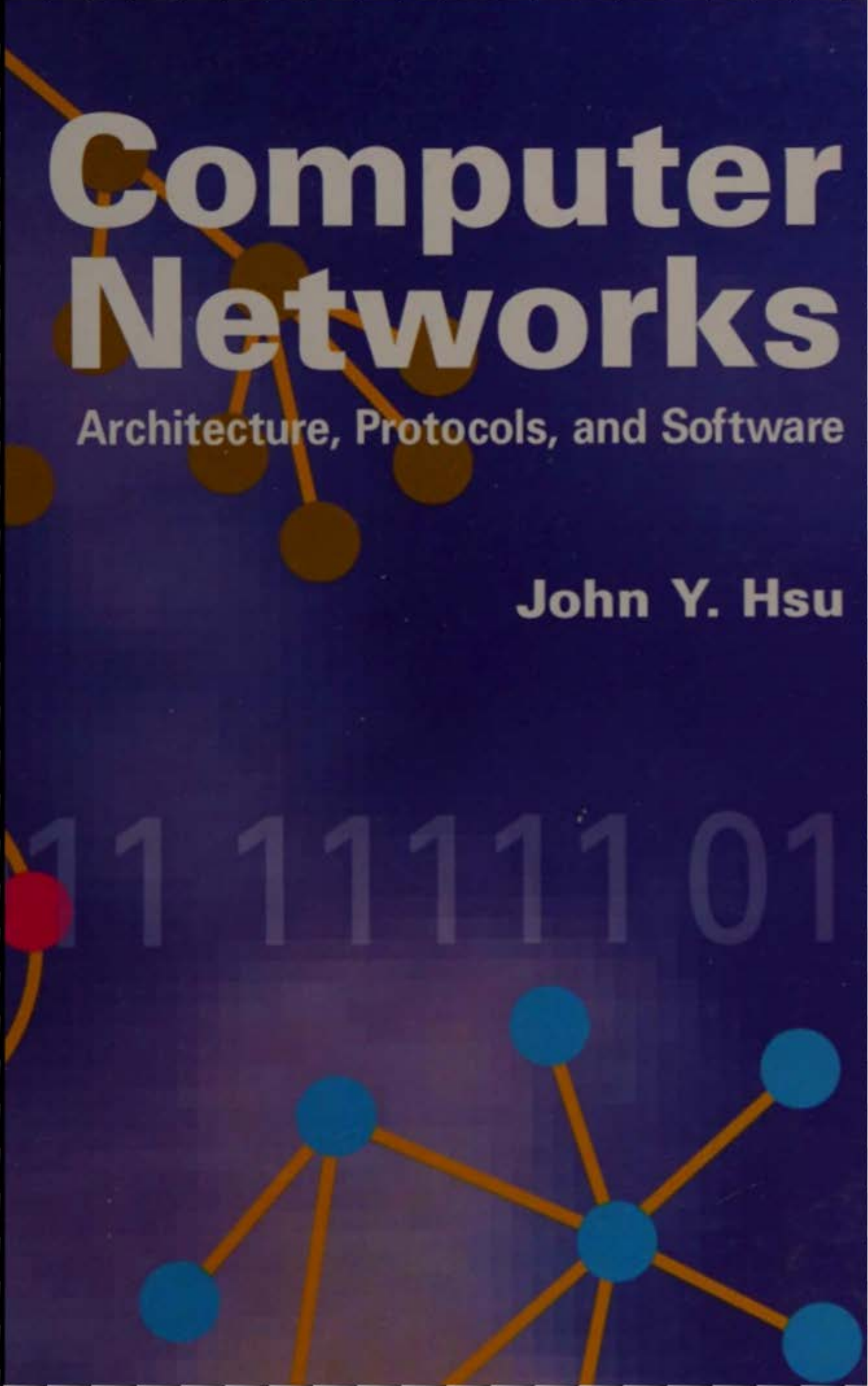


The background of the cover features an abstract network diagram. It consists of several orange lines connecting various circular nodes. Some nodes are brown, some are blue, and one is red. The lines and nodes are scattered across the dark blue background, creating a sense of interconnectedness.

