13 THE NETWORK LAYER

Like all the other OSI layers, the network layer provides both connectionless and connection-oriented services. The upper layers and transport provide these two types of service to satisfy a broad range of application and end-user needs. This is universally recognized as a "good thing." In the network layer, the presence of two types of service is generally considered to be a "bad thing." The reason for this is simple: everything above the network layer is host-specific and can be tailored to suit a particular application running among some set of mutually consenting hosts without affecting the communications of other hosts using the same internetwork; the network layer, however, provides the fundamental connectivity without which no communication of any kind can take place among hosts. Most people agree that having one type of service at the network layer would be preferable to having two; as always, though, the problem stems from having to decide *which one*.[1]

The TCP/IP architecture, which from the beginning was based on an "internetworking" model of the network layer,[2] avoided this controversy entirely; the TCP/IP network layer is exclusively connectionless.[3]

1. There are people who attempt to justify a "diversity of needs" at the network layer, but their sense of what constitutes *interoperability* is quite different from ours.
2. The best (and certainly the most succinct) description of the fundamental architectural premise of the TCP/IP network layer is the one that Vint Cerf uses to describe the TCP/IP internetworking model: "I P on everything."
3. The requirement to support new types of service in the Internet, such as real-time service for voice and video, may change the traditional "datagrams-only" model of the TCP/IP network layer. The new concept of a "flow," which is neither a connection nor datagrams, promises to make the TCP/IP network layer a bit more complicated (but also a bit more useful) in the near future (Partridge 1992).

The issues surrounding a "choose one" decision at the network layer were of such a highly charged political, economic, and emotional nature that convergence on a single networking solution for OSI was, and remains, a pipe dream. Compounding the problem, there is no generally recognized way to interwork between connection-oriented and connectionless networks. ISO/IEC 8648: 1987, *Internal Organization of the Network Layer*, does not provide a solution; however, demystifying its contents does provide meaningful insight into the problem, so it is a logical place to begin.

Architecture: The Internal Organization of the Network Layer

It is virtually impossible to understand the purpose and contents of the *Internal Organization of the Network Layer* standard without a historical perspective. In the abstract, the OSI network layer provides *the* switching fabric over which end systems communicate. The notion of a single, open networking environment for data is a powerful one, since it implies a very broad scale of connectivity. In fact, it is not hard to imagine (and even easier to desire) an *open* data network that is as ubiquitous as the voice network.

The postal, telephone, and telegraph (PTT) administrations of many national governments[4] have long been committed to standards as processed for telephony under the CCITT. A strong political incentive existed at the time of OSI network-layer standards development to institute a single, uniform network service standard for data. Publicly, advocates of this *network-centric* view claimed that a single service, modeled after the voice network, would scale well—the voice network certainly did. A framework for multiorganizational administration existed for nasty issues like addressing and interworking. Moreover, they continued, similar service characteristics could be expected irrespective of the location of the source and target applications. Privately, these parties hoped that by bringing computer communications further under the administration of the PTTs, they would "contracept" further expansion of the private networking offered by large multinational computer vendors. By so doing, they hoped to promote the interests of their native (especially European) com-

4. Some countries, including the United States and the United Kingdom, do not have a single "national" public network agency. To accommodate these countries, the CCITT considers that *recognized private operating agencies* (RPOAs), such as British Telecom and AT&T, have standing equivalent to the PTTs of countries with national public network agencies.

puter vendors and strengthen both their own domestic and international markets. In short, they hoped to expand the market for public networking services, helping their home computer companies in the process.

OSI network-layer development also began when data communication was, by today's standards, in its infancy. Turnkey remote job entry and terminal-to-mainframe communications were the principal applications. Local area networks were fledgling and pricey technologies. On-premises terminal connectivity was achieved by a hodgepodge of largely proprietary asynchronous "poll-select" technologies, and wide-area bandwidth was expensive, even for modest kilobit-rate services, so applications (and end users) were made to tolerate delay. Compared to even the smallest of today's regional TCP/IP networks, private networking was a small-scale, predominantly mainframe-to-mainframe enterprise, and the wide area network was the center of the universe.

When the price of local area network (LAN) technology dropped, the "public network is everything" paradigm was shattered forever. On-premises bandwidth was cheap and plentiful; applications and processing were distributed among large numbers of increasingly powerful yet smaller computers, initially across local area networks, but soon across higher-bandwidth wide area services. Thus, two paradigms for network service—local area network and wide area network—collided precisely at the time of the development of standards for the OSI network layer.

Renegades from the community of local area network equipment manufacturers and consumers rose to challenge the network-centric view of the world. Unlike voice service, they argued, for which the service characteristics necessary to support the primary application (speech) have remained constant for a century, the characteristics of network services necessary for computer-to-computer communications vary widely from application to application. Terminal access to remote computers requires only low bandwidth and can tolerate delay, whereas networked file services like Sun's Network File System (Sandberg 1988) require high bandwidth and low delay, but only for short periods or *bursts*. Other applications (electronic mail, file transfer) have requirements somewhere between these extremes.

Applications also have varying requirements for data integrity, reliability in data transfer, and other characteristics. The local area network proponents argued that true distributed applications needed LAN-like characteristics across a wide area. These can best be provided by adopting the internetworking protocol concepts demonstrated in the Defense Advanced Research Projects Agency experiments and deployed in proprietary architectures such as Xerox Network Systems (XNS), Digital

Network Architecture (DNA), and Burroughs Network Architecture (BNA). Privately, vendors of host computers, routers, and other computer-communications equipment hoped to use OSI as a means of penetrating hitherto single-vendor "closed-shop" environments held by the industry giants: in an open platform, smaller, nimbler manufacturers could compete for market share by specializing rather than offering a comprehensive micro-to-mainframe line of computers. The industry giants contributed their proprietary solutions to universal networking problems in an attempt to ease the inevitable transition. Computer vendors big and small—hereafter referred to as *host-centrics*—had a common objective: they wanted a solution that would enable them to sell more communications as well as host equipment.

The OSI network layer is thus *where net-worlds collide.* Two substantially different views of the way in which network service should be provided had been defined. Champions for both causes stepped forward, lines were drawn in the sand, and battles ensued. The *Internal Organization of the Network Layer* was to become the demilitarized zone of the "police action" initiated to unify the OSI network layer. Within this standard, it was expected that differences would be reconciled and a framework for interconnectivity would be defined.

The *Internal Organization of the Network Layer* began as a microarchitecture document for the OSI network layer. It was to provide a functional description of the network layer that related the OSI architectural model of the network layer to the "real-world" networks, switches (routers and bridges, but also carrier network packet switches), and host computers that comprised the OSI environment (for a historical perspective, see Hemrick [1984]). It was gradually transformed into a means of describing how to retrofit the OSI reference model view of the network layer onto existing real-world networks.

All discussions concerning "architecture" begin with the seemingly innocent question, "What is X?" Questions of this nature are the very essence of why, in many OSI standards meetings, a great deal is said and little is done. The ISO group responsible for producing the Internal Organization of the Network Layer *spent no less than three meetings fine-tuning the definition of a* subnetwork *so that all parties could point to their individual "collection of equipment and physical media which forms an autonomous whole and which can be used to interconnect real systems for purposes of communication" (ISO/IEC 8648: 1987) and say, "Now that's a real subnetwork."*

The *Internal Organization of the Network Layer* spends lots of time explaining the real world. Real subnetworks, particularly those that provide a public service tariffed by a common carrier, have a specific protocol that *data terminal equipment* (DTE) uses to access the packet-switching equipment of the public network provider; such protocols are called *subnetwork access protocols* (SNAcPs). Local area network medium access control (MAC) protocols, accompanied by logical link control (LLC) procedures and protocol, can also be said to be subnetwork access protocols.[5] There are consequently many subnetwork access protocols in the real world, providing a wide range of subnetwork services; in fact, creation of subnetwork access protocols is something of an annual event in the standards community.

Most or all of the subnetwork access protocols used in real-world subnetworks don't provide the OSI network service: some functions—for example, the ability to convey the very long and variable-length OSI network service access point addresses—were neither anticipated nor provided for in protocols such as CCITT Recommendation X.25-1980 or the TCP/IP internet protocol (RFC 791).

The proposition put forth in the *Internal Organization of the Network Layer* is that the (presumably small) discrepancy between the OSI network service and the service provided by each subnetwork can be fixed by adding functionality to "enhance" the subnetworks and the equipment connecting them (and connected to them) in one of several ways.

Hop-by-Hop Harmonization

One alternative is to pile functions and protocols on top of each individual subnetwork access protocol, one by one, so that each one is elevated to the level of the OSI network service. This is a case-by-case solution; hence, the term *hop by hop*. This method involves identifying those elements of the OSI network service that are missing from a given subnetwork access protocol and *adding* them. According to this strategy, if subnetwork A could provide all of the OSI network service features except the ability to signal a RESET, one could incorporate a RESET capability into subnetwork A's subnetwork access protocol, and the enhanced subnetwork A would then be capable of providing the OSI network service. If another subnetwork, B, could provide all but the ability to convey OSI network-layer addressing, one could extend the addressing capabilities of subnetwork B's sub-

5. By the time the ISO network-layer committee decided upon the term *subnetwork access protocol*, all of the straightforward acronyms containing the letters *SNA* were taken. In particular, it was obviously inappropriate to use *SNA protocol*, and the IEEE 802.1 committee had only recently devised the logical link control subnetwork access protocol (LLC/SNAP); hence, it was necessary to include a lowercase *c* in ISO's acronym.

network access protocol, and the enhanced subnetwork B would then be capable of providing the OSI network service as well. One could then plug subnetworks A and B together, and real end systems attached to subnetwork A could use the OSI network service to communicate with real end systems attached to subnetwork B.

In the real world, making modifications ("enhancements") to every subnetwork access protocol would require changes in far too many end systems, and end-user reaction to such an effort would be ugly indeed (Q: "Why did it take God only six days to create the world?" A: "He didn't have to worry about the installed base . . ."). Better to pile stuff on top of a handful of the most important network protocols to make them support the OSI network service, which is exactly what ISO and CCITT did. The *Internal Organization of the Network Layer* prescribes the use of a protocol *between* the transport protocol and the subnetwork access protocol to convey the missing elements of the OSI network service. As an example, the ability to carry big addresses was missing from the 1980 version of the X.25 packet-level protocol; specifically, the called and calling address fields of the *call request* packet were too small to convey jumbo OSI network service access point addresses. Oh so clever bit-twiddlers devised a way to employ the *call user data* facilities field of the call request packet to convey OSI network service access point addresses and other missing elements of the OSI connection-oriented network service, and thus extend the life of the data circuit-terminating equipment (DCE) in those public networks in which X.25-1984 availability was not imminent. Since the use of this particular technique was unique to X.25-1980, and similar techniques could be devised to operate over other existing subnetwork access protocols with shortcomings, the notion of a *subnetwork-dependent convergence protocol* (SNDCP) was born. (See Figure 13.1.)

In theory, this process could be applied to all subnetworks over which the OSI network service might be provided; in the extreme case, all the subnetworks in the world. Thus, the degree to which this approach can be successfully applied is directly related to just how many subnetwork access protocols one must modify and how many subnetwork-dependent convergence protocols one must create; i.e., how *heterogeneous* the existing subnetwork environment is. Since the network-centrics' view of the world was that there was considerable *homogeneity* among carrier data networks, this seemed to them that the number would be manageably small.

