/or derived from customers, applications, and devices. Such requirements are fundamental to a network's architecture and design–they form the basis for customer expectations and satisfaction. Requirements, in conjunction with measurements on the existing network (if there is one), are used to derive traffic flows (sets of network traffic that have some common attributes, such as source/destination address, information type, routing, or other end-to-end information). Analysis of these flows imparts location and directional information onto requirements. This is where performance requirements and architecture

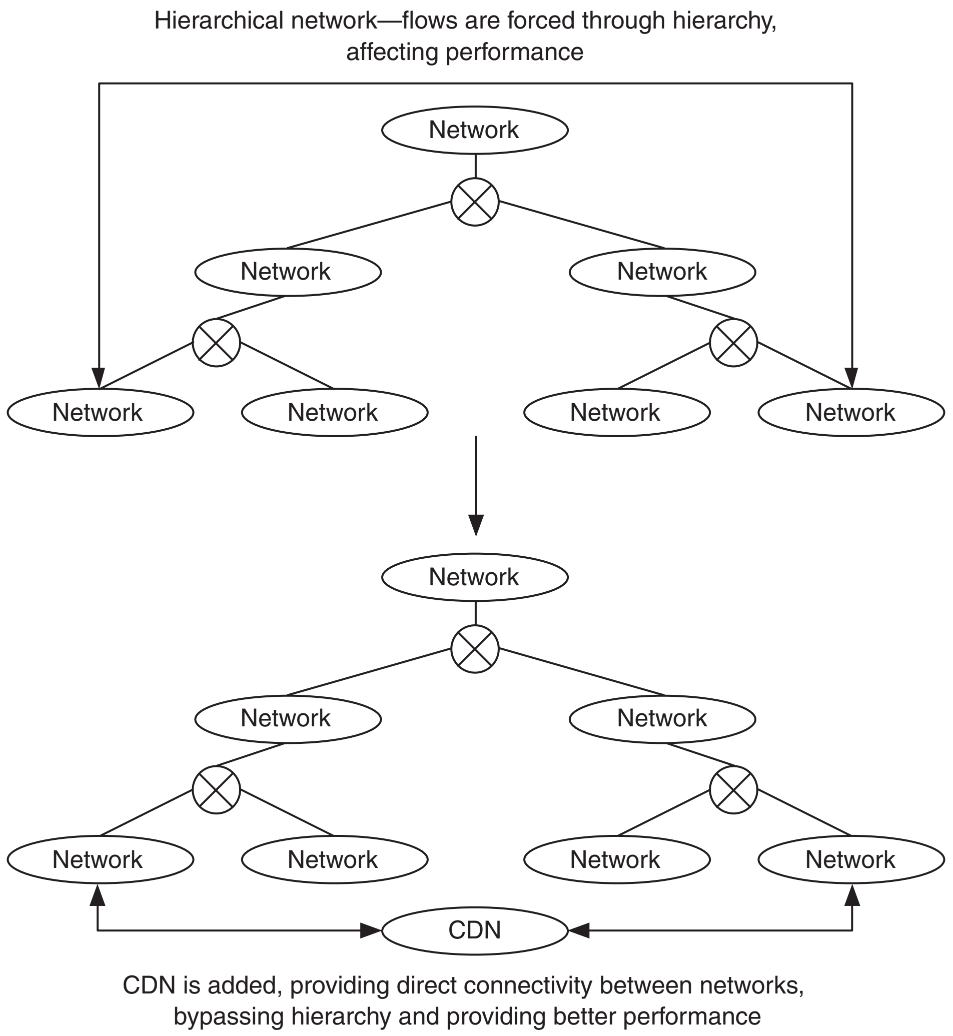
Hierarchical network—flows are forced through hierarchy, affecting performance
Network
Network
Network
Network
Network
Network
Network
Network
Network
Network
Network
Network
Network
Network
CDN
CDN is added, providing direct connectivity between networks, bypassing hierarchy and providing better performance

**FIGURE 1.6** Interconnectivity added to a network.

start to converge and is often the point in these processes where one can begin to see where "hot spots"—focal points for network performance—will appear in the network. As we will see, evaluating security risks is also part of the analysis process.

Results of the analysis process, the requirements and flow specifications, are then used as input for both network architecture and design. In developing the network architecture, a number of component architectures, targeting particular functions of the network, will be evaluated. Desired component architectures will then be combined into the reference architecture, which will provide a high-level view of your network. This high-level view is then physically detailed during the network design process.

Network analysis is important in that it helps us understand the complexity and nuances of each network and the systems they support. Analysis also provides data upon which various decisions are made, and these data can and should be documented as part of an audit trail for the architecture and design processes. Such data help ensure that the resulting architecture and design are defensible.

### *Understanding Network and System Complexity*

In general, networks and the systems that they support are becoming increasingly complex. Part of this complexity lies in the sophistication of the capabilities provided by that network. Consider, for example, how services can be incorporated into a current state-of-the-art network. Infrastructure capacity planning, which often includes traffic overengineering, may now be expanded to include support for delay-constrained applications and may contain a variety of capacity and delay control mechanisms, such as traffic shaping, quality of service at multiple levels in the network, service-level agreements to couple services to customers, and policies to govern and implement service throughout the network. (Note that *quality of service* refers to determining, setting, and acting on priority levels for traffic flows. A *service-level agreement* is an informal or formal contract between a provider and user that defines the terms of the provider's responsibility to the user and the type and extent of accountability if those responsibilities are not met. Finally, *policies* are high-level statements about how network resources are to be allocated among users.) Analysis of these mechanisms—how they work and interoperate—will be covered in detail later in this book.

Network and system complexity is nonlinear. Network optimization must consider competing and often conflicting needs. In addition, multiple groups with differing ideas and desires (e.g., users, corporate management, network staff) influence the network design. The network is either designed by committee or through a systematic approach that the groups can agree on.

Networks have evolved to incorporate more sophisticated capabilities. Early (first-generation) networks focused on supporting basic connectivity between devices and on how to scale networks to support growing numbers of users (e.g., segmenting networks using bridges or routers). Second-generation networks focused on interoperability to expand the scope and scale of networks to allow connections between multiple disparate networks. We are currently at the stage in network evolution where service delivery is important to the success of users and their applications. This is the generation of network services, which can be considered the third generation

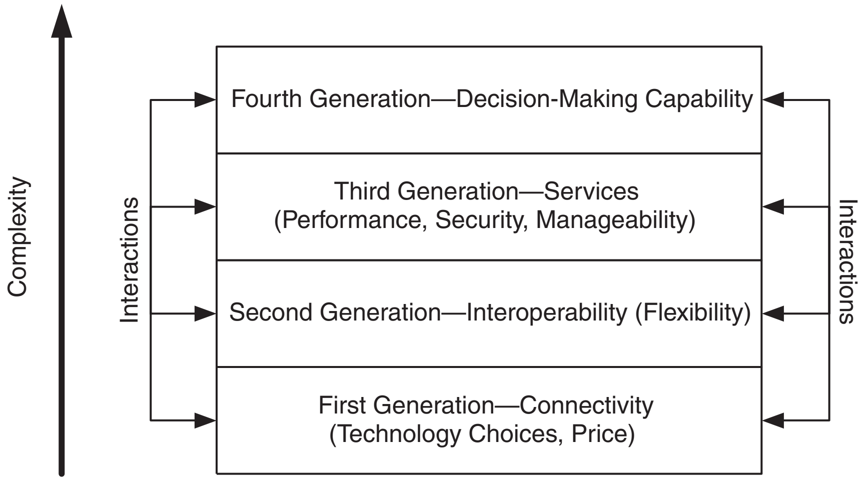
Complexity
Interactions
Fourth Generation—Decision-Making Capability
Third Generation—Services
(Performance, Security, Manageability)
Second Generation—Interoperability (Flexibility)
First Generation—Connectivity
(Technology Choices, Price)
Interactions

**FIGURE 1.7** Generations of networking.

of networking. Figure 1.7 illustrates the various generations of networking and their interactions.

We are beginning to see steps toward next-generation capabilities, such as rudimentary decision making within the network. It may be expected that components of the network will evolve to become self-configurable and manageable, especially for those networks that must be configured or administered by end users (e.g., telecommuters, users of mobility/portability services). Indeed, this will become necessary as the complexity and performance of networks increase and as services offered by networks become more sophisticated.

Along with networks, users, applications, and devices are also evolving more sophisticated capabilities. An example of this is the dynamic between hierarchy and interconnectivity as can be seen in the current Internet. As application and device traffic flows evolve to incorporate information regarding quality, performance, and cost (e.g., real-time streaming media), it is expected that these characteristics can be used to ensure paths through the Internet that will support high performance or high-quality delivery of such traffic. Hierarchy in the Internet often forces traffic flows through nonoptimal paths, hindering high-performance, high-quality delivery. Figure 1.8 shows a hierarchy of multiple levels of networks from core (or backbone) network providers to access network providers. Traffic flows between end users may travel across several levels of this hierarchy.

To counteract the impact of hierarchy, interconnectivity can be introduced into the Internet at strategic points, providing shortcuts that bypass parts of the Internet.

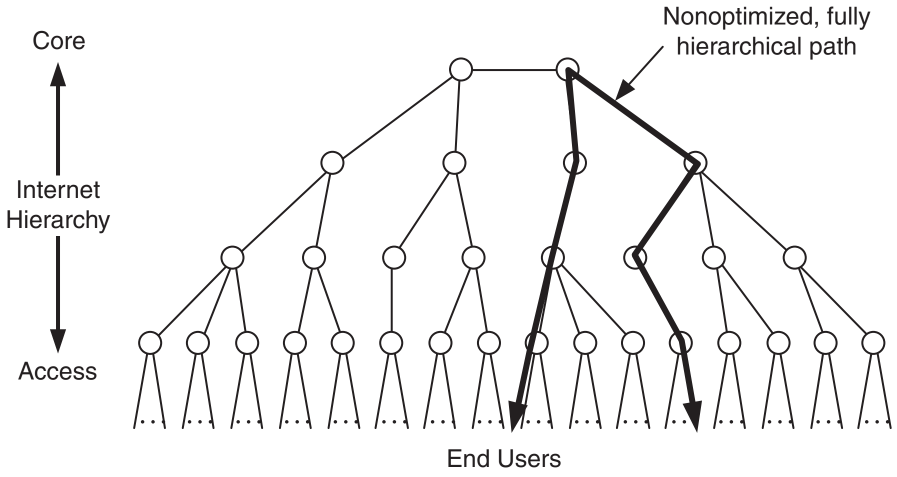

**FIGURE 1.8** Hierarchy and traffic flow.

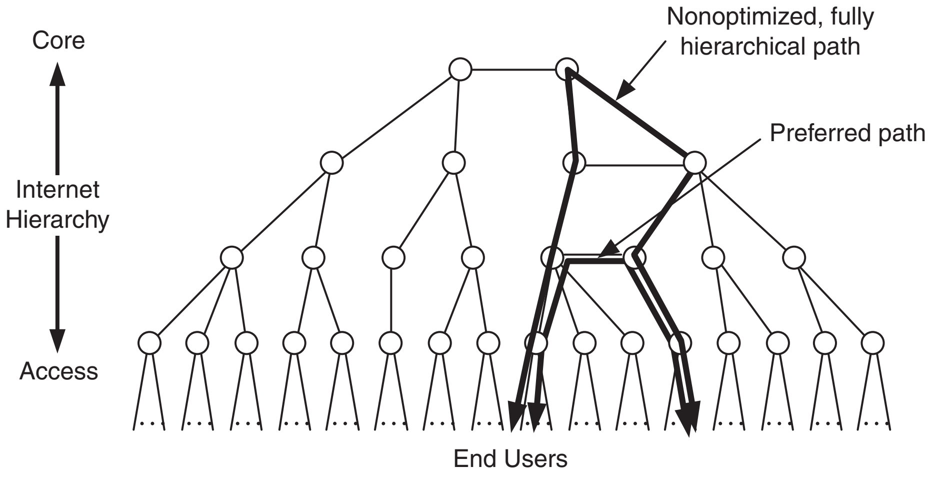

**FIGURE 1.9** Interconnectivity added to optimize traffic flow.

The result is that, for some select flows, paths are optimized for high-performance, high-quality delivery (Figure 1.9). The dynamic between hierarchy and interconnectivity exists in all networks to some degree, and part of the analysis process is determining where and how to apply it. In Figure 1.9, connections are added between networks at the same level of hierarchy, in essence providing a "shortcut" or "bypass" around part of the Internet, resulting in better performance characteristics.