# Computer Networks

Architecture, Protocols, and Software

**John Y. Hsu**

11 11111 01

## Library of Congress Cataloging-in-Publication Data

Hsu, John Y.

Computer networks: architecture, protocols, and software/John Y. Hsu.

p. cm.

Includes bibliographical references and index.

ISBN 0-89006-852-6 (alk. paper)

1. Computer networks I. Title.

TK5105.5.H78 1996

004.6-dc20

96-12832

CIP

## British Library Cataloguing in Publication Data

Hsu, John Y.

Computer networks: architecture, protocols, and software

1. Computer networks 2. Computer networks-Software 3. Data transmission systems 4. Data transmission systems-Software

I. Title.

004.6'5

ISBN 0-89006-852-6

© 1996 ARTECH HOUSE, INC.

685 Canton Street

Norwood, MA 02062

All rights reserved. Printed and bound in the United States of America. No part of this book may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system, without permission in writing from the publisher.

All terms mentioned in this book that are known to be trademarks or service marks have been appropriately capitalized. Artech House cannot attest to the accuracy of this information. Use of a term in this book should not be regarded as affecting the validity of any trademark or service mark.

International Standard Book Number: 0-89006-852-6

Library of Congress Catalog Card Number: 96-12832

10 9 8 7 6 5 4 3 2 1

telecommunications field to achieve endpoint-to-endpoint compatibility of international telecommunications, regardless of the countries of origin and destination. The State Department represents the United States in ITU-T.

International telecommunications standards are also set by the International Organization for Standardization (ISO), which is itself a member of ITU-T. The ISO members include 89 countries, with the American National Standards Institute (ANSI) representing the United States. Note that ANSI is a nongovernment and nonprofit organization whose members are U.S. manufacturers and other interest groups. Proposals submitted by ANSI are usually approved by ISO as international standards.

In the United States, there is the National Institute of Standards and Technology (NIST), which was called the National Bureau of Standards (NBS) until the Department of Commerce changed its name in 1988, and the Institute of Electrical and Electronics Engineers (IEEE), which is the largest professional society in the world. Both agencies tend to set standards for America and submit them to ISO for approval as international standards. All specification and standard documents are available from the respective agencies or from one common source [Common].

### **1.7.1 Layer Specifications**

Using a layered approach, we can divide and conquer the design of network I/O system software. What is a layer? Designers of the network system can designate a task, a group of subroutines, or a hardware controller as a layer. Some basic design philosophies, however, should be observed:

- Each layer should perform one or more well-defined functions.
- Information flow across the boundary should be minimized.
- Changes made in one layer do not require changes in other layers.
- Each layer usually communicates with the layer above or below except the network management routines.

The OSI reference model first proposed by ISO consists of seven layers in its entire protocol stack [X.200]. As network system specifications are concerned, the seven-layer model is fine. But in regard to NOS design, a five-layer model would be adequate [7–9]. The argument is that no layer is necessary if we can get by without it. To solve this dilemma, major firms such as AT&T, IBM, DEC, and HP, have all created their own application programming interfaces (APIs) for the upper layers. A user may essentially request service from any of the upper layers via its API.

The OSI reference model merely provides a recommendation of ideas. In order to connect computers to a network, it is necessary to obey the standards at both the hardware and software levels. Assume that a layer consists of many

software routines that provide one or more well-defined network functions. Between two adjacent layers, the lower layer is a service provider while its immediate upper layer is a service user. For pedagogical reasons, the seven layers of the OSI reference model are listed below:

1. Application (A);
2. Presentation (P);
3. Session (S);
4. Transport (T);
5. Network (N);
6. Data link (DL);
7. Physical (PH).

The A layer is on top while the PH layer is at the bottom. The layer-to-layer connections are shown in Figure 1.7. On the origination node, one application task goes through all seven layers in the protocol stack from top to bottom to transmit a piece of data on the line. On the destination node, another application task receives the same piece of data via all seven layers from bottom to top. The two application tasks are said to be peers, and each task is a peer entity or software routine. Below the application task, each layer communicates with its peer on a remote computer.