Internetworking Protocol Approach

Host-centrics approach the problem of subnetwork interconnection with a radically different perspective. The host-centric view of the problem

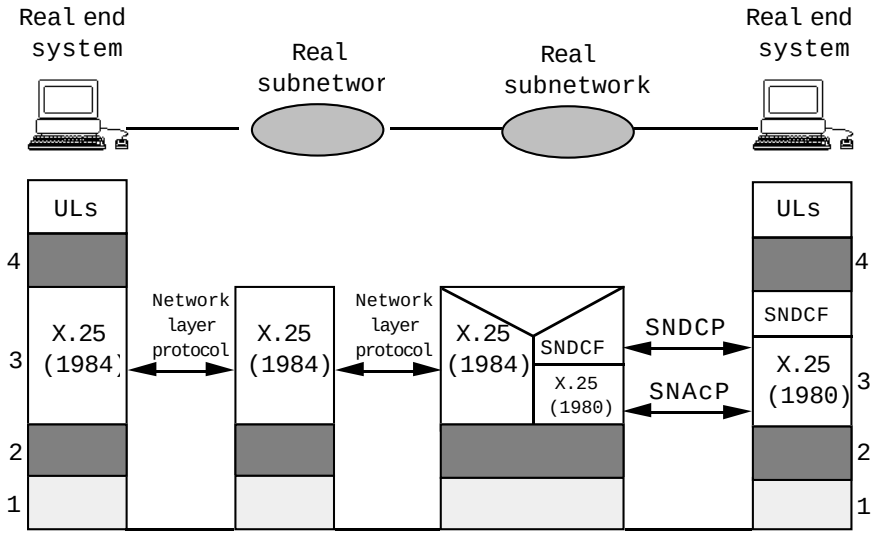


FIGURE 13.1 Hop-by-Hop Harmonization

reflects what folks who deal with LAN-based networks see every day:

- There are many different types of subnetworks in the world.
- They will inevitably be connected together.
- Trying to make them all look exactly alike is hopeless.[6]

For the heterogeneous environments encountered in the real world every day, the practical thing to do is to define one protocol that assumes minimal subnetwork functionality and place it firmly on top of every subnetwork access protocol; i.e., define a *subnetwork independent convergence protocol* (SNICP).

Technically, it is relatively simple to design a subnetwork-independent convergence protocol: treat every subnetwork and data-link service as providing a basic data pipe. Each pipe should support a service data unit large enough to accommodate the header of the subnetwork-independent convergence protocol and a reasonable amount of user data. This is the IP or OSI connectionless network protocol (CLNP) model of networking. In Figure 13.2, CLNP operates in end and intermediate systems in the same manner as IP operates in hosts and routers. Every subnetwork and data-link service that is to provide an underlying, supporting

6. In fact, electrical engineers have demonstrated that they can invent new networking technologies much faster than international standards bodies can develop and approve new subnetwork dependent convergence protocol standards.

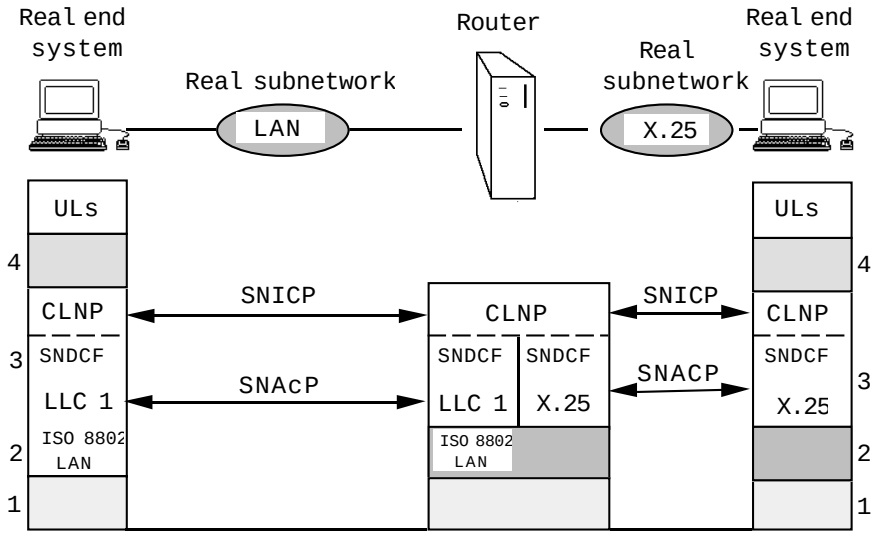


FIGURE 13.2 Internetworking Protocol Approach

service to CLNP must be capable of transferring 512 octets of data.

These minimal requirements are easily accommodated over ISO/IEC 8802 local area networks[7] without introducing convergence protocols; in such cases, CLNP packets are mapped directly onto the ISO/IEC 8802-2 logical link control (ISO/IEC 8802-2: 1990) service data units. The mapping is called a *subnetwork-dependent convergence function* (SNDCF). With different subnetwork dependent convergence functions, CLNP can also be run over connection oriented subnetworks like X.25 public data networks, X.21 circuit-switched data networks, or switched data links. However, the subnetwork-dependent convergence functions are more complex, since they must deal with *subnetwork connection management*. (See "Use of X.25 to Provide a Subnetwork Service in OSI Networks," later in this chapter.)

Hop-by-hop harmonization is something of a bottom-up approach: you look at what your subnetwork lacks and pump it up so that it offers the OSI network service. Internetworking is more of a top-down approach: begin with the assumption that the transport protocol will do what it is supposed to do, define a simple data-transmission service at the network layer, and write a protocol that allows you to forward packets over any underlying bit pipe.

7. The ISO/IEC standards for local area networks, consisting of multiple parts of ISO/IEC 8802 (8802-2, 8802-3, etc.), are equivalent to the corresponding IEEE local area network

Although the use of CLNP to provide a uniform connectionless network service over any combination of underlying "sub"networks seems natural to anyone familiar with the internetworking architecture that underlies the TCP/IP Internet (and many vendor-proprietary networks), it was politically naive to propose it as an alternative to hop-by-hop harmonization in the early years of OSI development. Why? The answer may be found in the phrase "placing it firmly and uniformly on top of every subnetwork access protocol," which suggests that those subnetwork access protocols—the pride and joy, not to mention the economic staples, of many large public network service providers—are somehow deficient, not up to the demands of providing the OSI network service. To recognize themselves as providers only of edge-to-edge service (from subnetwork entry-point to subnetwork exit point) rather than end-to-end service (from end user to end user), the public networks would have had to abandon their claims of networking omnipresence and omniscience. Naturally, they were initially unwilling to do this. With hindsight, however, it can be seen that their reluctance to recognize the importance of internetworking delayed the completion of work on the OSI network layer for at least five years and is one of the reasons why OSI networking today is more of a "missed opportunity" than a reality.

As if to prove that history repeats itself even in fields with the comparatively brief history of networking, the basic mistake of many X.25 proponents of the late 1970s—believing that a single network technology could be extrapolated into a monolithic global data network—is being replayed in the 1990s by some extremist proponents of asynchronous transfer mode (ATM), who argue that "internetworking" is obsolete, because they can build a worldwide ATM-everywhere wide area network that will "seamlessly" interconnect ATM local area networks, and there won't be any need to have Ethernets, token rings, fiber-distributed data interfaces, or any other "old" network technology. And nothing better will ever be invented. Ever. Really.

Connections or Connectionless?

Before we close the subject of OSI network layer architecture, it will be useful to examine the issues that can be debated when political hats are removed and the technical pros and cons of connection-oriented networks and datagram networks are examined.

Network connections generally have the following characteristics:

- Once a network connection is established, all data packets travel along the same path. Network connections thus offer low routing

standards, consisting of multiple parts of IEEE 802 (802.2, 802.3, etc.).

overhead, but they are not resilient; if one link in the path breaks, the connection is broken.

- Network connections provide fixed paths with guaranteed quality of service characteristics (bandwidth, delay, residual error rate); during connection establishment, resources are reserved at each "hop" to minimize quality of service variability. Although this is good for the established network connections, the reserved resources cannot be used by or for other network connections while an established network connection is idle.
- Network connection state must be maintained at every intermediate "hop."
- Network connection-establishment overhead is unacceptable for bursty data. Some applications require very high bandwidth for a very brief spurt or "burst." For example, if a user of a distributed file service attempts to retrieve a file from a remote server across a local area network, that user wants the read completed as quickly as if the file were stored locally (see Sandberg, 1988). The network service must exhibit both high throughput *and* low latency. Network connection establishment is more time-consuming than sending a datagram and in these circumstances is ill-advised; consider how difficult it might be for a file server to reserve a virtual circuit having the bandwidth necessary to read a 4-megabyte file in some measure of milliseconds and do so for many clients.

Connectionless data transfer has its own set of strengths and weaknesses:

- Datagrams are a best-effort delivery; reliability mechanisms, if required, must be provided at the transport layer.
- State machine handling of datagrams at intermediate hops is greatly simplified when compared to network connections.
- Since each packet is routed independently, routing overhead is imposed at each hop. In most routing systems, however, routing is adaptive in the face of failures.
- Resources are used as needed (no resource reservation per network connection); however, to minimize quality of service variability, multilevel congestion-management mechanisms and possibly type of service routing may be required.
- There is no connection setup. Resources are allocated as needed or "on demand." Connectionless data transfer is better suited for supporting bursty data applications.

A colleague (Winston Edmond, who works with one of the authors

at Bolt Beranek and Newman) has suggested a useful thumbnail description of the difference between datagrams and connections:

- *Datagrams:* The hosts don't tell the network anything about the traffic they are about to send. In this case, the best the network can do is to monitor the dynamic behavior of the hosts and hope that past history is a guide to future behavior (and that the network designer has provisioned the network so as to accommodate most, if not all, such behavior).
- *Connections:* The hosts tell the network what they want to do and ask for guarantees (bandwidth, maximum delay, maximum error rate, etc.). In this case, there is an actual setup phase during which the required resources are allocated (if possible; explicit connection setup also provides the network with the opportunity to refuse service entirely in situations in which it cannot provide the requested guarantees), and the success or failure of the setup is reported to the hosts before actual data exchange begins.

(Edmond goes on to compare these two types of network service with a hybrid called "flows," in which the hosts tell the network what they are about to do but don't ask for any guarantees—thereby enabling the network to improve somewhat its otherwise poorly informed anticipation of their future behavior without forcing it to preallocate its resources in order to be sure of meeting static per-connection performance criteria. See Partridge [1992] for a discussion of "flows.")

There are more pros and cons for both types of service, to be sure. Some of these are bit niggling (e.g., datagrams require more "header stuff" than network connections once the connection is established) and some religious (but we've *always* offered connections!). Although not exhaustive, these lists provide readers with enough insight to appear to be "in the know" the next time the issue of "connections or datagrams" comes up during polite dinner conversation.

The "Connections versus Connectionless" standoff is often cited as the ultimate tragi-comedy in OSI. But although the OSI environment is certainly partitioned into connection-oriented and connectionless worlds at the network layer, and different transport protocols are used over both types of network service, there remains hope for reunification. Several alternatives exist:

- *Build network relays. These are truly awful beasts. They constrain topologies, and are difficult to build. And they most often rely on the operation*

of a common transport protocol (class 4); but the presence (or absence) of TP4 is one among many other bones of contention that caused the partitioning of the network layer in the first place!