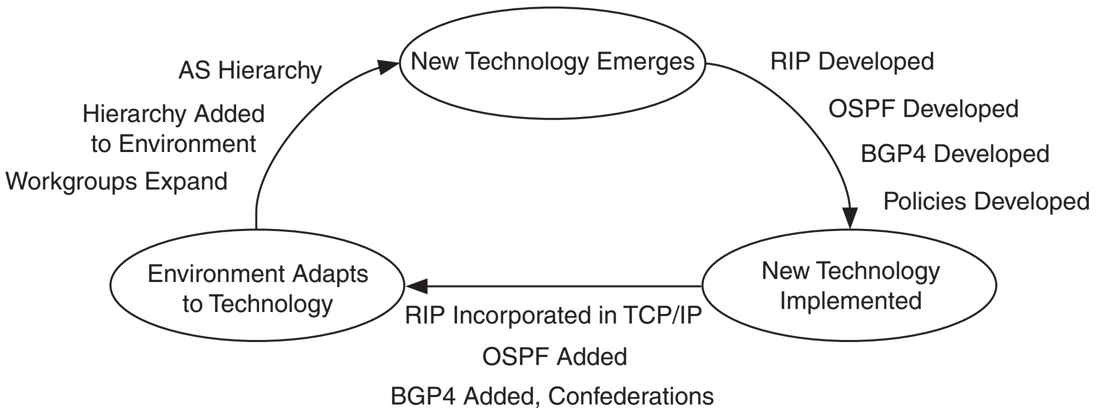
New Technology Emerges
RIP Developed
OSPF Developed
BGP4 Developed
Policies Developed
New Technology Implemented
RIP Incorporated in TCP/IP
OSPF Added
BGP4 Added, Confederations
Environment Adapts to Technology
Workgroups Expand
Hierarchy Added to Environment
AS Hierarchy

**FIGURE 1.10** Routing evolution.

Analysis helps us understand how technologies influence networks, users, applications, and devices (and vice versa). This is important in understanding how users of the network will adapt to the network, which affects the overall life cycle of the network. Consider, for example, the evolution of routing protocols, shown in Figure 1.10. Although the Routing Information Protocol (RIP), an Interior Gateway Protocol (IGP) deployed as part of early TCP/IP releases, was simple and easy to use, its limitations were stressed as networks expanded to accommodate larger groups of users (workgroups), even groups of networks (autonomous system [AS]). Routing technology adapted by adding hierarchy to routing, in terms of new IGPs such as Open Shortest Path First (OSPF), as well as in development of Exterior Gateway Protocols (EGPs) such as Border Gateway Protocol (BGP), which can accommodate hierarchy in groups of networks (AS hierarchy). This process continues today as high-level policies are being introduced to control routing at a level above IGPs or EGPs, and BGP4 introduces hierarchy through grouping BGP4 routers into confederations. We will discuss routing protocols in detail in the addressing and routing architecture (see Chapter 6).

Similarly, users, applications, and devices also influence their environment. As new, upgraded, or different users, applications, and devices are added to a network, the requirements on that network may change. The analysis process should examine how the impact of high-end compute and applications servers, data storage, analysis and archival systems, and specialized environment-specific devices such as video cameras or medical equipment changes the network.

Finally, the analysis process helps us understand the forces and changes at work within the system (network and its users, applications, and devices). Networks are highly dynamic, changing the rest of the system and being changed by the system. Some of the factors leading to change in the system include usage behavior and patterns; what, how, where, and when each user impacts the system; the emergence of new capabilities, for example, optical switching, faster central processing units (CPUs) and cheaper memory; and changes in scope and scale of the environment, including consolidation and outsourcing.

### *Architecture and Design Defensibility*

An important (and often overlooked) part of network analysis is that, when documented properly, analysis should provide information about decision making in the network architecture and design processes. During the analysis process, we are gathering data that can be used to determine which architectural and design decisions need to be made, details regarding each decision (including reasons for each decision), dependencies between decisions, and any background material used to arrive at these decisions.

Data from network analysis, along with any decisions made during the process, can be documented to form an *audit trail,* that is, the set of documents, data, and decisions, for the architecture and design. Audit trails are useful in describing and defending a particular network architecture or design. An audit trail helps address questions such as "Why did you choose that technology?" "Why doesn't this new network perform as expected?" or "Why did this network cost so much?" Having documented your analysis of the existing network, the problems to be addressed, requirements for the new network, and all decisions made regarding that network, you will be able to answer questions at any time about your new network.

Decisions made regarding the network architecture and design need to be defensible from several perspectives: budgetary, to ensure that network costs are within budget or to provide good reason why a budget has been exceeded; schedule, to ensure that time frames for development, installation, testing, and operations are being met; and resources, such as personnel or equipment, to ensure that the customer has everything necessary to build and operate this network.

An audit trail is also useful as a historical document about the network. Over time, after the network is made operational, new network personnel can review this document to understand the logic behind the way that network was designed. Ideally, this document should be periodically reviewed, with new information added regarding changes to the network. Thus, an audit trail becomes a history for that network.

Experience shows that the set of documents, data, and decisions in an audit trail can be vital in making day-to-day tactical design decisions throughout the project. The investment in time at this phase of the project can save large amounts of time and resources later in the project.

The Web is a great tool to use in building an audit trail. Since an audit trail contains information about the old network, the new network, and decisions made about the new network, having this information easily accessible by those who use the network makes a lot of sense. Putting such information on internal Web pages allows easy access by everyone, and changes or additions to the audit trail can be seen immediately. Although there may be some information that your customer might not want everyone to view, such as the results of a risk analysis, most information usually can be accessible to everyone. For information that is restricted (need-to-know), hidden and password-protected Web pages can be used. Of course, when putting sensitive information at a common location, such as a Web site, sufficient security from outside attacks (hackers) should be provided.

An audit trail can be developed using standard word processing and spreadsheet software tools, and software tools specialized for this purpose are available. Requirements, problem definitions, goals, decisions, and all background data are entered into the audit trail, and all information is time-stamped. Examples of audit trail information are presented later in this book.

### 1.4.3 Model for Network Analysis, Architecture, and Design

Networking traditionally has had little or no basis in analysis or architectural development, with designers often either relying on technologies that they are most familiar with or those that are new or popular or being influenced by vendors/or consultants. There are serious problems with this traditional approach, particularly that decisions may be made without the due diligence of analysis or architectural development and that such decisions made during the early phases of the project are uninformed.

As a result, there may not be an audit trail for the architecture and design; therefore, the architecture and design are not defensible. In addition, such an architecture/design may lack consistency in its technological approach. Lacking data from analysis and architecture, we may not have a basis for making technology comparisons and trade-offs. Therefore, network analysis, architecture, and design are fundamental to the development of a network.

Network analysis, architecture, and design are similar to other development processes in that they address the following areas:

- Defining the problems to be addressed
- Establishing and managing customer expectations
- Monitoring the existing network, system, and its environment

- Analyzing data
- Developing a set of options to solve problems
- Evaluating and optimizing options based on various trade-offs
- Selecting one or more options
- Planning the implementation

In defining the problems to be addressed, this should be a quick evaluation of the environment and project, in essence performing a sanity check on the task at hand, as well as determining the size and scope of the problems, determining that you are working on the right problems, and checking the levels of difficulty anticipated in the technologies, potential architectures and designs, administration, management, and politics in that environment. As you size up the problems faced in this project, you should begin to estimate the level of resources needed (e.g., budget, schedule, personnel). You should also develop your own view of the problems affecting that environment. You may find that, from your analysis of the situation, your definition of the problems may differ from the customer's definition. Depending on how far apart your definitions are, you may need to adjust your customer's expectations about the project.

---

**Example 1.2.** Once, in performing an analysis on a customer's metropolitan-area network (MAN), I realized that the problem was not what the customer thought—that the technology chosen at that time, switched multimegabit data service (SMDS), and the routing protocol (OSPF) were not working properly together. Rather the problem was that the network personnel had forgotten to connect any of their LANs to the MAN. Of course, when they ran tests from one LAN to another, no data were being passed. It was an easy problem to fix, but a lot of work was spent changing the customer's view on the problem and expectations of what needed to be done. The customer originally wanted to change vendors for the routing equipment and replace the SMDS service. Eventually, the customer was convinced that the equipment and service were fine and that the problem was internal to the organization.

Although SMDS is not widely available anymore, its behavior as a nonbroadcast multiple-access (NBMA) technology is similar to other currently available technologies such as ATM and frame relay.

---

An early part of every project is determining what your customer's expectations are about the project and adjusting these expectations accordingly. The idea here is not to give customers false expectations or to let them have unrealistic expectations; this will only lead to difficulties later in the project. Instead, the goal is to provide an accurate and realistic view of the technical problems in the network and what it will take to solve them. Customers' expectations will likely focus on budget, schedule, and personnel but may also include their opinions about technologies and methodologies. And, at times, politics become imbedded within the project and must be dealt with. The key here is to separate technical from nontechnical issues and focus on technical and resource issues.

Part of determining customers' expectations is understanding what customers want to do with their network. This may involve understanding the customer's business model and operations. In addition, the customer may expect to have significant input into the development of the network. As you set the customer's expectations, you may need to establish the lines of communication between the network architecture/design group, management, users, and network staff.

Having established what the customer's expectations are, you may need to adjust and manage these expectations. This can be done by identifying trade-offs and options for resources, technologies, architectures, and designs and then presenting and discussing trade-offs and options with your customer. Bringing customers into the process and working with them to make critical decisions about the network will help them become comfortable with the process.

If an existing network is part of this project, monitoring this network, as well as other parts of the system and its environment, can provide valuable information about the current behavior of users, applications, and devices and their requirements for the new network. Monitoring can also validate your and your customer's definitions of the problems with the existing network. When it is possible to monitor the network, you will need to determine what data you want to collect (based on what you want to accomplish with the data), any strategic places in the network where you want to collect this data, and the frequency and duration of data collection.

At this point in the process you should have several sets of information with which you can begin your network analysis. You may have historical data from network management; data captured during monitoring; requirements gathered from users, staff, and management; the customer's definition of the problem; and your definition. All of these data are used in the network analysis, of which there are three parts: requirements or needs analysis, flow analysis, and a risk (security) analysis. Information in these analyses can be placed on the customer's internal Web page, as mentioned earlier, although some information (e.g., the results of the risk analysis) may need to be kept private.

Results of the network analysis are used in the architecture and design processes, where sets of options are developed, including potential architectures, designs, topologies, technologies, hardware, software, protocols, and services.

These sets of options are then evaluated to determine the optimal solutions for the problems. Criteria will need to be developed throughout the analysis, architecture, and design processes in order to evaluate these options. Along with these criteria, you will use the results of the network analysis, including requirements, trade-offs, and dependencies between options.

Having selected one or more options, you will able to complete the network architecture and design and prepare for implementation. At this point you may consider developing a project plan to determine schedule, budget, and resources, as well as major and minor milestones, checkpoints, and reviews.