At the origination site, there is a physical connection between the node and the computer network. At the destination site, there is also a physical connection between the node and the network. There may be several intermediate nodes in the middle of the route. The upper layers all have direct virtual connections to their peers, as shown by the dotted lines, while the solid lines indicate a physical connection or data path. That is, the bottom three layers on each intermediate node work as a team merely to transmit and receive data. In the seven-layer OSI model, the term *computer network* means a subnet that consists of many communication processors running the bottom three layers.

### *Physical Layer*

The PH layer specifies the physical connection, such as connector shapes, pin assignments, voltage levels, bit positions, data rates, and handshaking procedures at the pin level. The PH layer software mainly consists of I/O drivers, which have access to hardware registers in the communication controller. The group of data bits placed on the line is called a frame. The I/O drivers instruct the hardware controller device to transmit and receive frames. To send a frame, an output driver sets the data bits into the hardware registers of the communication controller, which in turn transmits the bits on the line.

Electronically, a hardware transmitter varies the line voltage as a function of time, and the receiver can detect the voltage change and figure out the bit pattern. If the communication controller is a byte I/O device, the I/O drivers can

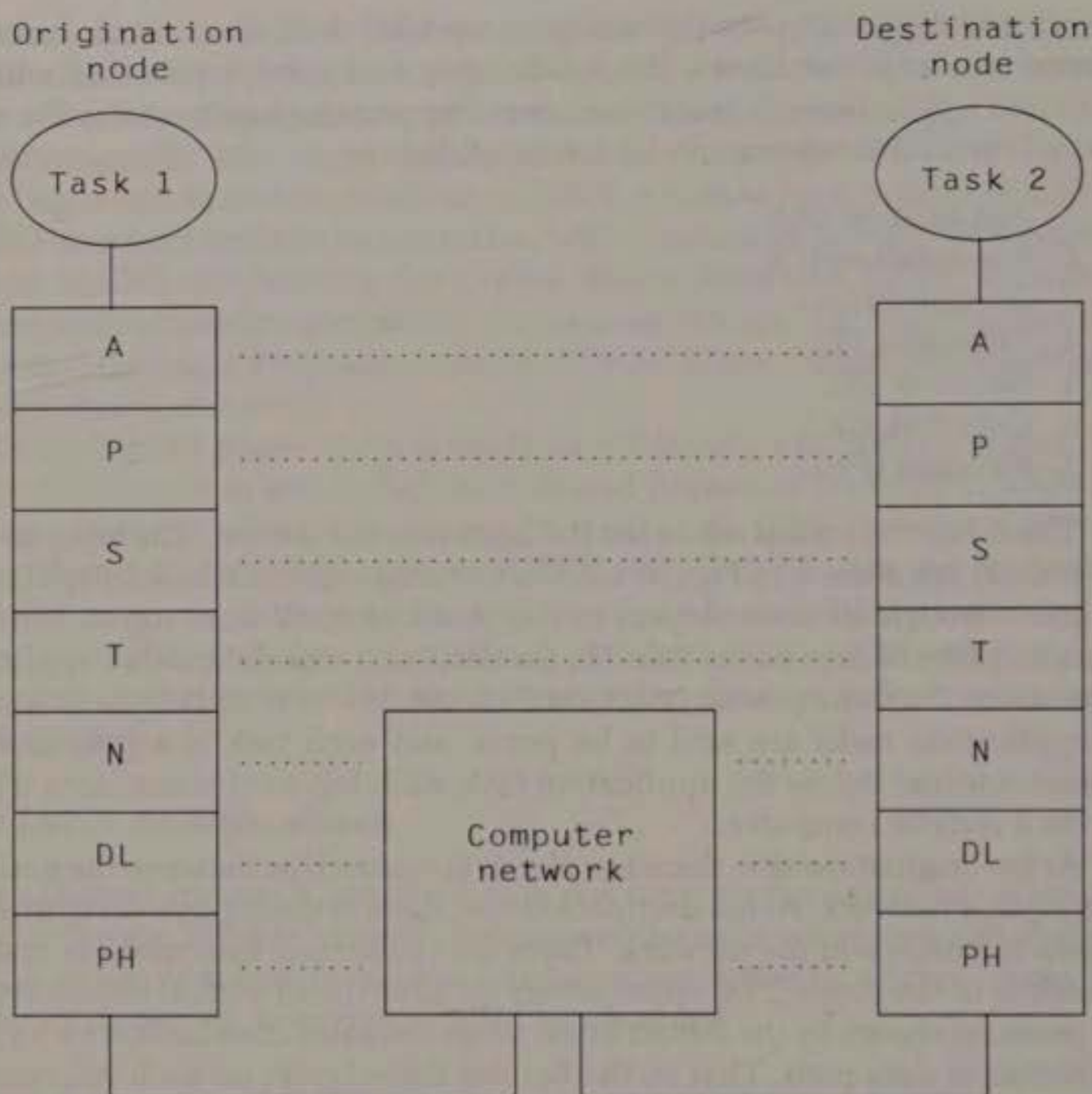


**Figure 1.7** Layer-to-layer connections in an OSI reference model.

team up with the interrupt service routines to make the controller look like a block I/O device. That is to say, the PH layer software enables a service user to transmit a 128-byte data block on the line via one driver call.

### *Data Link Layer*

The DL layer software performs two major functions: error control and flow control between two connection points. Each connection point has its own physical link, which is a hardware transmitter and receiver pair. Therefore, a DL can be thought of as a logical entity that contains hardware and software to ensure correct data transmissions on the line [SDLC]. In the case where a PH link is not reliable, a DL may be because of its software design. That is, whenever a

frame is not correctly received, the sender is notified to retransmit until the frame is correctly received.