- *Build transport relays rather than network relays. These, too, are difficult to build; they violate the OSI reference model (transport functions are supposed to operate end-to-end); and they are generally considered to be broken. For example, if you attempt to relay protocol mechanisms designed to support certain transport functions (e.g., security provided by transport protocol encryption), you break them.*
- *Provide interworking between connection-oriented and connectionless networks. The solutions offered to date are both complex and constraining.*
- *Support both services in all end and intermediate systems. This is generally perceived to be prohibitively costly to end systems.*
- *Build transport bridges. These have been demonstrated to be better alternatives than transport relays, especially when two useful transport protocols—TCP and TP4—play roles in the bridging mechanism. Transport bridges are often the only practical solutions.*
- *Pray that, over time, attrition will eventually reduce all the useless combinations to a single network service and a single transport protocol. A good place to start would be the elimination of all the OSI transport protocol classes except class 4 (TP4).*

Bridges and attrition are the best bets. Many of the same issues exist for multiprotocol (TCP/IP and OSI) networking and are discussed in Chapter 16.

Connection-oriented Network Service

An OSI network connection has the same three fundamental phases of operation—connection establishment, data transfer, release—as all previously described connection-oriented services. The primitives and parameters provided at the network layer for a connection-oriented service are illustrated in Table 13.1.

Again, the network connection is modeled after a telephone conversation: dial the phone, talk, hang up. However, a couple of astonishing perturbations are introduced:

- Any party can decide that it has lost track of the conversation (reset).
- The phone company will tell you whether or not the party you called heard what you said (receipt confirmation).

Where did these come from? Welcome to the world of grandfather-

TABLE 13.1 CONS Primitives

Primitives		*Parameters*
N-CONNECT		
	request indication	Called address, calling address, expecited data option, QOS, recept confirmation option, NS-userdata
	response confirm	Called address, responding address, expedited data option, QOS, receipt confirm option, NS-userdata
N-DATA		
	request indication	NS-userdata, confirmation request
N-DATA-ACKNOWLEDGE		
	request indication	
N-EXPEDITED-DATA		
	request indication	NS-userdata
N-RESET		
	request indication	Reason
	response confirm	Originator, reason
N-DISCONNECT		
	request indication	Reason, NS-userdata, responding address (Originator, reason, NS-userdata, responding address)

ing existing protocol features into new service definitions. In principle, a service definition provides the template for protocol design and development. Services are identified, functions required to support those services are defined, and a protocol capable of performing the functions and providing the services is specified.

In practice, some of the principle was lost. OSI's *Connection-oriented Network Service Definition* (ISO/IEC 8348: 1987), is a marvelous example of reverse service engineering. CCITT Recommendation X.25 for public packet-switching networks was published before the development of the connection-oriented network service. It had already been deployed by PTTs and common carriers to support *CCITT applications* (e.g., the

TABLE 13.2 Correspondence between Connection-oriented Network Service Primitive and X.25 Packets

CONS Primitives		*X.25 Packet*
N-CONNECT	request	Call request
	indication	Incoming call
	response	Call accepted
	confirm	Call connected
N-DATA	request, indication	Data
N-EXPEDITED-DATA	request, indication	Interrupt
N-RESET	request	Reset request
	indication	Reset indication
N-DISCONNECT	request	Clear request
	indication	Clear indication

"triple-X" recommendations—X.3, X.28, X.29 [CCITT Recommendations X.1–X.32 1989]—that support terminal application protocols), and for many, the use of X.25 to support the OSI network service was an expedient and politically correct solution.

To take advantage of as many features in X.25 as possible, advocates adopted a Machiavellian attitude toward the OSI connection-oriented network service: manipulate the service to conform to the service offered by X.25. This "Never let an architecture stand in the way of your implementation" approach is best illustrated by the nearly one-to-one correspondence between OSI connection-oriented network service primitives and X.25 packets, illustrated in Table 13.2.

X.25 Packet Level Protocol—OSI's Connection-oriented Network Protocol

Although ISO and CCITT devote reams of paper to codifying the operation of network connections (ISO/IEC 8208: 1987; ISO/IEC 8878: 1987; ISO/IEC 8881: 1989), it is rather easy to describe how the OSI network service is provided by the X.25 packet level protocol without descending to the bit level of detail.

The ISO/IEC standard for X.25 defines two interfaces:[8]

1. Between a computer (data terminal equipment, or DTE) and a carrier network node (data circuit-terminating equipment, or DCE)
2. Between two computers without an intervening public network (DTE/DTE)

8. CCITT Recommendation X.25 defines only the DTE/DCE interface.

The DTE/DCE physical interface can be leased or switched. The DTE/DTE physical interface can be a leased line, a switched circuit, or a local area network (ISO/IEC 8881: 1989). Both virtual-circuit and permanent virtual-circuit modes of operation are accommodated. In the virtual-circuit mode of operation, a connection must be set up, used, and disconnected. With a permanent virtual-circuit arrangement, a subscriber has facilities and resources permanently assigned between two DTEs by a carrier (the call is "always there"). These alternatives are illusrated in Figure 13.3.

Figure 13.4 illustrates (in the abstract) the three phases of a network connection established across a switched (VC mode of operation) X.25 circuit.[9] An N-CONNECT.request primitive causes DTE A to issue an X.25 *call request* packet. The DCE receives this request, and the network constructs a path between DTE A and DTE B, allocating and reserving resources of packet switches that comprise the path along the way. The

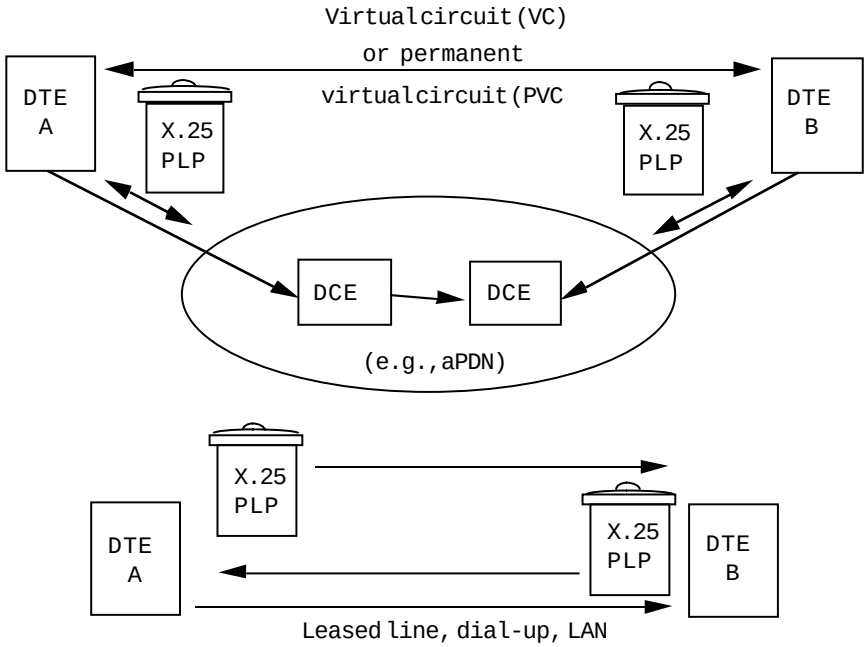


FIGURE 13.3 X.25 Interfaces

9. The procedures for operating the X.25 packet-level protocol between DTEs without an intervening public network are virtually identical. This description covers only those aspects of X.25 VC operation relevant to the packet level of operation. At the data-link level, both leased-line and circuit-switched access to a DCE are provided by many carrier networks. CCITT Recommendation X.32 describes additional packet-level considerations (e.g., user identification) typically required for switched access to a public network.

terminus DCE issues an *incoming call* packet to DTE B, which causes an N-CONNECT.indication. DTE B issues a *call-accepted* packet, and DTE A eventually receives a *call-connected* packet from the DCE. All X.25 *data* packets exchanged during this virtual call are forwarded along the path created by the DCEs during network connection establishment. When the network service user at DTE A elects to close the network connection, it issues an N-DISCONNECT.request, causing DTE A to issue a *clear request* packet. This is processed by the DCE and results in the generation of a *clear indication* packet to DTE B. DTE B acknowledges receipt of the clear indication by returning a *clear confirmation* packet. This packet eventually makes its way back to DTE A, causing the generation of an N-DISCONNECT.confirmation. Network connection disconnection is described in ISO/IEC 8348 as "unconditional and possibly destructive" (honest, this is what it says, you can look it up).

During the data phase, the X.25 packet level protocol provides sequence control, flow control, expedited data (using the *interrupt* facility and packets), and error notification. *Reset*, carefully distinguished as a feature rather than an error notification, offers a means by which a network connection "can be returned to a defined state and the activities of two network service users synchronized."[10] Reset can be network service provider– or network service user–initiated. Negotiation and use of the *D-bit* facility enables the *receipt confirmation* function of the OSI network service. Receipt confirmation allows a network service user to request end-to-end acknowledgment of the delivery of data it is transmitting. In its implementation, receipt confirmation signals to the network service user that the X.25 data packets transmitted have been acknowledged. Although this is a clear violation of the OSI reference model (the internal machinations of an [N]-layer protocol should be hidden from an [N]-service user), the facility was "grandfathered" into the OSI connection-oriented network service over the protests of purists, largely because it was a tariffed facility.

Use of X.25 to Provide a Subnetwork Service in OSI Networks

When a connectionless network service is provided to communicating transport entities in OSI, the X.25 packet-level protocol plays the role of a

10. ISO/IEC 8348 does not provide a definition of "defined state," nor does it describe what activities the network service users "synchronize." From personal experience, we may say that defined state is "Some of your data are lost; I hope someone was keeping track . . ." The synchronization activities are transport protocol–dependent; for example, upon receipt of a reset indication, transport classes 0 and 2 give up and disconnect (big help . . .), but classes 1 and 3 initiate resynchronization procedures (see Chapter 12). TP4 is completely resilient to reset indication, treating it like any other error . . . er, feature.

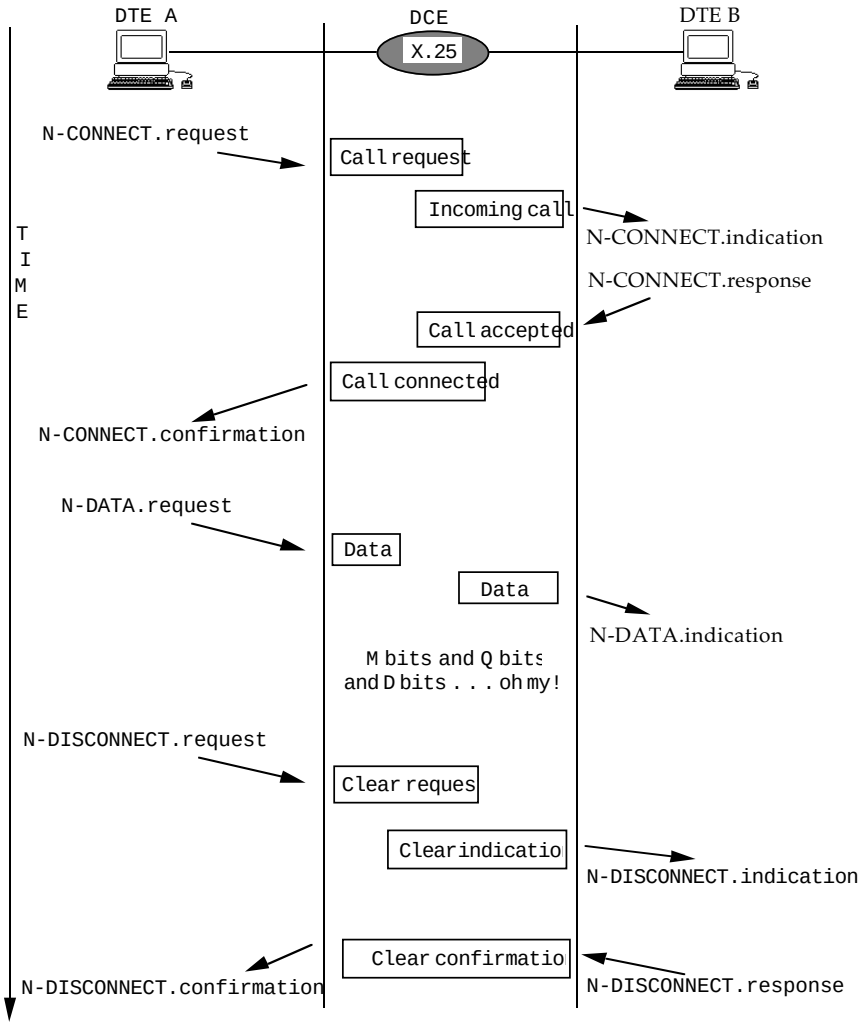


FIGURE 13.4 X.25 PLP Operation

subnetwork access protocol operating beneath an internetworking protocol (i.e., an SNICP; see Figure 13.2), and each X.25 virtual circuit serves as a simple bit pipe. OSI network datagrams (CLNP packets) are mapped onto these subnetwork connections and transferred between communicating CLNP network entities. A minimal subnetwork service is expected from X.25 networks operating in the subnetwork role; essentially, virtual circuits must be capable of handling a maximum service data unit size of no less than 512 octets. Many of the reliability features of the X.25 packet-level protocol remain useful across this single internetwork "hop"