## 1.5 A Systems Methodology

We begin the network analysis process with a discussion about the systems methodology approach to networking. Applying a systems methodology to network analysis, architecture, and design is a relatively new approach, particularly in the Internet Protocol (IP) world. *Systems methodology* (as applied to networking) is viewing the network that you are architecting and designing, along with a subset of its environment (everything that the network interacts with or impacts), as a system. Associated with this system are sets of services (levels of performance and function) that are offered by the network to the rest of the system. This approach considers the network as part of the larger system, looking at interactions and dependencies between the network and its users, applications, and devices. As you will see, the systems methodology reveals interesting concepts that will be used throughout this book.

One of the fundamental concepts of the systems methodology is that network architectures and designs take into account the services that each network will provide and support. This reflects the growing sophistication of networks, which have evolved from providing basic connectivity and packet-forwarding performance to being a platform for various services. As discussed earlier, we are currently at the stage in network evolution where services are important to the success of many networks (third-generation networks). Some examples of third-generation networks are service-provider networks that support multiple levels of performance and pricing to its customers, content-distribution networks that specialize in high performance transport, and enterprise networks that incorporate and apply billing and usage models to their customers.

When a network is viewed as part of a system that provides services, the systems methodology works quite well for a variety of networks, from small and simple to large, complex networks. It helps in determining, defining, and describing the important characteristics and capabilities of your network.

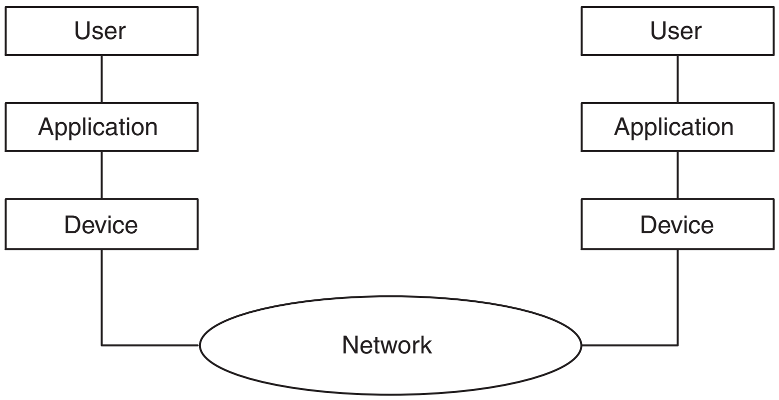

**FIGURE 1.11** Generic components of a system.

## 1.6 System Description

A *system* is a set of components that work together to support or provide connectivity, communications, and services to users of the system. Generically speaking, components of the system include users, applications, devices, and networks. Although users of the system can be considered outside of the system, they also have requirements that include them as part of the system. Throughout this book we will include users as part of the system. Figure 1.11 shows how these components are connected within the system.

Figure 1.11 shows the generic components of a system. These components can be subdivided, if necessary, to focus on a particular part of the system. For example, users in a corporate network could be further described as network and computer support personnel, as well as developers and customers of that corporation's product. In a similar sense, applications may be specific to a particular user, customer or group, generic to a customer base, or generic across the entire network.

If we were to compare this view of a system with the open system interconnect (OSI) protocol model, it would look like Figure 1.12. Note that in this comparison, some of the OSI layers are modified. This is to show that there may be multiple protocol layers operating at one of the system levels. For example, the OSI physical, data link, and network layers may be present at the device level and may also be present multiple times at the network level (e.g., at switches and routers throughout the network).

Figure 1.13 shows that devices can be subdivided by class to show specialized functions, such as storage, compute, or application servers, or an individual device may be subdivided to show its operating system (OS), device drivers, peripheral hardware, or application programming interface (API).

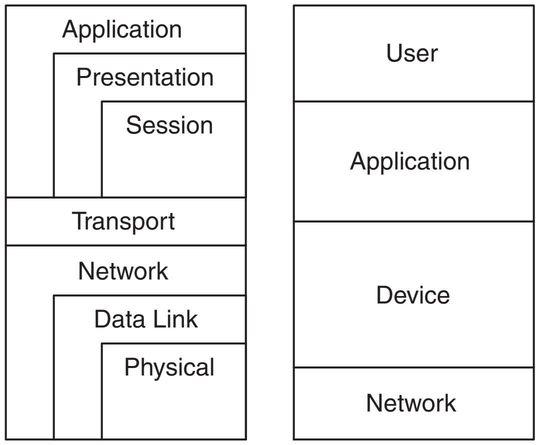

**FIGURE 1.12** Comparison of OSI layers to system levels.

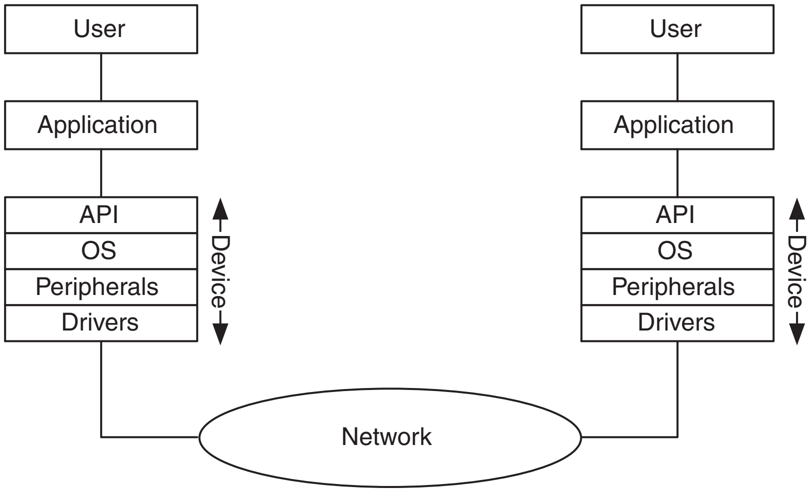

**FIGURE 1.13** Device component separated into constituents.

All of these components work together to provide connectivity and communications across the network, between users, applications, and devices. The connectivity and communications can be tailored to meet the specific needs of users and applications, such as real-time delivery of voice or streaming media, best-effort delivery of noninteractive data, or reliable delivery of mission-critical data.

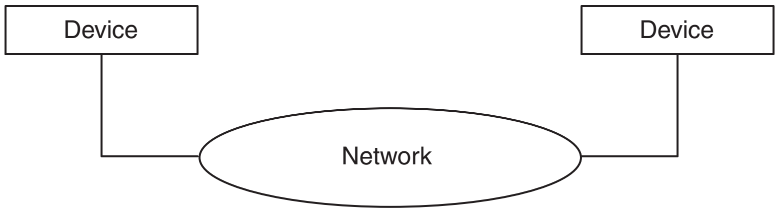

**FIGURE 1.14** Traditional view of system.

The degree of granularity used to describe system components is a trade-off between the amount of detail and accuracy you want in the description and how much time and effort you are willing to put into it. If you are the network architect responsible for a corporate network, you may have the ability to invest time and resources into developing a detailed description of your system's components, whereas a consultant or vendor's design engineer may have little time and resources to spend on such a description. It is important to note, however, that even a small amount of time invested here will pay dividends later in the analysis process.

The traditional view of a system focused on the network providing connectivity between devices (Figure 1.14) and typically did not consider the users or applications.

This traditional view of a system is not complete enough for today's networks. In particular, we need to include users and their applications in the system description. Experience shows that the degree of descriptiveness in the set (users, applications, devices, and networks) is usually sufficient to provide a complete and accurate description of most general-access systems, yet not so large as to be overwhelming to the network architect. (*General-access* is a term to describe common access of users to applications, computing, and storage resources across a network.) Within this set, users represent the end users, or customers, of the system. These end users are the final recipients of the network services supported by the system.

One reason for identifying components of the system is to understand how these components interface with each other across component boundaries. By defining what the components of the system are, we are also setting what is to be expected across each interface. For example, using the standard set (users, applications, devices, and networks), Figure 1.15 shows potential interfaces.

Although the description of the system given in Figure 1.15 is usually satisfactory for the start of most network architectures, there will be times when you will want to describe more components or have more defined interfaces. For example, the device-network interface may be simple or complex, depending on what you are trying to accomplish. For a network that will be providing simple connectivity, the network-

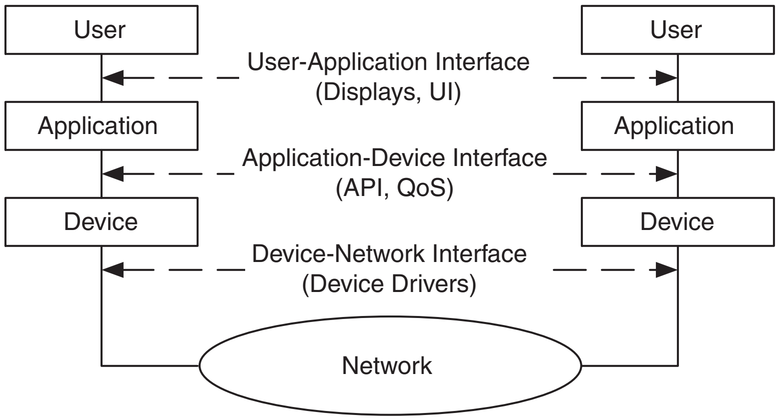

**FIGURE 1.15** Generic system with interfaces added.

device interface may be a standard LAN interface (e.g., 100BaseT Ethernet) without any prioritization or virtual LAN (VLAN 802.1p/q) tagging. For a network that provides more than simple connectivity, such as quality of service, the device-network interface may be more closely coupled with the device or application. This may be accomplished by using drivers that bypass portions of the protocol stack and APIs that can interpret application performance requirements.

To illustrate, consider the difference in network-device interfaces between a network using ATM, a link-layer technology, in the backbone versus a network where devices directly connect to ATM, sometimes referred to as a native ATM network. Figure 1.16 shows a system description where ATM is used in the backbone and Ethernet is used to connect to the devices. In this description, ATM does not directly interface with the devices, applications, or users and is isolated from them by IP routers. This is one of the scenarios described in the classical model of IP over ATM, described in RFC 2225 (RFC is a request for comment, a standards document of the Internet Engineering Task Force [IETF]) and used in more detail in the design section of this book. In this system description, ATM is embedded within the network component of the set (users, applications, devices, and networks). Why is this important? By isolating ATM within the network component of the system, any capabilities specific to ATM that may be directly useful to devices, applications, or users of the system are lost.