The DLs can be thought of as sections of an information highway and considered to be shared resources. If each DL is well-built, we have a very reliable info highway. To accomplish this goal, an error-detecting code is placed in the frame during data transmission. As far as software is concerned, a DL could be a system task that receives commands from the network layer and then issues a driver call to transmit the frame. When there are two or more nodes in a network, we have a routing problem. That is, the system should know how to deliver a piece of data from one node to the next. As a result of this, the N layer is incorporated in the network software design for routing and congestion control.

### *Network Layer*

The N layer mainly handles such tasks as routing control and congestion avoidance. There are two approaches to designing the N layer: the virtual circuit (VC) model and the datagram model. Each approach has its own merits.

A VC can be thought of as a series of data links. It is like a network pipe between two computers with intermediate nodes in the middle. If a packet is dropped into one end of the pipe, it pops out from the other end. In a VC model, the N layer must establish a VC first. After that, the same route must be followed by all subsequent data transmissions. In a datagram model, the N layer delivers a datagram just like dropping a letter in the mailbox. Each datagram contains its own full destination address. The letters having the same address may not follow a fixed route. They will eventually be delivered to the destination, but sometimes a few letters may be mishandled and lost.

Since the N layer provides either VC or datagram services, the programming entity at either end of a VC or datagram is a network service access point (NSAP). Each node that runs the network layer must be assigned a unique network address. The N layer at the origination node receives a message from its service user, interprets the message, and executes the message. A data request message is equivalent to a VC call or datagram call, depending on the N layer design. The N layer may also receive a data indication message from its DL layer asking for routing control.

If a WAN contains a lot of nodes, the N layer needs a large routing table that contains information on all the routes. Whenever a routing decision needs to be made, it must be made within a reasonable amount of time. The key difference between a VC model and a datagram model is that the VC model follows the same route but the datagram model does not.

When two or more users are running on the host and both need network services, a T layer must be in place. Each service user of the T layer may use a programming port to deliver a large message block to its peer.