(although the authors know of no implementations that make use of receipt confirmation when X.25 plays this role), but unnecessary. When X.25 virtual circuits are used to provide a subnetwork service to CLNP, subnetwork-dependent convergence functions specific to X.25 are used to set up, transfer data over, and tear down "subnetwork connections" between communicating CLNP network entities. The subnetwork connection-management functions for X.25 are described in ISO/IEC 8473 (the standard for CLNP), and the description is generic enough to apply to a family of switched, connection-oriented network services, including packet-mode Integrated Services Digital Networks (ISDN), Frame Relay, and new "cell-relay" services (see Chapter 15).

Subnetwork connection management begins as a relatively simple process. To send a CNLP packet across an X.25 public data network, for example, an X.25 virtual circuit must be established. If the CNLP packet is the first to be sent to a particular destination reachable via the X.25 public data network, the X.25 call-setup procedures must be initiated (connection establishment) and a virtual circuit must be established. Subsequent CLNP packets can be sent over the same virtual circuit.[11] The process is relatively simple if there is no cause to tear down the X.25 virtual circuit; if, however, there is a cost associated with X.25 call duration (for example, one is charged for how long one remains on a telephone regardless of whether one talks constantly or not), it may be more economical to tear down the virtual circuit when there are no longer any data to send (trading the cost of call duration for the cost of call setup the next time a packet must be sent to that destination). Determining *when* to tear down the X.25 virtual circuit is tricky; like telephones, the virtual circuit is full-duplex, and one party may think the conversation has ended while the party at the other end still has something to say. Timers are recommended to impose a maximum idle time on a virtual circuit; i.e., the expiration of a timer indicates that neither party has said anything for a while, so it's presumably safe to clear the virtual circuit. But timers alone do not entirely solve the problem of synchronization, which may require additional protocol interaction on behalf of the parties sharing the virtual circuit (i.e., one computer would ask permission to tear down the call, and would only do so if granted by its peer).

TCP/IP Use of X.25

In the Internet architecture, X.25 is not considered to be a protocol of the

11. Can be, but need not be; the standards permit considerable flexibility in the way in which virtual circuits are managed, so as to accommodate the wide variety of tariffs that govern the use of public networks.

network layer. X.25 is one of many network interfaces on top of which IP is run. Both DTE/DTE and DTE/DCE modes of operation are supported (Defense Communications Agency 1983; RFC 877). A more recent Internet RFC describes multiprotocol interconnection over X.25 and ISDN in the packet mode (RFC 1356). It doesn't change X.25's role in TCP/IP; rather, it corrects some of the errors and ambiguities in the RFC 877 text and aligns it with ISO and CCITT standards that have been written since RFC 877 was published. One important change to the IP encapsulation is that RFC 1356 recommends an increase in the allowed IP datagram maximum transmission unit from 576 to 1,600 octets, to facilitate local area network interconnection.

Connectionless Network Service

Both OSI and TCP/IP support a connectionless network service: OSI as an alternative to network connections and TCP/IP as the only game in town (TCP/IP networking can use connections to transfer IP datagrams, but IP offers only a connectionless service to its users). The OSI standards attach great importance to the way in which transport connections are bound to network connections when the OSI connection-oriented network service is used, because in OSI, the connection-oriented network service represents, in fact, an attempt to perform some of the functions of the transport layer in the network layer. When the connectionless network service of OSI or TCP/IP is used, the coupling of connections is not an issue, and neither service attempts to pass any of the connection-oriented features that may be present in underlying subnetworks through to the transport layer.

OSI's connectionless network service (CLNS) (ISO/IEC 8348: 1993) is a best-effort-delivery service. Like a letter one submits to the postal service, each network service data unit submitted to the OSI connectionless network service contains all the addressing and service quality information necessary to forward the packet from its source to its destination, over potentially many intermediate "hops" along the way (see Figure 13.5).

Datagram Service in OSI

The details of the OSI connectionless network service can be found in ISO/IEC 8348. A "datagram" primitive (N-UNITDATA) is used to describe the process of submitting user data to and receiving user data from the connectionless network service provider (see Table 13.3). The service definition is quite simple; after all, how much can one say about a datagram?

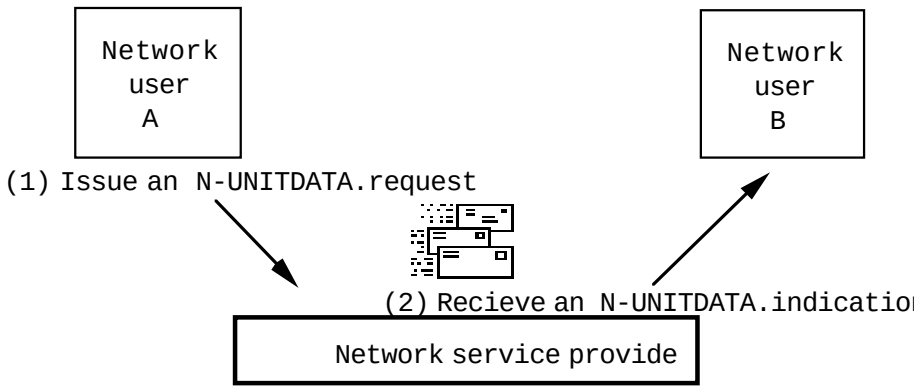


FIGURE 13.5 Connectionless Network Service

TABLE 13.3 The OSI Connectionless Network Service Primitives

Parameter	*N-UNITDATA Request*	*N-UNITDATA Indication*
Source address	X	X (=)
Destination address	X	X (=)
Quality of service	X	X
NS-USERDATA	X	X (=)

Actually, a lot. ISO/IEC 8348 is embellished by a great deal of explanatory material, since connectionless networking enjoyed a heretic's notoriety ten years ago that is difficult to imagine today. One of the more amusing parts of the standard describes the "queue model" for a connectionless service, which portrays the connectionless network service provider as nothing less than a reprehensible scoundrel who may "discard objects, duplicate objects . . . change the order of objects in the queue," as if such actions were performed deliberately as part of a malicious effort to subvert the true and proper goal of networking—which is of course embodied to its fullest extent only in the connection-oriented network service . . .

Where are the OSI network-layer protocols in a representative UNIX implementation? Following the model for TCP/IP, the ARGO 1.0/RENO network services, connection-oriented and connectionless, are accessed via procedure calls within the kernel (for debugging purposes, the "raw" socket may be used to access the connectionless service). To support two services, different transport layer to lower layer interfaces are provided. The CLNP and X.25 protocol-control blocks coexist under one transport layer, and the transport protocol-control block is separated from the network-layer protocol-control blocks to allow both CLNP and OSI transport protocols to interface to X.25 in the same way (see Figure 13.6).

Datagram Service in TCP/IP

RFC 793 describes the underlying service that TCP expects to receive from the internet layer (who says we have no architecture in the Internet?) In keeping with the TCP/IP design principle of end-to-end reliability provided by an end-to-end transport protocol, TCP's expectations are minimal: data transfer with nonzero probability of arrival, during which

- Data may be lost.
- Data may arrive in an order different from the order in which they were sent.
- The data that are received may not be precisely the same as the data that were sent.
- Data may be delivered to the wrong destination.

TCP expects the underlying service[12] to select a route and forward

12. In most implementations of TCP, the underlying service is provided by IP, but RFC 793 does not require that this be the case; in principle, any protocol that provides the same essential service as IP could be operated under TCP. This is the basic premise for the work that is being done under the name of "TUBA" (TCP and UDP with bigger addresses) in the Internet Engineering Task Force, which substitutes OSI's CLNP for IP as TCP's "underlying

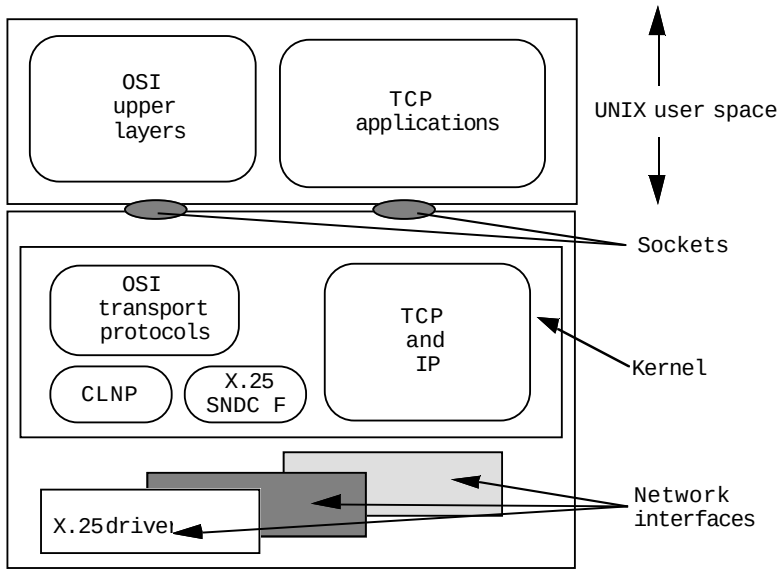


Figure 13.6 OSI Network Layer in a Representative UNIX Implementation

a packet based on the globally unique address it provides. The underlying service must also be capable of transferring maximum TCP segment size of at least 536 octets (see RFC 879).

RFC 793 also formally describes the send and receive operations. For send (NET_SEND), TCP submits the following parameters to the underlying service:

- Source and destination addresses
- Protocol identifier (identifying the service user as TCP)
- Type-of-service indicators—precedence, reliability, delay, throughput
- A "don't fragment" indicator
- A recommended "time to live" value
- The length of the data being sent, in octets
- A data identifier that distinguishes this unit of data from others sent by the same upper-layer protocol (TCP user)
- Options—a selection from the menu of options offered by IP or an equivalent underlying service
- Data

service" but keeps all the other TCP/IP protocols. A description of TUBA may be found in RFC 1347 and in Ford (1993), which is part of a special issue of IEEE *Network* magazine devoted to protocols that are proposed as successors to IP in the Internet.

For a receive operation (NET_DELIVER), TCP receives the following parameters from the underlying service:

- Source and destination addresses
- Protocol identifier
- Type-of-service indicators—precedence, reliability, delay, throughput
- The length of the data received in octets
- Options selected by the sender
- Data

In a TCP/IP implementation, only one transport–to–lower-layer interface is required (see Figure 13.6).