Figure 1.17, on the other hand, shows a native ATM network in which ATM is integrated into devices and will interface directly with applications. In describing the system this way, we are explicitly including capabilities, such as quality of service,

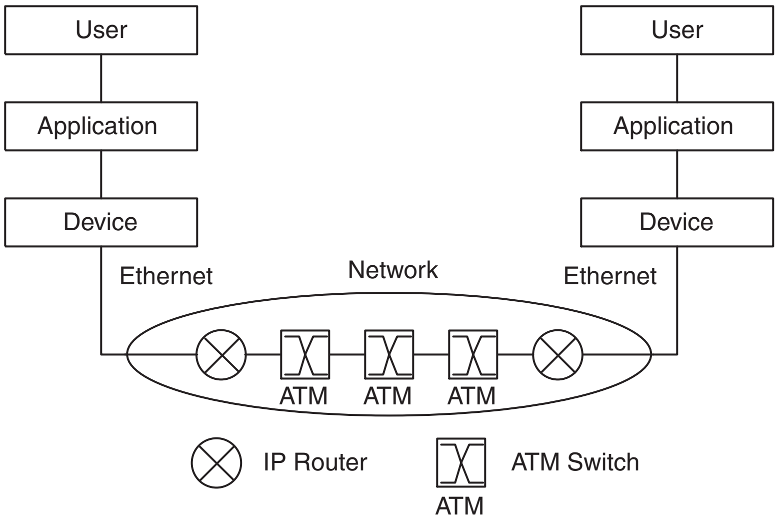

**FIGURE 1.16** Example of system with ATM in network.

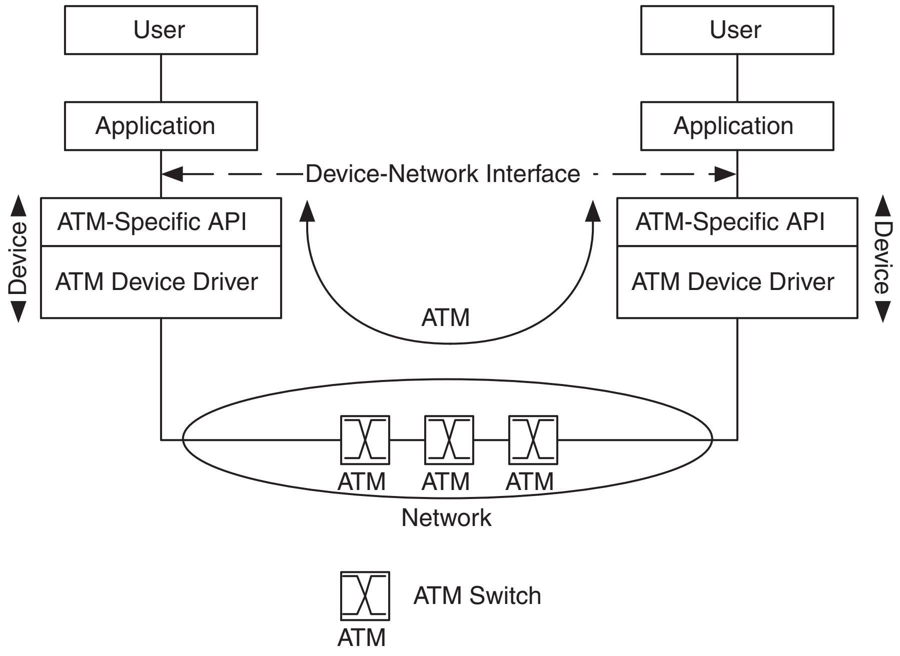

**FIGURE 1.17** Example of system with native ATM.

across the network-device interface. In this example the system may be better described as the set (users, applications, API, device driver, and network).

You should consider more detailed system descriptions when there is a need to identify system components that are likely to affect the delivery of services to users. In Figure 1.17, the ATM-specific API translates application requirements to the ATM network. As part of the system description, this ATM-specific API is clearly recognized as being important to the delivery of services.

Although the system description is an attempt to identify components across the entire system, we need to recognize that most systems are not completely homogeneous and that components may change in various parts of the system. This usually occurs in parts of the system that perform specific functions, such as a device-specific network (e.g., a video distribution network) or a storage-area network (SAN). For example, although an SAN may be described as the set (users, applications, devices, and networks), users may be other devices in the system, and the only application may be for system storage and archival.

## 1.7 Service Description

The concept of network services in this book builds upon the services work from the IETF. This organization has been developing service descriptions for IP networks. In general, they see network services as sets of network capabilities that can be configured and managed within the network and between networks. We apply this concept to network analysis, architecture, and design, integrating services throughout the entire system. This will help you take advantage of the services concept by analyzing, architecting, and designing based on services, and it will also prepare you for the near future, when services will be configurable and manageable within the network.

*Network services,* or services, are defined here as levels of performance and function in the network. We can look at this from two perspectives: as services being offered by the network to the rest of the system (the devices, applications, and users) or as sets of requirements from the network that are expected by the users, applications, or devices. Levels of performance are described by the performance characteristics capacity, delay, and RMA (reliability, maintainability, and availability), whereas functions are described as security, accounting, billing, scheduling, and management (and others). This will be described in more detail in the next section.

It is important to note that the concept of services in this book is based on what networks can deliver to the system. Thus, it is not to be confused with services that other parts of the system (e.g., applications) can deliver to each other (e.g., graphics

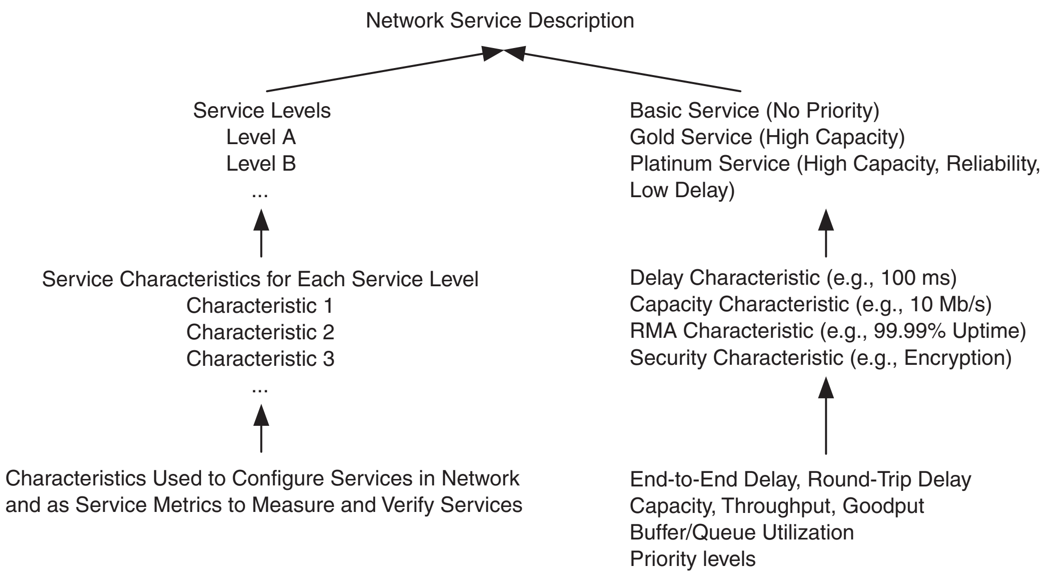
Network Service Description
Service Levels
Level A
Level B
...
Basic Service (No Priority)
Gold Service (High Capacity)
Platinum Service (High Capacity, Reliability, Low Delay)
Service Characteristics for Each Service Level
Characteristic 1
Characteristic 2
Characteristic 3
...
Delay Characteristic (e.g., 100 ms)
Capacity Characteristic (e.g., 10 Mb/s)
RMA Characteristic (e.g., 99.99% Uptime)
Security Characteristic (e.g., Encryption)
Characteristics Used to Configure Services in Network and as Service Metrics to Measure and Verify Services
End-to-End Delay, Round-Trip Delay
Capacity, Throughput, Goodput
Buffer/Queue Utilization
Priority levels

**FIGURE 1.18** Grouping characteristics into service levels and descriptions.

rendering services). When the term *service* is used in this book, it is in reference to network service.

Network services in most of today's networks are based on best-effort (unpredictable and unreliable) delivery. In addition to best-effort delivery, we will examine some new types of services, including high-performance, predictable (stochastic or probabilistic), and guaranteed services. These new services will require some different ways of looking at networks, and you will see how to incorporate such services into your architecture and design. We will also examine single-tier and multiple-tier performance in the network, how to distinguish between them, and how they relate to best-effort, predictable, and guaranteed services.

Network services are hierarchical, and individual service characteristics can be grouped together to form higher-level descriptions of a service, as shown in Figure 1.18.

## 1.8 Service Characteristics

One of the goals of network analysis is to be able to characterize services so that they can be designed into the network and purchased from vendors and service providers (e.g., via requests for information [RFI], quote [RFQ], or proposal [RFP], documents used in the procurement process). In addition to today's best-effort delivery, we will also examine new types of services (predictable and guaranteed), as well as single/multiple-tier performance.

*Service characteristics* are individual network performance and functional parameters that are used to describe services. These services are offered by the network to the system (the *service offering*) or are requested from the network by users, applications, or devices (the *service request*). Characteristics of services that are requested from the network can also be considered requirements for that network.

Examples of service characteristics range from estimates of capacity requirements based on anecdotal or qualitative information about the network to elaborate listings of various capacity, delay, and RMA requirements, per user, application, and/or device, along with requirements for security, manageability, usability, flexibility, and others.

---

**Example 1.3.** Examples of service characteristics are:

- Defining a security or privacy level for a group of users or an organization
- Providing 1.5 Mb/s peak capacity to a remote user
- Guaranteeing a maximum round-trip delay of 100 ms to servers in a server farm

---

Such requirements are useful in determining the need of the system for services, in providing input to the network architecture and design, and in configuring services in network devices (e.g., routers, switches, device operating systems). Measurements of these characteristics in the network to monitor, verify, and manage services are called *service metrics.* In this book we focus on developing service requirements for the network and using those characteristics to configure, monitor, and verify services within the network.

For services to be useful and effective, they must be described and provisioned end-to-end at all network components between well-defined demarcation points. Note that this means that end-to-end is not necessarily from one user's device to another user's device. It may be defined between networks, from users to servers, or between specialized devices (Figure 1.19). When services are not provisioned end-to-end, some components may not be capable of supporting them, and thus the services will fail. The demarcation points determine where end-to-end is in the network. Determining these demarcation points is an important part of describing a service.

Services also need to be configurable, measurable, and verifiable within the system. This is necessary to ensure that end users, applications, and devices are getting the services they have requested (and possibly paying for), and this leads to accounting and

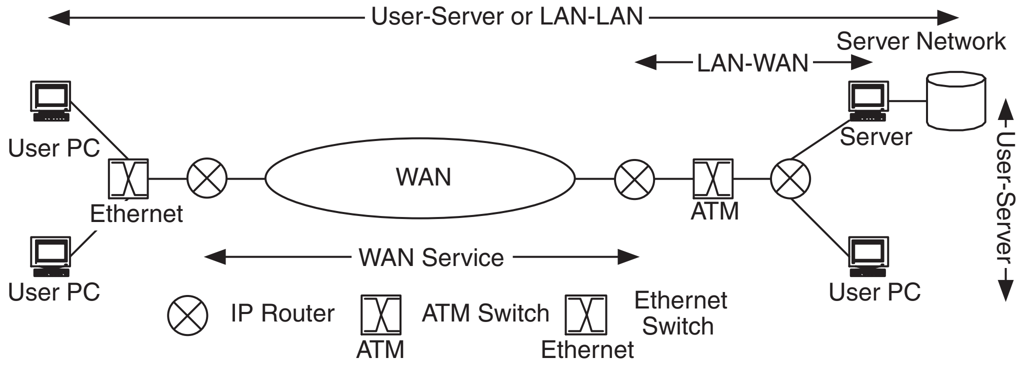

**FIGURE 1.19** Various demarcation points for end-to-end in a network.

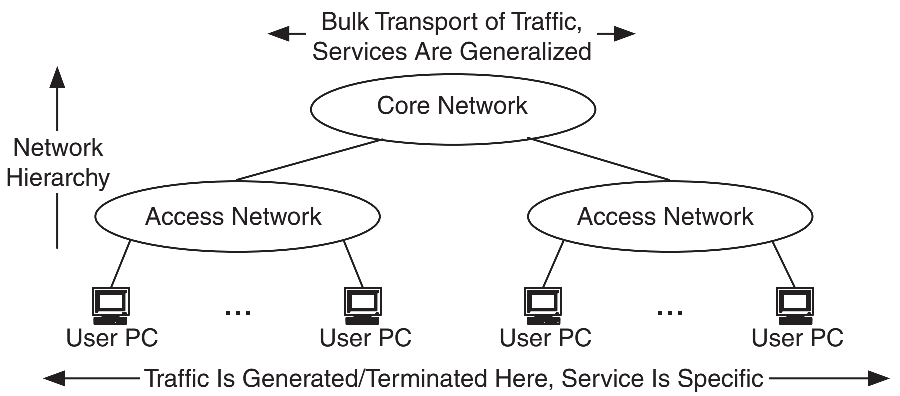
Bulk Transport of Traffic, Services Are Generalized
Core Network
Network Hierarchy
Access Network
Access Network
User PC
...
User PC
User PC
...
User PC
Traffic Is Generated/Terminated Here, Service Is Specific

**FIGURE 1.20** Example of service hierarchy within a network.

billing for system, including network, resources. You will see how service metrics can be used to measure and verify services and their characteristics.