### *Transport Layer*

The T layer mainly supports the multiplexing functions in a multiprogramming system on the host. It is capable of maintaining a programming port connection between two programs. A programming port is a private software resource to the service user. Below the port, we have either a VC or a datagram, which may be shared. The T layer also handles segmentation and reassembly. As an example, it may accept a message block as large as 16 MB from its user, chop the message into pieces, and deliver each piece in shared memory for the N layer to fetch, interpret, and execute. At the receiving end, the peer T layer knows how to reassemble the message block and deliver it to its user.

### *Session Layer*

Because the T layer services are not easy to use, it is a good idea to provide a system interface between the user task and the T layer in a five-layer model. Hence, the S interface becomes the fifth layer. In the OSI reference model, all three upper layers exist and the application interface becomes the eighth layer on top of application.

The S layer performs two major functions: connection management and synchronization. During the connection phase, a security check is also performed. In concept, the S layer should be simple and elegant, and it was designed as an API to achieve programming convenience [LU6.2, TCP/IP]. In such a design, the T layer details can be hidden from its service user and the services provided by the session layer are easy to use. Note that the interface is part of a NOS and should provide the same set of system calls to all applications.

### *Presentation Layer*

The P layer contains routines, such as compression, decompression, conversion, encryption, and decryption, for manipulating data. They should be designed just as subroutines. People write presentation service routines as commercial software products. Specifications, however, are needed to convert data from one format to another. In fact, the data encryption and decryption functions can be handled by any layer, or even the hardware.

### *Application Layer*

The A layer consists of all the application routines running on the network. Whether the application routines should be considered as a layer remains a debatable issue. One may argue for breaking a particular network application program into two parts, one written by the system people and the other one written by a user. The part written by the system people may be called the A

layer. Consequently, the user needs to interface with the application part in the system, which makes the programming job much easier.

The A layer specifications, however, are necessary. The data format and opcode in a message must be defined for a particular application. For example, in a file transfer protocol, we need to specify subcommands such as get, put, dir, and quit, that are local to this particular application. Each subcommand entered via the keyboard is treated as data to the application service routine, which merely interprets and executes.

By the same token, a banking network must specify the formats of subcommands, such as deposit, withdraw, verify, and report. Because automatic teller machines can only accept one format to transmit data to a bank, it is necessary to standardize the format so that any customer can withdraw cash not only from any automatic teller machine but also from any bank [10]. If a bank uses a different format, then a routine must be employed to do the necessary conversion.

If all the I/O drivers in the PH layer are embedded into the DL layer, then the NAMs merely consist of four layers—namely, DL, N, T, and S—plus the NMT routines. Each layer contains a lot of lines of code, which necessitates an arduous development effort. To communicate between two adjacent layers, the layer primitives are designed for this purpose, as introduced in Section 1.7.2.

### 1.7.2 Layer Primitives

The layer primitives are issued to exchange information between a service provider and its user in an abstract way. In simple terms, a layer primitive may be a request to the target layer or a Confirm from the target layer. A service user may issue a request to the target layer or service provider. After receiving a request, the target layer starts to take action. When the requested service is completed, the target layer sends a Confirm to its user, which is the upper layer. In a way, the layer primitives are used to describe network I/O operations, and they may be designed as subroutines or just messages between two adjacent layers.

An application task at the origination node may issue a system call to send a data message to its peer application task at the destination node. On the origination node, each layer passes a message down until it reaches the PH layer, which actually does the transmission. On the destination node, each layer passes a message up until it reaches the peer application task.

ITU-T has established a uniform syntax to specify layer primitives [X.210]. Each layer primitive starts with a target layer id followed by an underscore, an action, a period, a basic primitive, and parameters enclosed in parentheses. The underscore, period, left parenthesis, and right parenthesis, are terminal symbols in compiler jargon that show up in the language syntax. The metalanguage used to describe the syntax of a layer primitive is shown in Figure 1.8(a).

containing its address mask. From this address mask, the ICMP knows whether a given IP node is attached to this subnet or not.