Internetworking Protocols

OSI's CLNP (ISO/IEC 8473: 1993) is functionally identical to the Internet's IP (RFC 791), so the two internetworking protocols—subnetwork-independent convergence protocols in *Internal Organization of the Network Layer* jargon—can be discussed in parallel. Both CLNP and IP are best-effort-delivery network protocols. Bit niggling aside, they are virtually identical. The major difference between the two is that CLNP accommodates variable-length addresses, whereas IP supports fixed, 32-bit addresses. Table 13.4 compares the functions of CLNP to those of IP. Figures 13.7 and 13.8 illustrate the header formats of CLNP and IP, respectively.

The functions performed by the two protocols are also closely related as indicated in the following subsection (see also Postel, Sunshine, and Cohen [1981] and Piscitello and Chapin [1984]).

Header Composition Function This function[13] interprets the Ns-UNITDATA.request parameters and constructs the corresponding CLNP data unit. In IP, this amounts to IP's processing the upper-layer protocol's send (NET-SEND) parameters and constructing the corresponding datagram.

Header Decomposition Function This function interprets the header

13. CLNP has a silly protocol function called "header format analysis," which amounts to examining the first octet of the protocol (the network-layer protocol identifier, or NLPID) to determine whether CLNP or the brain-damaged "inactive network layer protocol" is present. The inactive network layer protocol is an abomination; a single octet of 0 may be encoded in the NLPID field to denote "There's no network layer protocol present."

TABLE 13.4 Comparison of CLNP to IP

Function	*ISO CLNP*	*IP*
Version identification	1 octet	4 bits
Header length	1 octet, represented in octets	4 bits, represented in 32-bit words
Quality of service	QOS maintenance option	Type of service
Segment/fragment length	16 bits, in octets	16 bits, in octets
Total length	16 bits, in octets	Not present
Data unit identification	16 bits	16 bits
Flags	Don't segment, more segments, suppress error reports	Don't fragment, more fragments
Segment/fragment offset	16 bits, represented in octets (value always a multiple of 8)	13 bits, represented in units of 8 octets
Lifetime, time to live	1 octet, represented in 500-millisecond units	1 octet, represented in 1-second units
Higher-layer protocol	Not present	Protocol identifier
Lifetime control	500-millisecond units	1-second units
Addressing	Variable-length	32-bit fixed
Options	Security	Security
	Priority	Precedence bits in TOS
	Complete source routing	Strict source route
	Partial source routing	Loose source route
	Record route	Record route
	Padding	Padding
	Not present	Timestamp
	reason for discard (ER PDU only)	

information of the received datagram and creates the corresponding NS-UNITDATA.indication. For IP, this amounts to extracting the receive (NET-DELIVER) parameters to be passed to the upper-layer protocol along with the data.

Lifetime-Control Function This function limits the amount of time a datagram may remain in the network. The originator of the CLNP or IP packet determines how long it should take the datagram to reach its destination and places this value in the datagram header (using 500-millisecond units for CLNP, 1-second units for IP). This value is decremented by each of the intermediate systems/gateways that subsequently process the datagram. The datagram is discarded if the *lifetime* field (*time to live* in IP) reaches a value of 0 before the datagram is delivered to the destination.

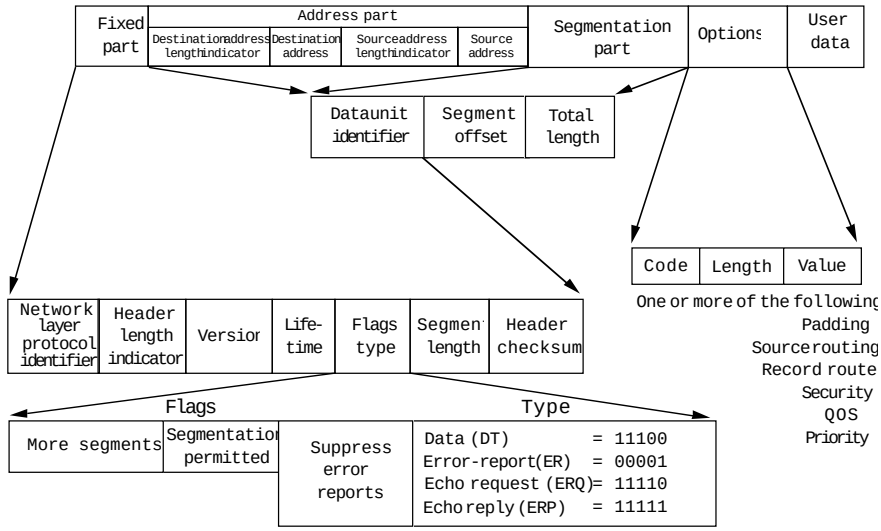


FIGURE 13.7 CLNP Data Unit Format

The CLNP lifetime-control function is a combination of hop count and time. The amount by which the value is to be decremented is the sum (in milliseconds) of the estimated or measured transit delay in the subnetwork from which the CLNP packet was received and the delay within the intermediate system that processed the CLNP packet. The lifetime field must be decremented by at least 1 by each intermediate sys-

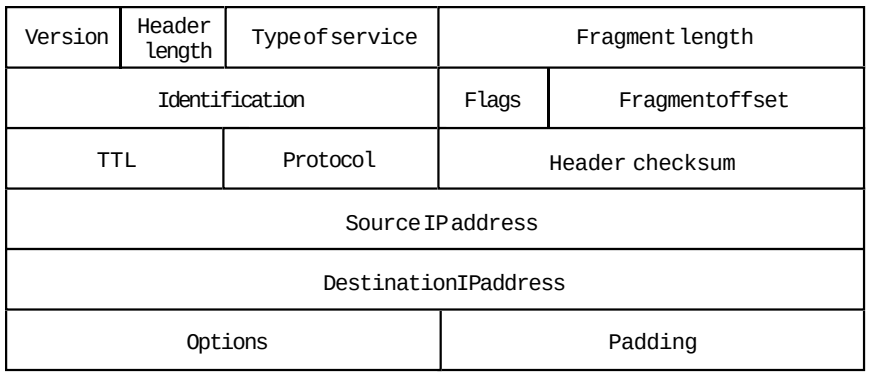


FIGURE 13.8 IP Datagram Format

tem.[14] IP's time to live (TTL) operates on a time basis; the upper-layer protocol provides a maximum datagram lifetime in 1-second units among the send parameters submitted to the datagram service.

Route PDU and Forward PDU Functions The *route PDU* function determines the network entity to which the datagram must be forwarded and the underlying service (i.e., the link) over which the datagram must be sent to reach this "next hop." The *forward PDU* function submits the datagram to the underlying service selected by the route PDU function for transmission to the next hop (it is here that subnetwork-dependent convergence functions might be invoked—for example, to establish an X.25 virtual circuit to transmit the CLNP packet).

Header Error-Detection Function This function protects intermediate systems from undetected (bit) errors in the protocol-control information of each datagram. An example of such an error is the misdelivery of a datagram as a result of the corruption of the destination address field. A 16-bit arithmetic *checksum* based on Fletcher (1982) is employed in the CLNP; a similar mechanism protects the IP header from transmission errors.

The rules for processing the CLNP header checksum are as follows:

- *Generate* the checksum only once, when the initial packet is created.
- *Check* it at each intermediate system, using a separate algorithm from the one used for checksum generation.
- Do not recompute the checksum, but *adjust* it (again, using a separate algorithm) when modifying the header. (Fields of the CLNP header may be modified by an intermediate system when it performs lifetime control and segmentation and when it performs certain optional functions, such as recording of route or setting the congestion-experienced bit in the QOS maintenance option.)

The rules for processing the IP header checksum are the same as those for CLNP: generate the checksum when the IP segment is created and check it at each router. In IP, the checksum is either adjusted without full recomputation (as for CLNP) if only the time to live field in the header is changed, or completely recomputed if any other header fields are changed.

Segmentation/Fragmentation Functions The terms *segmentation* and *fragmentation* both refer to the process whereby an IP or a CLNP packet of

14. Appendix B of ISO/IEC 8473 provides reasonable implementation guidelines for determining datagram lifetime and reassembly lifetime control.

size *N* is broken up into smaller pieces if and when it becomes necessary to transmit the packet over a subnetwork for which the maximum packet size is less than *N*. The OSI standards prefer *segmentation* and the TCP/IP standards *fragmentation*, but in most of what follows, the two terms (and other variations of the roots *segment* and *fragment*) are used interchangeably.

Segmentation is the process of composing two or more new *derived* CLNP packets—segments—from an initial CLNP packet.[15] Segmentation can be performed only if the segmentation-permitted flag is set to 1; otherwise, an intermediate system that receives a CLNP packet requiring segmentation must discard that packet.

CLNP segments are identified as being from the same initial packet when they have the same source and destination address pair and the same *data unit identifier*. The value of the *total length* field in all segments of a given initial packet remains the same as the value originally specified in the initial packet. This value may be used by a system to allocate buffer space for the entire CLNP packet regardless of which segment (the first, any of the middle ones, or the last) is received first. The *segment offset* of each CLNP segment is set to the octet at which the segment begins with respect to the beginning of the initial packet (see Figure 13.9). The *more segments flag* is set to 0 if the final octet of the initial packet is contained in this segment; otherwise, it is set to 1.

All header information from an initial CLNP packet *except* the values of the segment length, segment offset, and checksum fields is copied into the header of all segments derived from that packet by fragmentation. The value of the checksum field must be adjusted to reflect the changes in the values of the segmentation part of the CLNP header, lifetime field reductions, and changes to options (if any).

The IP *fragmentation* process is nearly identical to CLNP segmentation. The protocol header of a fragment is copied or derived from the original IP datagram, including its options. IP fragmentation can be performed only if the *don't fragment flag* is set to *may fragment* (0); otherwise, a gateway that receives an IP datagram requiring fragmentation must discard it. Fragments are identified as being from the same IP datagram when they have the same values for the following fields: datagram iden-

15. The "initial" CLNP packet is the one constructed by the original sender of the packet in the source end system. When an incoming packet is broken up into segments for transmission over the "next-hop" subnetwork, each segment is transmitted as a CLNP packet, which becomes the "incoming" packet at the next-hop system on the path—where it may have to be fragmented again. An "initial" packet, therefore, really means either the initial packet emitted by the source or any incoming packet (datagram or datagram segment) that is "initial" with respect to the operation of CLNP in the system to which it is "incoming."