Services are also likely to be hierarchical within the system, with different service types and mechanisms applied at each layer in the hierarchy. For example, Figure 1.20 shows a service hierarchy that focuses on bulk traffic transport in the core of the network while placing specific services at the access network close to the end users, applications, and devices.

## 1.8.1 Service Levels

Service characteristics can be grouped together to form one or more *service levels* for the network. This is done to make service provisioning easier in that you can config-

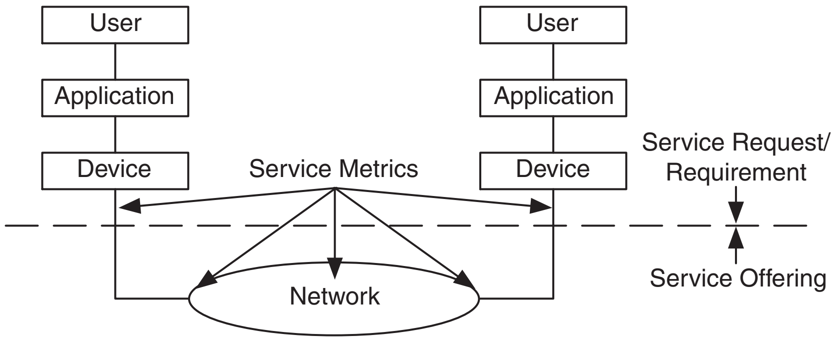

**FIGURE 1.21** Service requests, offerings, and metrics.

ure, manage, account, and bill for a group of service characteristics (service level) instead of a number of individual characteristics. For example, a service level (e.g., premium) may combine capacity (e.g., 1.5 Mb/s) and reliability (as 99.5% uptime). Service levels are also helpful in billing and accounting. This is a service-provider view of the network, where services are offered to customers (users) for a fee. This view of networking is becoming more popular in enterprise networks, displacing the view of networks as purely infrastructure of cost centers.

There are many ways to describe service levels, including frame relay committed information rates (CIRs), which are levels of capacity; ATM classes of service (CoSs), which combine delay and capacity characteristics; and IP types of service (ToSs) and qualities of service (QoSs), which prioritize traffic for traffic conditioning functions, which are described in the performance architecture (see Chapter 8). There can also be combinations of the aforementioned mechanisms, as well as custom service levels, based on groups of individual service characteristics. These combinations depend on which network technology, protocol, mechanism, or combination is providing the service.

In Figure 1.21 service offerings, requests, and metrics are shown applied to the system. In this example, demarcation of services is shown between the device and network components. Depending on the service requirement or characteristic, however, demarcation may also be between the device and application components (as in Figure 1.17).

---

**Example 1.4.** A request from the customer was that each building should have Fast Ethernet (FE) capacity to the rest of the network. As part of the requirements analysis, this request became a requirement for 100 Mb/s peak capacity from the users in each building.

This *service request* was then matched in the requirements and design processes by a technology choice that meets or exceeds the request; in this case, FE was chosen as the technology, and the service offering is 100 Mb/s to each building. *Service metrics* were then added, consisting of measuring the FE connections from the IP switch or router at each building to the backbone.

---

Services and service levels can be distinguished by their degrees of predictability or determinism. In the next section we will discuss best-effort delivery, which is not predictable or deterministic, as well as predictable and guaranteed services.

Services and service levels are also distinguished by their degrees of performance. You will see how the service performance characteristics capacity, delay, and RMA are used to describe services and service levels.

### 1.8.2 System Components and Network Services

Network services are derived from requirements at each of the components in the system. They are end-to-end (between end points that you define) within the system, describing what is expected at each component. Service requirements for the network we are building are derived from each component: There can be user requirements, application requirements, device requirements, and (existing) network requirements. Since the network component is what we are building, any requirements from the network component come from existing networks that the new network will incorporate or connect to.

Component requirements add to each other, refining and expanding requirements as we move closer to the network. User requirements, which are the most subjective and general, are refined and expanded by requirements from the application component, which are then refined and expanded by the device and network components. Thus, requirements filter down from user to application to device to network, resulting in a set of specific requirements that can be configured and managed in the network devices themselves. This results in a service offering that is end-to-end, consisting of service characteristics that are configured in each network device in the path (e.g., routers, switches, hubs). As in Figure 1.22, service characteristics are configured and managed within each element and at interfaces between elements. These services are the most specific of all and have the smallest scope (typically a single network device).

Defining network services and service metrics helps keep the system functioning and can provide extra value or convenience to users and their applications. By defin-

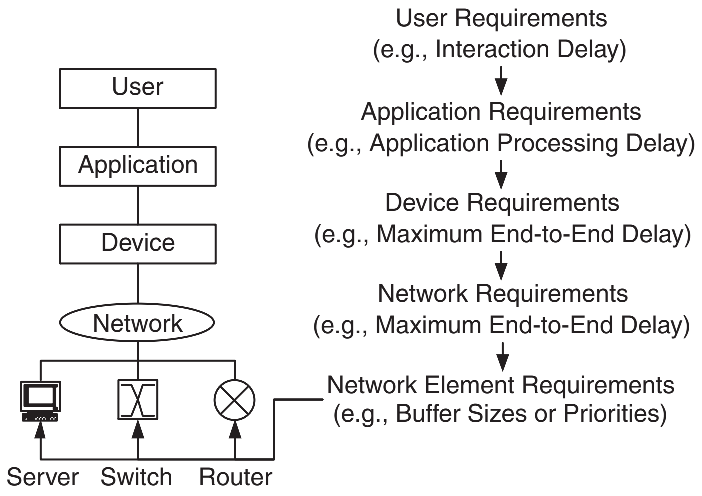
User
Application
Device
Network
Server
Switch
Router
User Requirements (e.g., Interaction Delay)
Application Requirements (e.g., Application Processing Delay)
Device Requirements (e.g., Maximum End-to-End Delay)
Network Requirements (e.g., Maximum End-to-End Delay)
Network Element Requirements (e.g., Buffer Sizes or Priorities)

**FIGURE 1.22** Requirements flow down components to network.

ing service metrics, we are determining what we will be measuring in the network, which will help us in network monitoring and management.

Recall that network services are sets of performance and function, so requirements may also include functions of one of the components. Examples of functions include network monitoring and management, security, and accounting. Services such as these need to be considered an integral part of the network architecture and design. In this book, security (and privacy) and network management each have their own architectures. This may seem obvious, but traditionally, services such as security and network management have been afterthoughts in architecture and design, often completely forgotten in the architecture, until problems arise.

---

**Example 1.5.** The network path shown in Figure 1.23 was designed to optimize performance between users and their servers. The graph at the bottom of the figure is an estimate of the expected aggregate capacity at each segment of the path. In this network, a packet over SONET (POS) link at the OC-48 level (2.544 Gb/s) connects two routers, which then connect to Gigabit Ethernet (GigE) switches.

After it was implemented, a security firewall was added at the users' LAN (with FE interfaces), without it being considered part of the original analysis, architecture, or design. The result was that the firewall changed the capacity characteristics across the path by reducing throughput between the user PCs and the GigE switch, as shown in Figure 1.24.

---

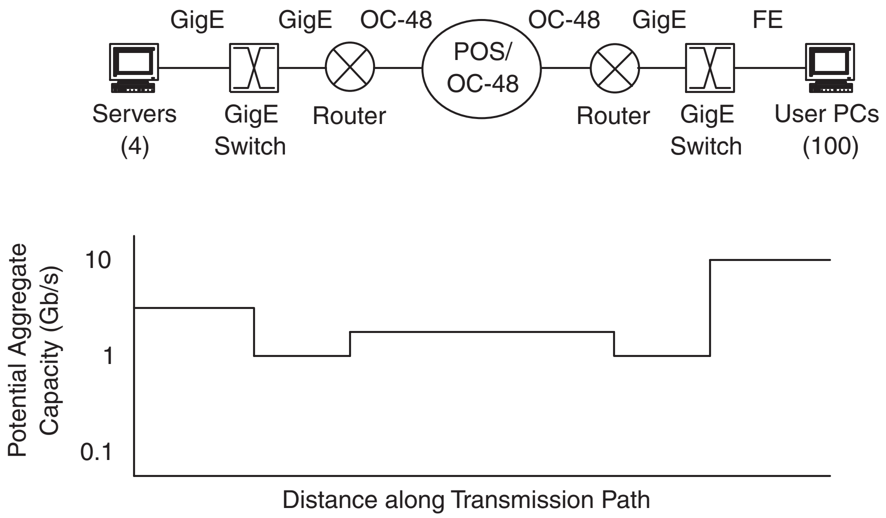

**FIGURE 1.23** Capacity at each point in transmission path before security firewall.

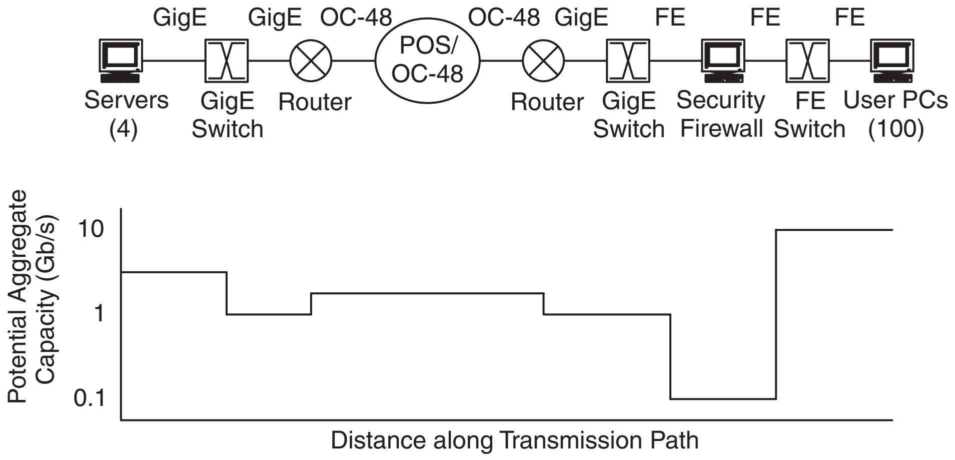

**FIGURE 1.24** Capacity at each point in transmission path after security firewall.

One of our architectural and design goals is to identify such performance bottlenecks before the network is implemented. By considering security, network management, services, and routing and addressing in the analysis process, we are much more likely to understand their behavior and effect on each other and the network. We will therefore be able to architect the network to accommodate their requirements and interoperability.

When service characteristics apply to individual network devices, such as routers, switches, data service units (DSUs), and so on, some of these characteristics may be vendor specific. In this book we will focus on those characteristics that are part of public standards and not vendor specific.

It is important to note that although standards-based characteristics are "standardized" on the basis of having their descriptions either publicly available (e.g., via an IETF RFC), sanctioned by an organization recognized by the networking community, or generally accepted and used as a *de facto standard,* the implementation of characteristics is open to interpretation and often varies across vendors and vendor platforms.

### 1.8.3 Service Requests and Requirements

Service requests and requirements are, in part, distinguished by the degree of predictability needed from the service by the user, application, or device making the request. Based on predictability, service requests are categorized as best effort, predictable, and guaranteed. Service requests and requirements can also be for single or multiple-tier performance for a network.

*Best-effort service* means that there is no control over how the network will satisfy the service request—that there are no guarantees associated with this service. Such requests indicate that the rest of the system (users, applications, and devices) will need to adapt to the state of the network at any given time. Thus, the expected service for such requests will be both unpredictable and unreliable, with variable performance across a range of values (from the network being unavailable to the lowest common denominator of performance across all of the technologies in the end-to-end path). Such service requests either have no specific performance requirements for the network or requirements that are based solely on estimates of capacity. When requirements are nonspecific, network performance cannot be tuned to satisfy any particular user, application, or device requirement.