There are other network application programs, such as network statistics (`netstat`), routing table (`route`), and name server lookup (`nslookup`), that can be invoked by the user. The `netstat` command can be entered by any user to examine the statistics of the network. For example, the following command may be entered to display the routing table:

```
netstat -r
```

where `-r` denotes the routing table display.

The `route` command can be used to add or delete entries in the routing table by a system administrator. For example, a command may be entered as

```
route add default 129.65.17.250 1
```

where `add` specifies one of two options (`add` or `delete`), `default` specifies the default entry in the routing table, `129.65.17.250` is the gateway IP address for the next hop, and `1` denotes that the gateway is a remote one.

Another example is

```
route add net 129.65.128 129.65.54.100 1
```

where the keyword `net` specifies a subnet address. Any IP destination address that matches the given upper 24 bits takes the next hop via the specified gateway.

The `nslookup` command asks the name server to translate a given symbolic name into an IP numeric address, as shown below:

```
nslookup tuba.calpoly.edu
```

After logging on an Internet computer, the user may obtain the description of any system command by entering the manual (`man`) command followed by a particular command name, such as `netstat` or `route`, as shown below:

```
man netstat
```

Finally, we would like to explore the network devices developed in the field for different purposes.

## 5.11 NETWORK DEVICES

A network device is a computer that provides the packet-switching function in a computer network. All network devices described in this section are routers (i.e., switching nodes of some kind using different protocols). The network device executes the bottom three layers, and sometimes the N or DL layers may be very simple or even empty.

### 5.11.1 Packet Assembler/Disassembler

A Packet Assembler/Disassembler (PAD) is a computer that connects many terminals to a packet-switching network, as shown in Figure 5.21(a). It is also a concentrator.

On top of the N layer there is an application layer that reads the characters from various terminals in asynchronous character transmission mode, assembles the characters into a packet, and delivers the packet to its N layer based on X.25. On the X.25 network side, the N layer determines the route and asks its DL to deliver the packet. The DL layer adds its own PCI to the packet and transmits the frame on the line. On the reverse direction of data flow, the application layer receives packets from the X.25 network via the bottom three layers, disassembles the packet into characters, and transmits each character one by one to the destination terminal via the UART port. It should be stressed that on the terminal side, the A layer deals with the PH layer directly and its N and DL layers are essentially hollow.

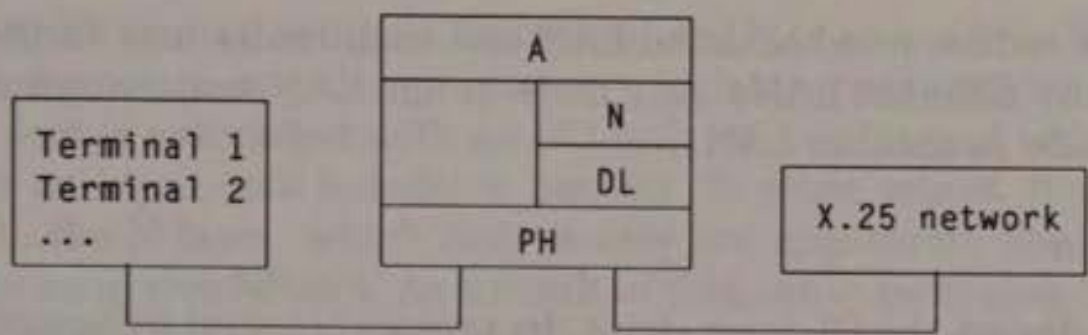
A PAD has UART chips to communicate with the asynchronous terminals at the character level and USRT chips to communicate with the X.25 network synchronously at the byte block level. Via commands from keyboard, a user may connect to any computer in the network.