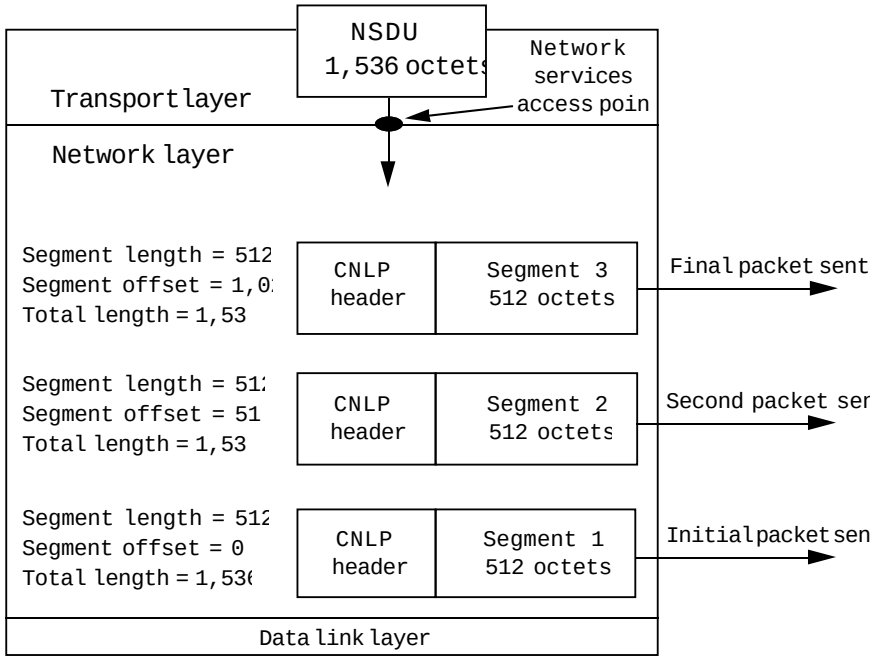


FIGURE 13.9 CLNP Segmentation (Conceptual)

tification, source and destination addresses, security, and protocol. The value of the *fragment offset* field is set to the octet at which the segment begins with respect to the beginning of the initial IP datagram. The value of the *total length* field is set to the length of the IP fragment. (The IP header does not include a field that gives the total length of the initial packet.) The more fragments flag is set to *last fragment (0)* if the final octet of the IP datagram is contained in this fragment or to *more fragments (1)* if fragments containing additional octets from the initial packet follow those contained in this fragment.

IP fragmentation differs from CLNP segmentation in the following ways:

- Data must be fragmented on 8-octet boundaries.
- The value of the fragment offset represents the position of this fragment relative to the beginning of data in 8-octet groups.
- The total length of the initial IP datagram is not encoded in the IP header.

Fragmentation and reassembly have a direct bearing on overall network performance. Experience demonstrates that it is nearly always bet-

ter for the source end system to compose and send one large datgram rather than lots of little "micrograms," but a poorly implemented fragmentation strategy can cause serious problems if that large initial packet must later be fragmented to cross a subnetwork that cannot swallow it whole (Kent and Mogul 1987). CLNP implementers can benefit by studying and extracting the directly relevant information recorded during the nearly 20 years of practical experience in implementing IP and observing its behavior. For example, Clark discusses IP datagram reassembly algorithms in RFC 815. Clark also provides a marvelous explanation of how transport and internetwork protocols can be efficiently implemented in RFC 817. And the host and gateway requirements documents RFC 1122 and RFC 1009 provide Internet guidelines for the implementation of fragmentation and reassembly procedures.

Discard and Error-reporting Functions As a last resort, both CLNP and IP discard datagrams when things go bad; e.g., when

- A protocol error is detected.
- A checksum computation fails to yield the value indicated in the checksum field of the datagram header.
- Buffers are not available to store the datagram.
- A datagram arrives that must be segmented (fragmented) before it may be forwarded, but the flag settings don't allow segmentation (fragmentation).
- Conditions for processing options cannot be satisfied.
- The lifetime of a datagram expires.
- The time allotted for reassembly of a datagram expires.

The "Internet Control Message Protocol" (ICMP; RFC 792) provides an elaborate mechanism for reporting errors in IP datagram processing at hosts and gateways.[16] Equivalent functions are provided for CLNP using a *reason for discard option* conveyed in the CLNP error report. ICMP messages are encapsulated in IP datagrams and are distinguished from other upper-layer protocols conveyed in IP datagrams by the value 1 in the *protocol* (PROTO) field, whereas CLNP error reports are distinguished from CLNP data packets by assigning a value of 1 to the *type* field of the CLNP header.

ICMP messages and CLNP error reports are both datagrams, and delivery of these messages is not guaranteed; generation of error-report messages, however, is mandatory in both protocols when certain errors

16. Error reporting is not the only function of ICMP; see "TCP/IP and OSI Control Messages" later in this chapter.

occur. The header of a discarded CLNP packet and as much of the data as can be accommodated without segmentation is returned in the CLNP error report. Similarly, every ICMP error message includes the IP header and at least the first 8 data octets of the IP datagram that caused the error. More than 8 octets may be sent, so long as the ICMP datagram length remains less than or equal to 576 octets.

The errors reported by ICMP and CLNP are compared in Table 13.5; for TCP/IP, however, it is always useful to refer to "Requirements for Internet Hosts—Communication Layers" (RFC 1122) for the precise circumstances that precipitate the generation of an ICMP error message.

The process of generating error reports and ICMP messages is virtually the same. One never generates an error message about an error occurring during the processing of a CLNP error report or an ICMP message. ICMP messages are generated only when an error is detected in the fragment whose fragment offset is set to 0 (the first chunk of the datagram); since no such restriction is explicitly stated in CLNP, one should conclude that it is appropriate to generate an error report PDU about an error occurring in any segment of a CLNP data unit.

Options Both CLNP and IP have lots of options. The sets of options defined for the two protocols are virtually identical, but the processing is slightly different (as, of course, are the bits). Table 13.6 compares CLNP and IP options, and provides a brief explanation.

The composition of CLNP error reports is affected by the options present in the discarded CLNP data packet; specifically, the CLNP error report must have the

- Same priority and QOS maintenance (if these options are supported).
- Same security. If this option is not supported, the CLNP error report should not be generated.
- Reversed complete source route. If this option is not supported, the CLNP error report should not be generated.

Padding, partial source routing and record route may be specified by the intermediate system generating the CLNP error report if these options are supported.

ICMP and IP are completely independent protocols; however, which options from the discarded IP datagram are copied into the datagram that conveys the ICMP error message datagrams remains an issue in the Internet Engineering Task Force. Currently, the IP datagram that is used to carry an ICMP message must set type of service to 0, and any ICMP message returned in response to a packet that contains an IP secu-

rity option should include a security option identical to that found in the discarded IP datagram.

TABLE 13.5 Comparison of ICMP Messages and CLNP Error Reports

Category	*CLNP Error Report*	*ICMP Message*
General	Reason not specified	Parameter problem
	Protocol procedure error	Parameter problem
	Incorrect checksum	Parameter problem
	PDU discarded—congestion	Source quench
	Header syntax error	Parameter problem
	Segmentation needed, not permitted	Fragmentation needed, but don't fragment flag is set
	Incomplete PDU received	Parameter problem
	Duplicate option	Parameter problem
Addressing-related	Destination address unreachable	Network unreachable*
	Destination address unknown	Host unreachable
Source routing	Unspecified source-routing error	Source route failed
	Unknown address in source-routing field	Source route failed
	Path not acceptable	Source route failed
Lifetime	Lifetime expired while data unit was in transit	Time to live exceeded in transit
	Lifetime expired during reassembly	Reassembly time exceeded
PDU discarded	Unsupported option, unspecified error	Parameter problem
	Unsupported protocol version	Parameter problem
	Unsupported security option	Parameter problem
	Unsupported source-routing option	Parameter problem
	Unsupported record-route option	Parameter problem
Reassembly	Reassembly interference	Reassembly time exceeded

*Gateways and hosts that generate ICMP destination unreachable messages are encouraged to "choose a response code that most closely matches the reason why the message is being generated." RFC 1122 identifies an expanded list of reason codes.

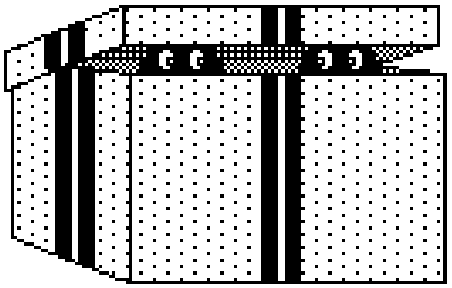
TABLE 13.6 CLNP and IP Options

CLNP Option	*Corresponding IP Option*	*Comments*
Complete source routing allows the originator of a packet to dictate the entire and only route (i.e., every router) a packet can take.	*Strict source and record routing* enables a ULP to name the IP modules that must be visited.	In both cases, all segments/fragments must follow the same route (option is copied into all fragments).
Partial source routing allows the originator to provide a partial list of routers (pointing the packet in the right direction).	Like CLNP's option, *loose source and record routing* allows a ULP to provide "hints" (the "inverse" route is recorded along the way).	CLNP options are copied into every segment; loose source and record routing is only present in fragment 0.
Partial route recording records the list of intermediate systems visited by a packet, in order of traversal.	*Record route* records the list of gateways a datagram visits.	Each router that forwards the packet records its address in the header (and updates the checksum). The size of the option is fixed by the originator, and the packet may continue even when space for recording is exhausted.
Complete route recording is a restricted variation of partial route recording.		A complete route must be recorded or else the packet is discarded and an error report is generated.
Priority indicates relative priority of this data packet with respect to other data packets.	*Precedence* denotes the importance or priority of the datagam with respect to other datagrams.	Assumes that routing information will assist in ordering packets for forwarding.
QOS maintenance provides information intended to influence routing decisions.	*Type of service* provides information intended to influence routing decisions. (see RFC 1349)	**Examples:** maintain sequence to the extent possible (follow the leader); take the route with the lowest error rate; take the path with the smallest transit delay; take the path with the least cost.
	Timestamp provides a list of timestamps (and optionally, the addresses) from the gateways traversed by the datagram.	
Padding is used to lengthen the header to a convenient length.	*Padding* is the last field of the datagram and is used to ensure that the IP header ends on a 32-bit boundary.	Padding is used to align user data to machine word sizes.
A code point for a *security* option is reserved but is as yet undefined.	Security provides a means of conveying security level, compartmentation, handling restriction codes, and user group parameters. (See RFC 1108)	In many implementations, and in the absence of OSI standards for security, RFC 1108 is encoded in the CLNP security option.

TCP/IP and OSI Control Messages

ICMP defines messages other than error reports. The *source quench* message serves as a coarse congestion-notification mechanism, providing routers with the means to tell hosts to reduce the rate at which they are sending IP packets. The same function is accomplished in OSI using a CLNP error report with the *reason for discard* field set to the value that means "congestion experienced." The ICMP *echo request* and *echo reply* messages are useful in determining whether a remote IP entity is "alive and well." A host issues an echo request and determines from the receipt of a corresponding echo reply message that the remote IP entity is able to process IP (and ICMP) packets. This is accomplished in a nearly identical manner in OSI using CLNP *echo request* and *echo reply* packets, which have been adopted as part of the second edition of ISO/IEC 8473. The ICMP *redirect* message is used by router A when it receives an IP datagram from a host "foo" and determines from its local routing information that router B offers a better (shorter) path to the destination IP address "bar." Router A will forward the original datagram toward its destination and also return a redirect message advising foo to forward future packets destined for bar to router B. (Redirection is accomplished in OSI through the use of the end-system to intermediate-system routing protocol, which is described in Chapter 14.) ICMP also offers a *timestamp* request and reply function, which is used to determine round-trip-times across an internet, and an information request and reply function, which may be used by a host to determine its IP network number. (Currently, OSI has no corresponding capabilities defined for CLNP.)

OSI Protocol Combinations



IONL was Pandora's box . . .
ISO/IEC 8880 explains what to do once you've opened it!

The *Internal Organization of the Network Layer* describes the relationships among the real-world components that might be used (individually or collectively) to provide the OSI network service. In a rash moment of pragmatism, it was decided that far too many combinations of real-

world protocols existed and a weeding-out process was needed. ISO/IEC 8880: 1990, *Protocol Combinations to Provide and Support the OSI network Service, Part 1—General Principles*, describes the weeding-out process; Parts 2 and 3 describe the combinations that survived the initial cut. To avoid the embarrassing accusation of lacking vision, Part 1 also includes "*Criteria for Expansion of ISO 8880*."[17]

Parts 2 and 3 describe *protocol combinations* (e.g., CLNP over ISO/IEC 8802 LANs, or X.25 packet level protocol over the data link service provided by ISO/IEC 7776), *environments* (LAN, PSDN, CSDN, point-to-point subnetwork), and the obligatory *conformance* for a given environment. Part 2 is devoted to provision of the connection-oriented network service, Part 3 to the connectionless network service. (In keeping with accepted practice in OSI, the issue of interworking between the two types of network service is finessed.)