*Guaranteed service* is the opposite of best-effort service. Where best-effort service is unpredictable and unreliable, guaranteed service must be predictable and reliable to such a degree that, when service is not available, the system is held accountable. A guaranteed service implies a contract between the user and provider. For periods when the contract is broken (e.g., when the service is not available), the provider must account for the loss of service and, possibly, appropriately compensate the user.

With best-effort and guaranteed services at opposite ends of the service spectrum, many services fall somewhere between. These are *predictable services,* which require some degree of predictability (more than best effort), yet do not require the accountability of a guaranteed service.

Predictable and guaranteed service requests are based on some a priori knowledge of and control over the state of the system. Such requests may require that the service operates either predictably or is bounded. Therefore, such services must have a clear set of requirements. For the network to provision resources to support a predictable or guaranteed service, the service requirements of that request must be configurable, measurable, and verifiable. This is where service requests, offerings, and metrics are applied.

Note that there are times when a service could be either best effort, predictable, or guaranteed, depending on how it is interpreted. Therefore, it is important to understand the need for a good set of requirements because these will help determine the types of services to plan for. Also, although the term *predictable* lies in a gray area between best effort and guaranteed, it is the type of service that is most likely to be served by most performance mechanisms, as we will see in Chapter 8.

For example, if a device requests capacity (bandwidth) of between 4 and 10 Mb/s, there needs to be a way to communicate this request across the network, a way to measure and/or derive the level of resources needed to support this request, a way to determine whether the required resources are available, and a method to control the information flow and network resources to keep this service between 4 and 10 Mb/s.

We can consider capacity to be a finite resource within a network. For example, the performance of a 100 Mb/s FE connection between two routers is bounded by that technology. If we were to look at the traffic flows across that 100 Mb/s connection, we would see that, for a common best-effort service, capacity would be distributed across all of the traffic flows. As more flows were added to that connection, the resources would be spread out until, at some point, congestion occurs. Congestion would disrupt the traffic flows across that connection, affecting the protocols and applications for each flow. What is key here is that, in terms of resource allocation, all traffic flows have some access to resources.

This is shown in Figure 1.25. In this figure, available capacity (dashed curve) decreases as the number of traffic flows increases. Correspondingly, the loading on the network (solid curve) from all of the traffic flows increases. However, at some point congestion affects that amount of user traffic that is being carried by the connection, and throughput of the connection (heavy curve) drops. As congestion interferes with the end-to-end transport of traffic, some protocols (e.g., TCP) will retransmit traffic. The difference between the loading and the throughput curves is due to retransmissions. This is undesirable, for while the connection is being loaded, only a percentage of that loading is successfully delivered to destinations. At some point, all of the traffic on that connection could be due to retransmissions and throughput approaches zero. This approach is used in best-effort networks.

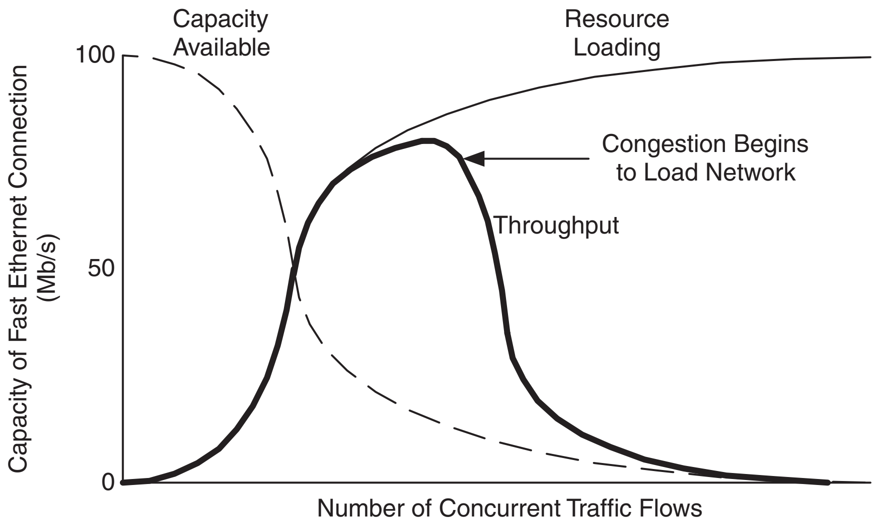

**FIGURE 1.25** Performance of Fast Ethernet connection under best-effort conditions.

In contrast, consider a telephony network. Calls are made on this network, and resources are allocated to each call. As more calls are added to the network, at the point where all of the resources have been allocated, additional calls are refused. The exiting calls on the network may suffer no performance degradation, but no new calls are allowed until resources are available. *Call admission control* (CAC) is a mechanism to limit the number of calls on a network, thereby controlling the allocation of resources.

This is shown in Figure 1.26. Individual calls are shown in this figure, and each call is 10 Mb/s for simplicity. As each call is accepted, resources are allocated to it, so the availability drops (dotted line) and loading increases (solid line) for each call. When the resources are exhausted, no further calls are permitted (dashed line). Congestion is not a problem for the existing calls, and throughput is maximized. This approach is similar to a guaranteed service.

There is a trade-off between these two approaches to resource allocation in a network. Although a best-effort network will allow access to as many traffic flows as possible, performance degradation across all of the traffic flows can occur. Admission control preserves resources for traffic flows that have already been allocated resources but will refuse additional traffic flows when resources are exhausted. In many networks, both approaches (or a hybrid between them) are desired. This will be discussed in detail in Chapter 8.

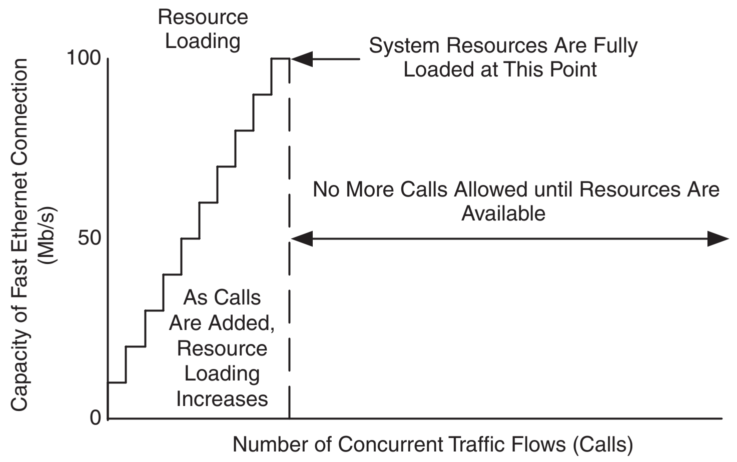

**FIGURE 1.26** Performance of Fast Ethernet connection under CAC.

Service requests and requirements can also be low or high performance in terms of capacity, delay, and RMA. Low- and high-performance requirements depend on each particular network. A requirement is low or high performance relative to other requirements for that network. *Low performance* is an indicator that the service request or requirement's performance characteristics are less than a performance threshold determined for that network. Likewise, *high performance* is an indicator that the service request or requirement's performance characteristics are greater than a performance threshold determined for that network. Thus, in determining low and high performance for a network, we will develop one or more performance thresholds for that network. Multiple-tier performance indicates that there are multiple tiers of performance for that network. Single-tier performance requirements are roughly equivalent within a network.

Note that low and high performance are not described in terms of best-effort, predictable, or guaranteed service because they are independent of each other. *Best-effort, predictable,* and *guaranteed service* refer to the degree of predictability of a request or requirement, whereas *low* and *high performance* refer to a relative performance level for that request or requirement. For example, a network can be entirely best effort (most current networks are), yet we can often distinguish low- and high-performance requirements for such a network. And when a network has low- and high-performance regions for capacity, delay, and RMA, there may be predictable or guaranteed requirements in either region.

By their nature, each service has its associated set of requirements. These requirements are based on the levels of performance and function desired by the user, application, or device requesting service. Performance requirements are described in terms of capacity, delay, and RMA, whereas functional requirements describe specific functions needed in the service, such as multicast, security, management, or accounting. We will use requests for performance and function in developing the network architecture and design, for example, in describing the overall level of performance needed in the network.

As mentioned earlier, service performance requirements (capacity, delay, and RMA) can be grouped together, forming one or more service levels. For example, a service request may couple a specific capacity (e.g., 1.5 Mb/s) with a bound on end-to-end delay (e.g., 40 ms). At times, such service levels can be mapped to well-known service mechanisms such as ATM CoS, frame relay CIR, SMDS CoS, or IP ToS or QoS. Thus, service levels are a way to map performance and functional requirements to a well-known or standard network service offering. A properly specified service will provide insight into which performance characteristics should be measured in the network to verify service delivery.

### 1.8.4 Service Offerings

Service requests that are generated by users, applications, or devices are supported by services that are offered by the network. These service offerings (e.g., via ATM CoS, frame relay CIR, SMDS CoS, or IP ToS or QoS mentioned in the previous section) are the network counterparts to user, application, and device requests for service.

Service offerings map to service requests and thus can also be categorized as best effort, predictable, and guaranteed. Best-effort service offerings are not predictable—they are based on the state of the network at any given time. There is little or no prior knowledge about available performance, and there is no control over the network at any time. Most networks today operate in best-effort mode. A good example of a network that offers best-effort service is the current Internet.

Best-effort service offerings are compatible with best-effort service requests. Neither the service offering nor request assume any knowledge about the state of or control over the network. The network offers whatever service is available at that time (typically just available bandwidth), and the rest of the system adapts the flow of information to the available service (e.g., via TCP flow control).

---

**Example 1.6.** An example of a best-effort service request and offering is when a file transfer (e.g., using FTP) occurs over the Internet. FTP uses TCP as its transport protocol, which, via a sliding-window flow-control mechanism, adapts to approximate the current state of the network it is operating across. Thus, the service requirement from FTP over TCP is best effort, and the corresponding service offering from the Internet is best effort. The result is that, when the FTP session is active, the performance characteristics of the network (Internet) and flow control (TCP windows) are constantly interacting and adapting, as well as contending with other application sessions for network resources. In addition, as part of TCP's service to the applications it supports, it provides error-free, reliable data transmission.

---

On the other hand, predictable and guaranteed service offerings have some degree of predictability or are bounded. To achieve this, there has to be some knowledge of the network along with control over the network in order to meet performance bounds or guarantees. Such services must be measurable and verifiable.

Just because a service is predictable or guaranteed does not necessarily imply that it is also high performance. For example, when using our definition of predictable service, the telephone network offers this type of service. Although at first look this may seem unlikely, consider the nature of telephone service. To support voice conversations, this network must be able to support fairly strict delay and delay variation tolerances, even though the capacity per user session (telephone call) is relatively small, or low performance. What is well known from a telephony perspective is somewhat new in the current world of data networking. Support for strict delay and delay variation is one of the more challenging aspects of data network architecture and design.

Predictable and guaranteed service offerings should be compatible with their corresponding service requests. In each case, service performance requirements (capacity, delay, and RMA) in a service request are translated into the corresponding performance characteristics in the service offering.

---

**Example 1.7.** An example of a predictable service request and offering can be seen in a network designed to support real-time streams of telemetry data. An architectural/design goal for a network supporting real-time telemetry is the ability to specify end-to-end delay and have the network satisfy this delay request. A real-time telemetry stream should have an end-to-end delay requirement, and this requirement would form the basis for the service request. For example, this service request may be for an end-to-end delay of 25 ms, with a delay variation of ±400 μs. This would form the request and the service level (i.e., a QoS level) that needs to be supported by the network. The network would then be architected and designed to support a QoS level of 25 ms end-to-end delay and delay variation of ±400 μs. Delay and delay variation would then be measured and verified with service metrics in the system, perhaps by using common utilities, such as ping (a common utility for measuring round-trip delay) or TCPdump (a utility for capturing TCP information), or by using a custom application.