### 5.11.2 Bridge

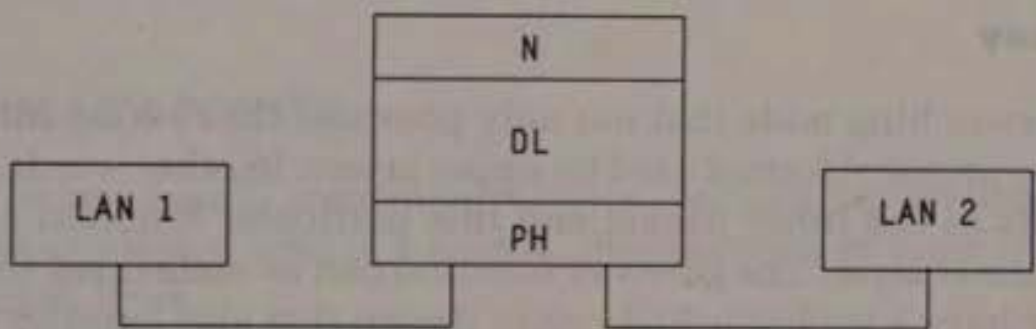
A bridge is a computer that usually connects two LANs, as shown in Figure 5.21(b). The bridge receives and interprets frames for both LANs, which may be homogeneous or heterogeneous. If the frame received belongs to the LAN on the same side, the bridge takes no action. However, if the frame belongs to the LAN on the other side, then the bridge retransmits the received frame to the other LAN. A bridge is a simple router that has only two ports and mainly executes the bottom two layers. Because a bridge does not require much routing intelligence, both its N and A layers can be combined into one. A special bridge protocol data unit (BPDU) is designed to notify the bridge about the topology change. The service user issues a `DL_UNITDATA.request` to transmit this BPDU to the bridge.

### 5.11.3 Router

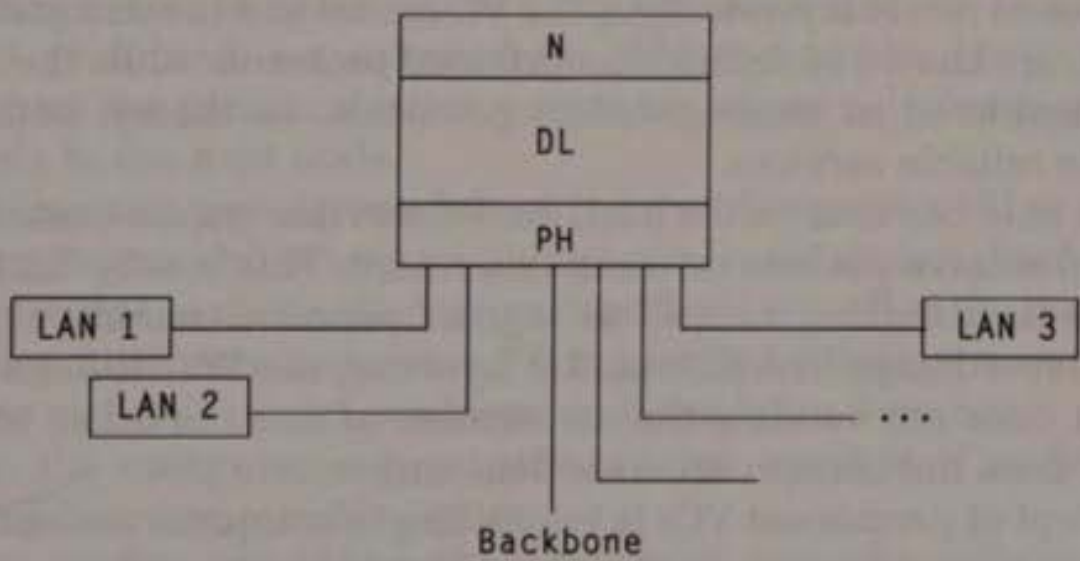
In a datagram model, a router is the switching node to support many different ports as shown in Figure 5.21 (c). Its DL layer receives frames from one port and passes the inner packet to its N layer to determine the route. From the destination node id in the packet header, the N layer determines the next hop. Usually, if the next node belongs to a LAN, the N layer sends the packet to a particular DL using the same or a different protocol. With N and A layers combined into one single layer, the router contains more lines of code than the bridge. In practice,



(a)



(b)



(c)

**Figure 5.21** Network devices: (a) packet assembler-disassembler, (b) bridge, and (c) router.

many routers resides in a backbone LAN and each router may further be connected to many different LANs. Any node in one LAN can communicate with a different node in another LAN.