Inclusion in the protocol combinations and environments was something of a popularity contest. The combinations considered broadly applicable are included. Since there is a clause covering expansion, it is inevitable that additional combinations will be considered and incorporated in the future. The aspect of ISO/IEC 8880 that is both amusing and disappointing is that Parts 2 and 3 are merely *pointer documents*: they identify the ISO/IEC standards included in a given protocol combination/environment but provide little in the way of implementation guidelines beyond that which is contained in the cited documents. The illustrations of the protocols and combinations to support the OSI network service in Parts 2 and 3 are the essence of the standard (Figures 13.10 and 13.11).

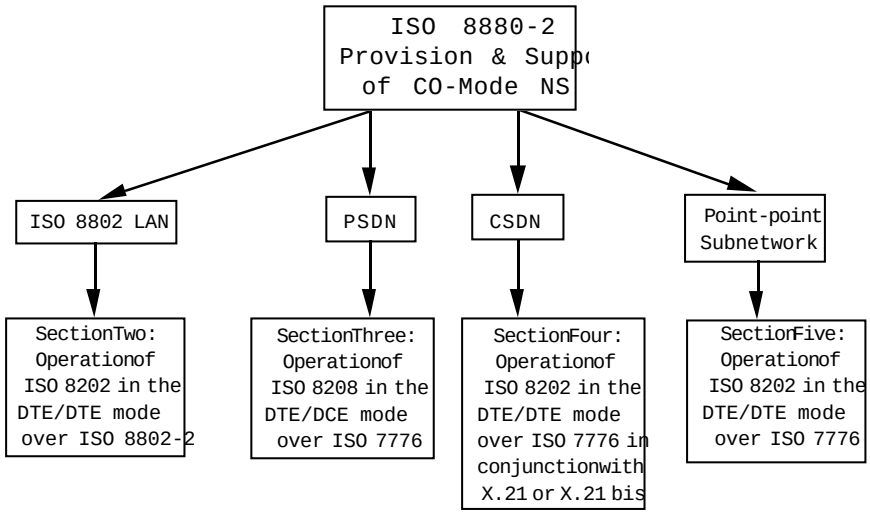
TCP/IP Protocol Combinations

The TCP/IP equivalent of OSI's protocol combinations is a series of much more practical documents that describe the mundane but essential details of IP encapsulation and other idiosyncratic behavior associated with IP protocol operation over the many network interfaces (IEEE 802 LANs, Ethernet, SMDS, FDDI, amateur packet-radio link-layer protocol, point-to-point links—see RFC 1042, RFC 894, RFC 1209, RFC 1188, RFC 1226, and RFC 1171, respectively).

Network-layer Protocol Identification

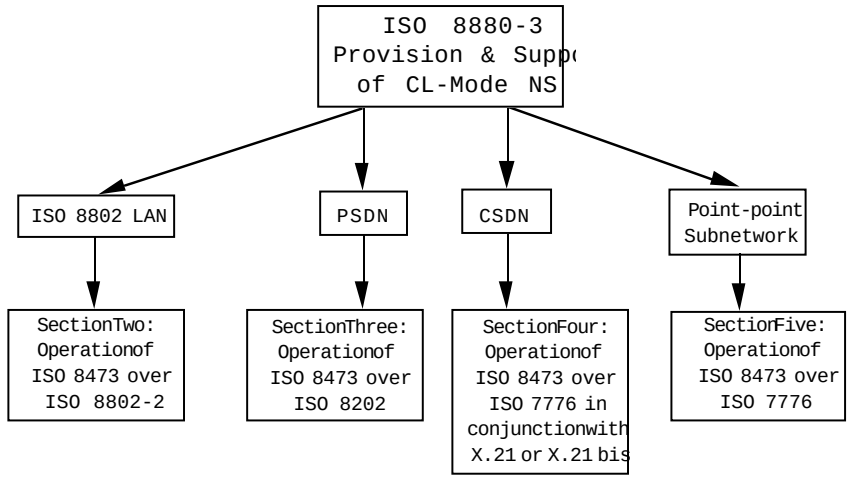
Given that there are potentially lots of protocols in the OSI network layer, and they are sometimes stacked together, how do you distinguish

17. Rather than expanding ISO/IEC 8880, the standards developers decided in early 1993 to replace all 3 parts with an omnibus technical report entitled "Provision of the OSI Network Service" (ISO/IEC 13532: 1993).



Source: ISO/IEC 8880 (1990), "Protocol Combinations to Provide and Support the OSI Network Service, Part 2—Provision and Support of Connection-mode Network Service."

FIGURE 13.10 Provision and Support of Connection-oriented Network Service



Source: ISO/IEC 8880 (1990), "Protocol Combinations to Provide and Support the OSI Network Service, Part 3—Provision and Support of Connectionless Network Service."

FIGURE 13.11 Provision and Support of Connectionless Network Service

one from the other? By convention (ISO/IEC 9577: 1990), all OSI network-layer protocols must reserve the first octet as the *initial protocol identifier* (IPI); the first octet of CLNP, for example, is binary 1000 0001. Thus, the first octet of the data link service data unit identifies which of the OSI network-layer protocols is conveyed (Figure 13.12).

It should come as no surprise that the X.25 packet level protocol treatment of this octet is an exception. The first octet of X.25 packets is a combination of the *general format indicator* (GFI) and the upper 4 bits of logical channel identification—so X.25 hogs dozens of network-layer protocol identifier bit assignments (Figure 13.13).

In OSI, X.25 is a beast of many burdens; it can carry CLNP, OSI routing protocols for CLNP, triple-X protocols, or OSI transport protocols. This multiple role for X.25 necessitates the use of a *subsequent protocol identifier* (SPI) to indicate "what's in the X.25 data packets." The subsequent protocol identifier is the first octet of the call user data field in the X.25 *call request* packet; if the call user data in the X.25 PLP call request packet is absent, then it is assumed that the OSI transport protocol is present in X.25 data packets.

(Note that in theory, one could encapsulate X.25 in CLNP or TCP or the OSI transport protocol: "Tunneling" X.25 is an increasingly common practice in multiprotocol routers.) CLNP does not yet have an option identified for a subsequent protocol identifier or IP PROTO field.

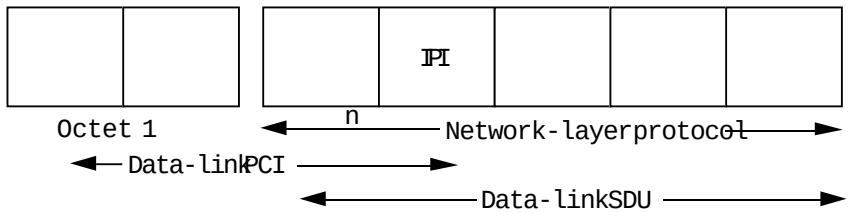


FIGURE 13.12 Initial Protocol Identifier

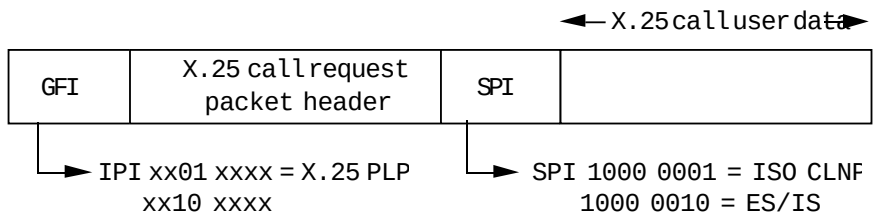


FIGURE 13.13 Subsequent Protocol Identifier

Network-Layer Protocol Identification in TCP/IP and Multiprotocol Environments

With some cooperation from ISO/IEC, IP protocol identification has recently been expanded to accommodate coexistence with OSI and other network-layer protocols on X.25 and ISDN networks. This work obsoletes RFC 877. Network-layer identification for multiprotocol internets combines the scheme described in ISO/IEC TR 9577 and the IEEE subnetwork access protocol (SNAP) technique.

For multiprotocol internets, especially dual-stack internets, some of the relevant values of the initial protocol identifier and subsequent protocol identifier are :

- For IP, binary 1100 1100 (hex CC, Internet decimal 204)
- For CLNP, binary 1000 0001 (hex 81, Internet decimal 129)
- For the *End System to Intermediate System Routeing Exchange Protocol for Use in Conjunction with ISO 8473* (ISO/IEC 9542: 1988; see Chapter 14), binary 1000 0010 (hex 82, Internet decimal 130)
- For the *Intermediate System to Intermediate System Routeing Protocol for Use in Conjunction with ISO 8473* (ISO 10589: 1992; see Chapter 14), binary 1000 0011 (hex 83, Internet decimal 131)

The binary value 1000 0000 (hex 80, Internet decimal 128) identifies the use of the IEEE 802 subnetwork access protocol to encapsulate and identify a non-OSI network-layer protocol in X.25 (see Figure 13.14). The

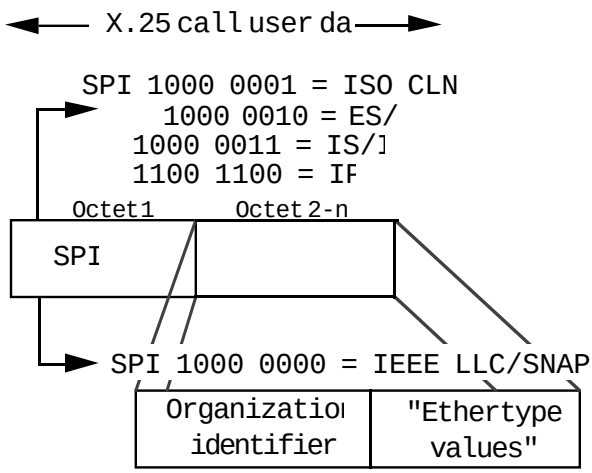


FIGURE 13.14 Relevant IPI/SPI Values for Multiprotocol Internets

IEEE 802 SNAP is a five-octet header consisting of an *organization identifier*, which is set to the value of the Internet Ethernet address block number (see RFC 1340, *Assigned Numbers*), and the *ethertype value* that identifies the encapsulated protocol (e.g., IP, ARP).

Network Layer Addresses

Network-layer addresses identify the hosts (end systems) attached to TCP/IP and OSI networks. IP and OSI network service access point (NSAP) addresses share several common attributes:

- They are globally unique—address administrations and policies exist to ensure that no two host machines use the same network address.
- They accommodate (and prescribe) hierarchies, which enable groups of hosts to be associated with a specific domain (and possibly subdomains within that domain).
- They convey information that is used to determine routes between hosts, across potentially many (sub)networks.

Network addressing in the Internet accommodates a single network service; network addressing in OSI must accommodate not only multiple network services but a very large number of public numbering and private addressing plans. In the following subsections, readers should consider how Internet and OSI network addressing are influenced by differences in scope and in the expectations that have been imposed on them.