---

We will use various methods to describe service performance requirements and characteristics within a network, including thresholds, bounds, and guarantees. We will also learn how to distinguish between high and low performance for each network project.

This approach does not mean that best-effort service is inherently low performance or that predictable or guaranteed services are high performance. What this approach does is signify that predictability in services is also an important characteristic and is separate from performance. There are times when a network is best architected for best-effort service, and other times when best-effort, predictable, and guaranteed services are needed. We will see that when predictable or guaranteed services are required in the network, consideration for those requirements will tend to drive the architecture and design in one direction while consideration for best-effort service will drive the architecture and design in another direction. It will be the combination of all services that will help make the architecture and design complete.

### 1.8.5 Service Metrics

For service performance requirements and characteristics to be useful, they must be configurable, measurable, and verifiable within the system. Therefore, we will describe performance requirements and characteristics in terms of *service metrics,* which are intended to be configurable and measurable.

Since service metrics are meant to be measurable quantities, they can be used to measure thresholds and limits of service. Thresholds and limits are used to distinguish whether performance is in conformance (adheres to) or nonconformance (exceeds) with a service requirement. A *threshold* is a value for a performance characteristic that is a boundary between two regions of conformance and, when crossed in one or both directions, will generate an action. A *limit* is a boundary between conforming and nonconforming regions and is taken as an upper or lower limit for a performance characteristic. Crossing a limit is more serious than crossing a threshold, and the resulting action is usually more serious (e.g., dropping of packets to bring performance back to conformance).

For example, a threshold can be defined to distinguish between low and high performance for a particular service. Both low- and high-performance levels are conforming to the service, and the threshold is used to indicate when the boundary is

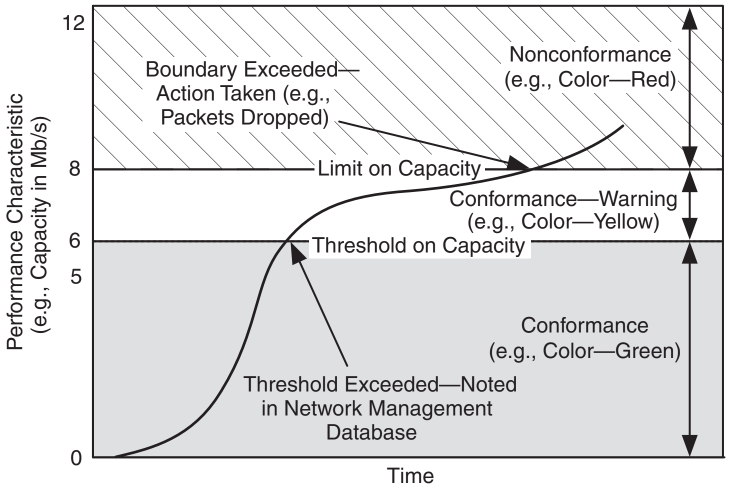

**FIGURE 1.27** Performance limits and thresholds.

crossed. This threshold can be measured and monitored in the network, triggering some action (e.g., a flashing red light on an administrator's console) when this threshold is crossed. An example of this might be in measuring the round-trip delay of a path. A threshold of *N* ms is applied to this measurement. If the round-trip times exceed *N* ms, an alert is generated at a network management station. We will discuss this in greater detail in the network management architecture.

In a similar fashion, limits can be created with service metrics to provide upper and lower boundaries on a measured quantity. When a limit is crossed, traffic is considered nonconforming (it exceeds the performance requirement) and action is taken to bring the traffic back into conformance (e.g., by delaying or dropping packets). Figure 1.27 shows how limits and thresholds may be applied in the system. In this figure, a threshold of 6 Mb/s is the boundary between low and high performance for a service requirement, and an upper limit of 8 Mb/s is the boundary between conformance and nonconformance for that service. When traffic crosses the 6 Mb/s threshold, a warning is sent to network management (with a color change from green to yellow). These notices can be used to do trend analysis on the network, for example, to determine when capacity needs to be upgraded. When traffic crosses the 8 Mb/s limit, the network takes action to reduce the capacity used by that traffic flow and an alert is sent to network management (with a color change from yellow to red) until the capacity level drops below 8 Mb/s and is again conforming.

Thresholds and limits are useful applications of service metrics to understand and control performance levels in the network, in support of services.

## 1.9 Performance Characteristics

Services may include one or more of the performance characteristics we have mentioned so far in this chapter: capacity, delay, and RMA. Each characteristic is actually a label for a class of characteristics of that type. For example, the term *capacity* is used as a label for the class of characteristics that involves moving information from place to place, including bandwidth, throughput, goodput, and so forth. Similarly, delay is a label for the class of characteristics that includes end-to-end delay, round-trip delay, and delay variation. RMA is a label for the class of characteristics that includes reliability, maintainability, and availability. Thus, when the terms *capacity, delay,* and *RMA* are used in this book, you can use other terms from each class, depending on your network.

There are times when it makes more sense to describe capacity in terms of throughput, for example, when developing requirements for applications. Round-trip delay is commonly used as a measure for delay, although at times delay requirements are expressed in terms of one-way delay.

### 1.9.1 Capacity

*Capacity* is a measure of the system's ability to transfer information (voice, data, video, or combinations of these). Several terms are associated with capacity, such as bandwidth, throughput, or goodput. Although we will use the generic term capacity throughout this book to reference this class of characteristics, you may choose to use another term in place of or along with capacity.

*Bandwidth* is the theoretical capacity of one or more network devices or communications links in the system. Theoretical, or raw, capacity does not take into account overhead from higher-layer protocols or the performance loss related to inefficiencies in the system (e.g., delays in the operating system or peripheral devices).

*Throughpu*t is the realizable capacity of the system or its network devices. Values for throughput vary, depending on system design, types and configurations of equipment, and where in the protocol stack the measurement for capacity is being taken.

---

**Example 1.8.** The bandwidth of a SONET OC-3c link is 155.52 Mb/s, which is three times the bandwidth of an OC-1 link (51.84 Mb/s). This bandwidth does not include data-link, network, or transport-layer protocol (e.g., ATM, IP, or transport control protocol/user datagram protocol [TCP/UDP]) overhead or, in the case of wide-area networks, the loss in performance due to the bandwidth × delay product in the network. When a network or element is performing at its theoretical capacity, it is said to be performing at *line rate.* When an OC-3c circuit was tested, values of realizable capacity (throughput) ranged from approximately 80 to

128 Mb/s (measurements taken at the transport [TCP] layer of the National Research and Education Network [NREN] and Numerical Aerodynamic Simulation [NAS] networks, NASA Ames Research Center, March 1996).

A discussion on protocol overhead is presented in Chapter 11.

---

Both bandwidth and throughput are important terms when describing system, network, application, and device capacities. Each gives a different and compatible view of capacity.

### 1.9.2 Delay

*Delay* is a measure of the time difference in the transmission of information across the system. In its most basic sense, delay is the time difference in transmitting a single unit of information (bit, byte, cell, frame, or packet) from source to destination. As with capacity, there are several ways to describe and measure delay. There are also various sources of delay, such as propagation, transmission, queuing, and processing. Delay may be measured in one direction (end-to-end) and both directions (round-trip). Both end-to-end and round-trip delay measurements are useful; however, only round-trip delays can be measured with the use of the practical and universally available utility *ping.*

Another measure of delay incorporates device and application processing, taking into account the time to complete a task. As the size of a task increases, the application processing times (and thus the response time of the system) also increase. This response time, termed here *latency,* may yield important information about the behavior of the application and the network. Latency can also be used to describe the response time of a network device, such as the latency through a switch or router. In this case the processing time is of that switch or router.

Delay variation, which is the change in delay over time, is an important characteristic for applications and traffic flows that require constant delay. For example, real-time and near-real-time applications often require strict delay variation. Delay variation is also known as *jitter.*

Together, delay (end-to-end and round-trip), latency, and delay variation help describe network behavior.

### 1.9.3 RMA

*RMA* refers to reliability, maintainability, and availability. *Reliability* is a statistical indicator of the frequency of failure of the network and its components and represents the unscheduled outages of service. It is important to keep in mind that only failures that prevent the system from performing its mission, or mission-critical failures (more on this in Chapter 2), are generally considered in this analysis. Failures of components that have no effect on the mission, at least when they fail, are not considered in these calculations. Failure of a standby component needs tending to but is not a mission-critical failure.

Reliability also requires some degree of predictable behavior. For a service to be considered reliable, the delivery of information must occur within well-known time boundaries. When delivery times vary greatly, users lose confidence in the timely delivery of information. In this sense the term *reliability* can be coupled with *confidence* in that it describes how users have confidence that the network and system will meet their requirements.

A parallel can be seen with the airline industry. Passengers (users) of the airline system expect accurate delivery of information (in this case the passengers themselves) to the destination. Losing or misplacing passengers is unacceptable. In addition, predictable delivery is also expected. Passengers expect flights to depart and arrive within reasonable time boundaries. When these boundaries are crossed, passengers are likely to use a different airline or not fly at all. Similarly, when an application is being used, the user expects a reasonable response time from the application, which is dependent on the timely delivery of information across the system.

Along with reliability is maintainability. *Maintainability* is a statistical measure of the time to restore the system to fully operational status after it has experienced a fault. This is generally expressed as a mean-time-to-repair (MTTR). Repairing a system failure consists of several stages: detection; isolation of the failure to a component that can be replaced; the time required to deliver the necessary parts to the location of the failed component (logistics time); and the time to actually replace the component, test it, and restore full service. MTTR usually assumes the logistics time is zero; this is an assumption, which is invalid if a component must be replaced to restore service but takes days to obtain.

To fully describe this performance class, we add availability to reliability and maintainability. *Availability* (also known as operational availability) is the relationship between the frequency of mission-critical failures and the time to restore service. This is defined as the mean time between mission-critical failures (or mean time between failures) divided by the sum of mean time to repair and mean time between mission-critical failures or mean time between failures. These relationships are shown in the following equation, where *A* is availability.

$$\mathrm{A} = (\mathrm{MTBCF})/(\mathrm{MTBCF} + \mathrm{MTTR}) \text{ or } \mathrm{A} = (\mathrm{MTBF})/(\mathrm{MTBF} + \mathrm{MTTR})$$

Capacity, delay, and RMA are dependent on each other. For example, the emphasis of a network design may be to bound delay: A system supporting point-of-sale

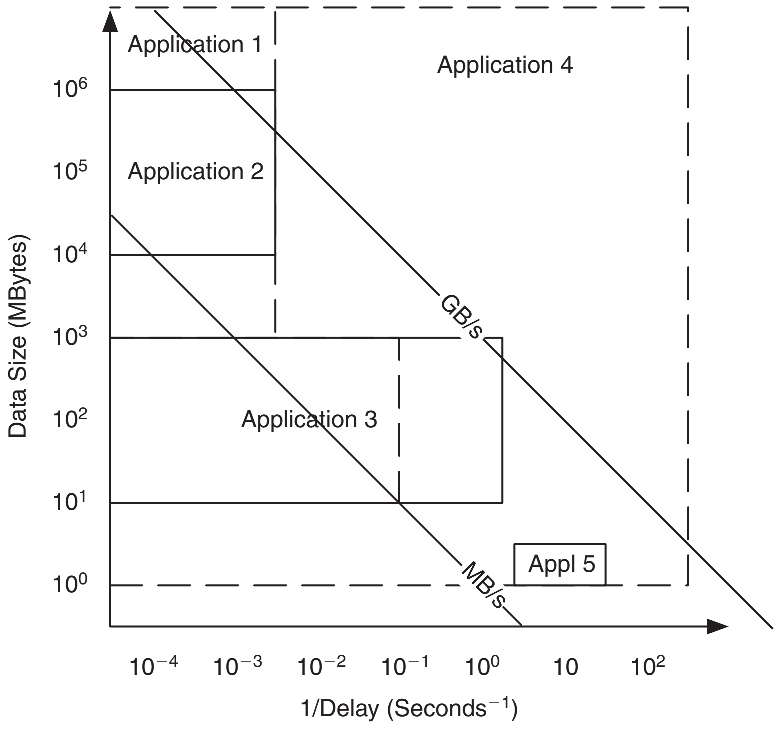

**FIGURE 1.28** Example of a 2D performance envelope.

transactions may need to guarantee delivery of customer information and completion of the transaction within 15 seconds (where the network delay is on the order of 100s of ms); a Web application can have similar requirements. However, in a compute-intensive application we may be able to optimize the system by buffering data during periods of computing. In this case, delay may not be as important as a guarantee of eventual delivery. On the other hand, a system supporting visualization of real-time banking transactions may require a round-trip delay of less than 40 ms, with a delay variation of less than 500 μs. If these delay boundaries are exceeded, the visualization task fails for that application, forcing the system to use other techniques.