### *Encapsulation*

In the router design, the DL layer strips the protocol control information off the frame and delivers the inner packet to its N layer. After determining the route, the N layer sends down the packet to a DL, which in turn adds the new PCI to the packet and transmits the frame on the line. Encapsulation means that PCI is added to the packet before transmission.

### **5.11.4 Gateway**

A gateway is a switching node that not only provides the routing function, but also converts the protocol format used by upper layers. In other words, a gateway modifies the bits in the inner packet and this particular function can also be combined into the N layer. The gateway function can be embedded into a router or bridge, so we have a sophisticated router design that also handles the format conversion function.

## **5.12 CONCLUSIONS**

We have discussed two N layer designs: the VC model and the datagram model. The VC models are known as connection-oriented protocols while the datagram models are considered as connectionless protocols. In theory, both models should provide reliable services.

If there is only one user on the host, the VC services are adequate for a user to transmit and receive packets between two nodes. This is why X.25 embeds the end-to-end handshaking as well as segmentation/reassembly capabilities into its packet layer design. The X.25 packet layer may use SWP to detect missing packets, but it does not validate the correctness of data. In other words, the packet header does not contain an error-detecting code.

The concept of permanent VCs is interesting in computer network design. As an example, we could have a mesh network backbone just like our interstate highway system. Horizontally, we have Interstate 10, 20, . . . to 120 and vertically, 5, 15, . . . to 95. Each VC has a global unique id. In the packet header, there should be a VC id, but also the destination address in order to provide the correct exit point. For example, if we take Interstate 70 and decide to exit at Denver, 70 is the unique VC id and Denver is the destination address.

As technology advances, new devices are developed to transmit real-time entertainment data, such as voice or video, where speed is critical but data correctness is not. The primary goal is to transmit/receive data in time. If no

handshaking is required at the DL layer level, the N layer becomes the main task running on the communication processor, which interacts with many I/O tasks. Each com port is associated with an I/O task pair: an output task to transmit packet and an input task to receive packets. In other words, the DL layer is merged into the N layer, which may or may not implement the handshaking protocol between two NSAPs. As a result of this, other switching devices such as frame relay and asynchronous transfer mode are developed for future network design. They are also the future protocols proposed for narrowband and broadband ISDNs. Such switching devices are all routers with different names that can be interconnected to make a WAN or LAN backbone. Therefore, we will continue discussing their unique design features and specifications in the next chapter.

### 5.13 SUMMARY POINTS

1. The N layer mainly transmits/receives packets between two nodes. The basic transmission unit is a packet, therefore the N layer is also known as the packet layer.
2. A packet may contain either control information or data.
3. Each computer on the network is assigned a global node address that is unique to the network. The node address is actually a numerical address represented by a string of bits.
4. The N layer mainly handles the routing function. It also provides congestion control as well as concatenation/separation of smaller packets.
5. If a DL protocol is not enforced, the N layer could be designed as the main driver on the communication processor whose sole function is to deliver packets to the next node.
6. There are two popular models of N layer design: the VC or the datagram.
7. A datagram model is very much like our Postal system. Each datagram has a header that contains the receiver's address as well as the sender's address.
8. The key difference between a VC model and a datagram model is that the former maintains strict roadsigns and the latter does not.
9. When the computer network has a large number of nodes, distributed routing becomes a viable option.
10. Source routing means that the origination node determines the route.
11. The routing table is the local database used by the N layer. The routing table specifies the topology, routing information, and so forth of the network.
12. Congestion means that a node has encountered a traffic jam and the N layer may report this condition to the upper layer or network management routines for remedial action.
13. The N layer does not perform any I/O operation directly, therefore it should execute from beginning to end without interruption. Even if interrupt is allowed when the N layer executes, control should be returned immediately