## 1.9.4 Performance Envelopes

Performance requirements can be combined to describe a performance range for the system. A *performance envelope* is a combination of two or more performance require-

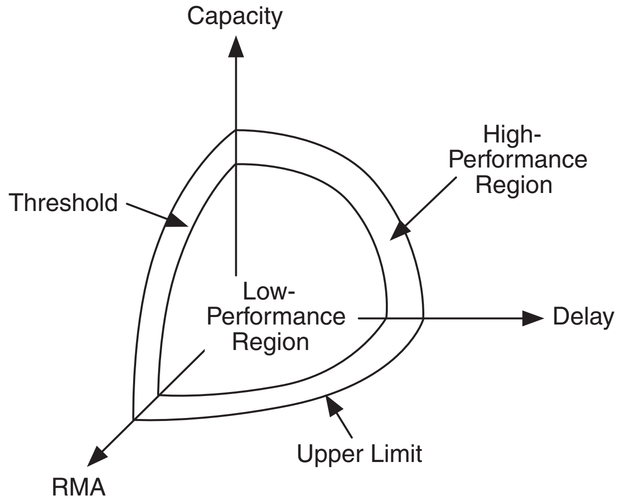

**FIGURE 1.29** Example of a 3D performance envelope.

ments, with thresholds and upper and/or lower limits for each. Within this envelope, levels of application, device, and/or network performance requirements are plotted. Figures 1.28 and 1.29 show two such envelopes. The performance envelope in Figure 1.28 consists of capacity, in terms of data sizes transferred across the network, and end-to-end delay. In this figure, delay is shown as 1/delay for consistency.

Figure 1.29 is a 3D performance envelope, showing capacity, delay, and RMA. This envelope also describes two regions of performance, low and high performance, which are functions of the limits and thresholds for capacity, delay, and RMA.

Performance envelopes such as these are useful for visualizing the regions of delay, capacity, and RMA in which the network will be expected to operate based on requirements developed for that network. In Chapter 2 we will discuss how requirements are developed for a network.

## 1.10 Network Supportability

The ability of the customer to sustain the required level of performance (that architected and designed into the network) over the entire life cycle of the network is an area of networking that is often neglected. It is a mistake to assume that a successful network architecture and design meets the requirements only on the day that it is delivered to the customer and that future requirements are the responsibility of the customer.

Experience indicates that 80% of the life cycle costs of a system are the operations and support costs, and only 20% is the cost incurred to develop, acquire, and install it. Good network architects/designers will take into account the major factors that affect operability and supportability as they make their decisions. Knowledgeable customers will insist that they understand the operations and support implications of a network architecture and design. At times, such issues may be of more concern than the feasibility of a new technology.

The postimplementation phases of a network's life cycle can be broken into three elements: operations, maintenance, and human knowledge. The operations element focuses on ensuring that the network and system are properly operated and managed and that any required maintenance actions are identified. The maintenance element focuses on preventive and corrective maintenance and the parts, tools, plans, and procedures for accomplishing these functions. The human knowledge element is the set of documentation, training, and skilled personnel required to operate and maintain the network and system. Design decisions affect each of these factors and have a direct impact on the ability of the customer to sustain the high level of service originally realized upon implementation of the network.

Failure to consider supportability in the analysis, architecture, and design processes has a number of serious consequences. First, a smart customer, when faced with a network architecture/design that obviously cannot be operated or maintained by his or her organization, will reject the network project or refuse to pay for it. Second, a customer who accepts the architecture/design and subsequent implementation will have inadequate resources to respond to network and system outages, experience unacceptable performance after a period of time, and may suffer adverse effects in his or her operation or business (e.g., a loss of their customers or revenue). Other customers will be highly dissatisfied with their networks and either require the architect/designer to return and repair it by providing adequate materials to sustain its required performance level or will prematurely replace it. None of these cases reflects positively on the network architect/designer or implementation team and often lead to finger pointing that can be more painful than any acceptance test.

Key characteristics of a network architecture and design that affect the postimplementation costs include:

- Network and system reliability
- Network and system maintainability
- Training of the operators to stay within operational constraints
- Quality of the staff required to perform maintenance actions

Some examples of key network architecture/design decisions that affect these characteristics include:

- Degree of redundancy of critical-path components in network architecture/design
- Quality of network components selected for installation
- Location and accessibility of components requiring frequent maintenance
- Implementation of built-in test equipment and monitoring techniques

Supportability must be considered throughout the life cycle of the network. An accurate assessment of the requirements for continuous service at full performance level must be included in the requirements analysis process, along with a statement of specific, measurable requirements. During the architecture and design processes, trade-offs must take into account the impact of supportability and the concept of operations must be formulated. Last, during implementation, two major tasks must be accomplished to ensure supportability:

1. Conformance to the network architecture and design must be validated and nonconformance corrected or (at least) documented to ensure that performance is adequate and that maintenance can be performed.
2. Operations and maintenance personnel must understand and be trained in the technologies that are being deployed, including how to operate the network and system properly, when to perform maintenance, and how to most quickly restore service in the event of a fault.

A detailed discussion of how supportability fits into the overall architecture and design processes is provided in Chapter 2.

## 1.11 Conclusion

In this chapter you learned definitions of network analysis, architecture, and design; the importance of network analysis in understanding the system and providing a defensible architecture and design; and the model for the network analysis, architecture, and design processes.

You have also learned that networks are not independent entities but rather a part of the system and that the delivery of network services is a goal of the system. Network services consist of performance and function and are offered to users, applications, and devices so that they can accomplish their work on the system. In order to architect and design a network to support services, you need to know what they are, how they work together, and how to characterize them. Once you do this, you will have a broad view of what the network will need to support, which you can take to the next levels of detail as you proceed with the network analysis.

By describing the system as a set of components (e.g., user, application, device, network), you can apply interfaces between these components to help understand the relationships, inputs, and outputs between each of the components.

You have also learned about different types of services, from best-effort, unpredictable, and unreliable service to predictable, bounded, and somewhat predictable service, to guaranteed services with accountability.

To go to a level deeper in the discussion about services, we considered the service performance characteristics capacity, delay, and RMA (reliability, maintainability, and availability). These characteristics will be useful only if we can measure and verify their values in the system. We discussed these values, as well as service metrics, thresholds, and boundaries. We learned that performance characteristics could be combined into a performance envelope.

Having thought about systems, services, and their characteristics, we are now ready to quantify what we want from our networks. To do this, we first need to gather, analyze, and understand the requirements from the system. This is *requirements analysis,* the next step in the network analysis process.

## 1.12 Exercises

1. In Example 1.1, an analogy was drawn between a network's architecture and design and a home's architecture and design. Provide a similar analogy, using a computer's architecture and design.
2. Hierarchy and interconnectivity are a fundamental trade-off in networks. Given the network hierarchy shown in Figure 1.30, with costs assigned to each link, show how interconnectivity would improve the performance of traffic flowing between Joe's computer and Sandy's computer. Costs are shown as numbers but could represent the capacity of each link or the costs incurred by using each link. What is the total cost of traveling the hierarchy between Joe's computer and Sandy's? In this figure, where would you add a link of cost 15 so that the total cost between Joe's computer and Sandy's is less than it is when you travel the entire hierarchy?
3. In Figure 1.9, connections are added between networks in the Internet to provide a better performing path for select traffic flows. An example of this is a

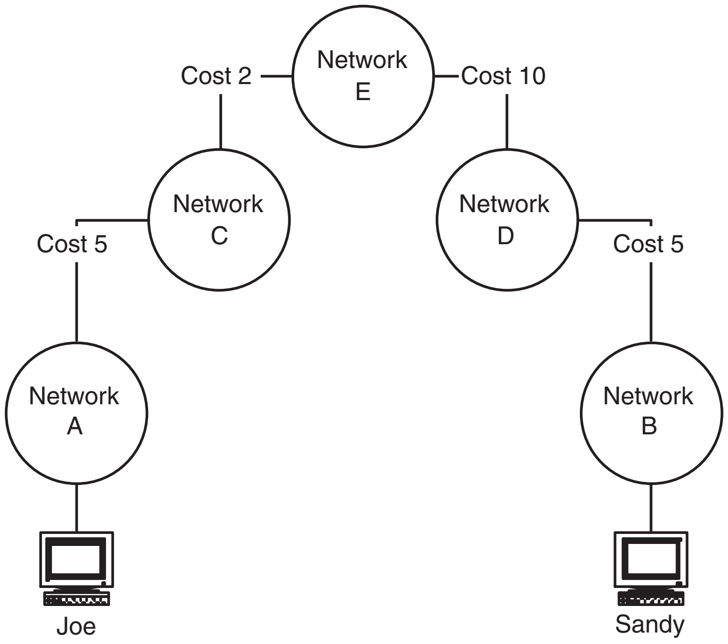

**FIGURE 1.30** Network hierarchy for Exercise 2.

content delivery network (CDN). What is a CDN? Show how a CDN uses interconnectivity to provide better performance characteristics to its users.

4. In defining where services can be applied in a network, end-to-end is determined by where you want a service to start and stop. For example, if your WAN is supplied by a service provider (e.g., an ATM or frame relay service), you may want to define the end points and characteristics of that service. If you use IP routers at each LAN-WAN interface to that service, describe the following: (1) at which network devices would you define the end points of the service, and (2) what characteristics (service metrics) would you use to measure the service?
5. Service requirements flow from user to application to device to network, becoming more specific along the way. If you were given an application requirement for end-to-end delay (e.g., 100 ms) between an application server on one network and users on another network, for example, how might that translate into delay in the network and devices? What types of service metrics could you use to measure it?
6. For Example 1.5, the delay characteristics for the segments (including the processing at the switches) are as follows: for each GigE segment, 100 μs; for the

PoS OC-48 segment between routers, 1 ms; for each FE segment, 200 µs; and for the security firewall, 5 ms. Draw graphs showing the end-to-end delay performance (in the direction from user PC to server) before and after the security firewall is added.

7. Which of the following applications require best-effort (unpredictable and unreliable), guaranteed (predictable and reliable, with accountability), or predictable service. Give reasons for your choices.
   - High-quality (phone company–grade) voice calls
   - Voice over IP (VoIP) calls
   - File transfers via FTP

- Audio file downloads
- A commercial video-on-demand service
- User access to servers in a corporation

8. Show how performance boundaries and thresholds could be used in the following scenarios.
    - An application has a service requirement for round-trip delay to be less than 100 ms. If delay is greater than 100 ms, notify the network administrator.
    - A user requires capacity of up to 512 Kb/s but may not exceed 1.5 Mb/s. You want to keep track of how much time the user's capacity is between 512 Kb/s and 1.5 Mb/s.