Internet (IP) Addresses

The Internet network address is more commonly called the "IP address." It consists of 32 bits, some of which are allocated to a high-order *network-number* part and the remainder of which are allocated to a low-order *host-number* part. The distribution of bits—how many form the network number, and how many are therefore left for the host number—can be done in one of three different ways, giving three different *classes* of IP address (see Figure 13.15).[18]

The network number identifies a real subnetwork, and the host number identifies a physical interface to that subnetwork. Whereas the OSI network-layer address identifies the abstract service access point

18. In fact, there are two other IP address classes which are not discussed here; class-D addresses, which begin with the bit pattern 1110, are reserved for IP multicasting, and class-E addresses, which begin with the bit pattern 1111, are reserved for experimental use.

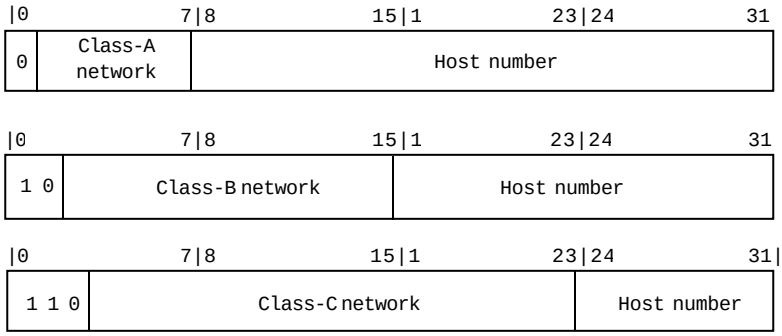


FIGURE 13.15 Class-A, Class-B, and Class-C IP Addresses

between the transport and network layers, the Internet address identifies the actual point of attachment of a computer system to a real subnetwork (the "network interface"). In OSI, this would be referred to as a "subnetwork point of attachment address," which is not at all the same thing as an NSAP address (see "OSI Network Layer Addressing," later in the chapter).

IP Subnetting Internet addressing inherently accommodates a two-level hierarchy (network and host), and all of the bits in the address are useful for routing. Additional levels of hierarchy can be created within a single class-A, class-B, or class-C network by applying the notion of *subnetting*, which is accomplished by borrowing bits from the host portion of an Internet address. (The Internet Standard subnetting procedure is defined in RFC 950.) Subnetting is visible only within the physical network in which it is applied; hosts and routers outside the physical network do not (in fact, must not) know that the host portion is partitioned into subfields. Routers and hosts within a subnetted network distinguish a *subnet number* from a host number by applying a *bit mask* (see Figure 13.16).

Applying Subnetting to Accommodate Growth Given the original scope of the ARPANET, it is easy to forgive the developers of IP for having failed to anticpate the success of the Internet. Once a single network with 4 hosts, the Internet now consists of well over 10,000 individual networks with more than a million hosts and is approximately doubling in size every year. The 32-bit IP address space cannot withstand such a growth rate for much longer. By March 1993, nearly 32 percent of the available IP network numbers had been assigned, including:

- Aproximately 39 percent of the 128 class-A network numbers,

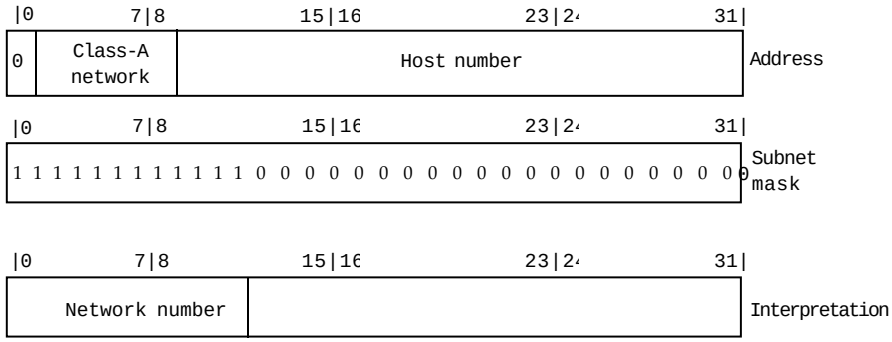


FIGURE 13.16 Subnetted Internet Address

which represent half of the total address space. (Sixty-four of the class-A numbers have been reserved by the Internet Assigned Numbers Authority and are therefore "unassigned"; the number of unreserved class-A network numbers that are available for assignment is therefore much smaller than the 39 percent figure would suggest.)

- Just over half of the 16,384 class-B network numbers, which represent one-fourth of the total address space. (Class-B addresses are by far the most popular, as they provide for a large number of hosts per network, with considerable room for subnetting within the 16-bit host-number space.)
- Approximately 6 percent of the 2,097,152 class-C network numbers, which represent one-eighth of the total address space.
- All of the class-D network numbers, which are reserved as "multicast" addresses, representing one-sixteenth of the total address space.
- All of the class-E network numbers, which are reserved by the Internet Assigned Numbers Authority, representing one thirty-second of the total address space.

The net result is that a significant fraction—a bit more than 42 percent—of the network-number space had already been either allocated (classes A, B, and C) or reserved (classes D and E) by March 1993 (not counting the 654 reserved class-A network numbers, which could in principle be released by the Internet Assigned Numbers Authority):

$$(1/2)*.39 + (1/4)*.51 + (1/8)*.06 + (1/16) + (1/32) = .42375$$

Clearly, we cannot continue to assign new IP network numbers at an annual growth rate of 100 percent for much longer! Fortunately, there appears to be at least a short-term solution to the problem, based on the observation that the depletion of the IP network-number space has been artificially accelerated by a mismatch between the number of hosts per network allowed by the fixed class-B and class-C partitions (64K and 8K, respectively) and the number of hosts that are typically attached to real-world networks (something in between, but often much smaller than 64K). One way to add years to the life of the existing IP address space is to extend the practice of IP subnetting and take advantage of the large number of unassigned class-C network numbers. An RFC in progress recommends that the rigidly fixed octet boundaries that define the three classes of IP address be broken (ignored) and that more flexible bit-level masks be used to distinguish the network number from the host number. This "classless" interdomain routing (CIDR) strategy would permit blocks of contiguous class-C network numbers to be assigned to sites that would ordinarily need a class-B network number (to accommodate more than 256 hosts). Although this strategy clearly does not increase the total supply of network numbers, it encourages more efficient (denser) use of the host-number space associated with each network number (or set of contiguous network numbers), thereby slowing the rate at which new network numbers must be assigned.

By itself, the assignment of a block of class-C network numbers rather than a single class-B network number creates a problem that is potentially much worse than the problem it attempts to solve: if routers must advertise each of the class-C network numbers individually, the "not enough addresses" problem has been solved at the expense of creating a new "too many routes" problem. Internet routers cannot efficiently maintain the very large routing tables that would result from the advertisement of many tens of thousands of networks. The classless interdomain routing scheme therefore requires the development and deployment of interdomain routing protocols that summarize ("aggregate") routing advertisements, so that a single routing table entry can be maintained for a set of networks that share a common IP network-number prefix and routing metrics (see Chapter 14). Rather than maintaining an entry for each network number (as would be the case with current interdomain protocols), sites with multiple contiguous class-C network numbers would be routed in the interdomain system using a single routing table entry.

It is in fact necessary for this aggregation to occur recursively at the transit routing domain boundaries (for example, the boundaries between

the National Science Foundation backbone NSFnet and the NSFnet regionals; between NSFnet and the European backbone network EBONE; or between the Nordic regional network NORDUnet and the commercial network Alternet) in order to arrest the growth of routing tables in transit domain routers. This requires greater coordination in the assignment of network numbers: sites that are topologically "close" with respect to transit domain routing must be allocated network-number blocks with adjacent prefixes. Making intelligent choices when assigning network numbers in this way is likely to be possible (if it is possible at all) only if the assignment authority is delegated to the organizations that are responsible for coordinating international and interorganizational routing. Classless interdomain routing may not be effective if network-number assignment remains centralized.

Classless interdomain routing will require Internet network operators to use one of the two interdomain routing protocols that currently support variable-length subnet masks (see Chapter 14, "Interdomain Routing") in place of the popular "exterior gateway protocols" (see Chapter 14, "Interdomain Routing in TCP/IP"). It will also be the case that networks that do not default external routes to a major transit network will be unable to continue using an intradomain routing protocol that does not support variable-length subnet masks, because incoming route advertisements representing aggregated sets of network addresses would have to be expanded, resulting in what would eventually be unsupportable increases in the amount of router memory required to store all of the routes individually.

The expected "time to impact" for classless interdomain routing is two to three years, given the time it takes to implement and deploy a new routing protocol and the network-operation adjustments that will be required. Routing experts do not agree on the question of whether classless interdomain routing buys enough time to wait for a completely new Internet routing and addressing architecture or only breathing room for the deployment of a near-term functionally equivalent replacement for IP. However, there is general agreement that classless interdomain routing is worth doing in either case, and the Internet Engineering Steering Group has begun the process of approving an Internet standard that will mandate the use of classless interdomain routing in the Internet.

A Real-World Example of IP Address Internet address data structures may be represented in a UNIX environment in the following manner:

```
typedef char in_addr[4] /* 4-octet IP address */
struct sockaddr {
```

```
     short      sa_family;     /* address family, e.g., internet */
     char       sa_data (14);  /* space for the address */
}
struct sockaddr_in {
     short      sin_family;    /* address family is internet */
     u_short    sin_port;      /* the internet port  */
     in_addr    sin_addr;      /* the IP address a.b.c.d */
     char       sin_zero[8];   /* unused */
}
```

OSI Network Layer Addressing

Like every networking architecture that preceded it, OSI needed addresses to identify hosts and gateways so that routes could be computed and packets could be forwarded. By the time OSI network-layer addressing was considered, however, OSI had already succumbed to a political compromise that demanded the accommodation of multiple solutions to providing the network service. By the rules of this bargain, X.25 public data networks, X.21 circuit-switched data networks, and CLNP-based enterprise networks were all equally capable of providing (and all equally entitled to provide) the OSI network service. This all but eliminated the opportunity to design a logical network-layer addressing scheme from scratch; the common carriers providing X.25 packet-switching service or X.21 circuit-switched service naturally wished to continue to use the CCITT X.121 numbering plan for public data networks or the CCITT E.163 numbering plan for the public switched telephone network, and simply call the X.121 or E.163 numbers "OSI NSAP addresses." Integrated services digital networks (ISDNs) would use the CCITT E.164 *Numbering Plan for the ISDN Era*; these numbers, too, had to be OSI NSAP addresses.[19] CLNP-based networks, on the other hand, would benefit from NSAP addresses structured like the IP addresses of the Internet, so that routing topology information could be derived from the composition of the address and efficient routes could be computed. The OSI standards community briefly considered subsuming all OSI network-layer addressing under the administration of one of the existing numbering plans, but the existence of a half-dozen eager candidates for this role put a quick end to this alternative. After protracted debate, it was finally agreed that the only way forward was to "embrace them all" in a grand compromise that gave the appropriate recognition and status to each of the contenders.

Hemrick (1985) provides an excellent historical account of the

19. Addressing plans such as these are sensitive first to administrative needs, and the routing information that can be derived from such addresses is typically limited to geographic locality.

development of the OSI network-layer addressing scheme. Briefly, the foremost issue confronting the standards bodies ended up being not how to define "bits to route with" but how to administer the use within OSI of the many pre-OSI addressing and numbering plans that went into the compromise solution. OSI network-layer addressing therefore focuses primarily on the administrative aspects of address allocation and assignment and leaves the issue of defining the "bits to route with" to the routing and interworking standards.

The OSI Network Addressing Architecture The network-layer addressing standard contained within the network service definition (ISO/IEC 8348: 1993) defines a hierarchically structured, globally unambiguous network addressing scheme for OSI. The *global network addressing domain* is composed of multiple subdomains called *network addressing domains*. An addressing *authority* exists for each network addressing domain; it is responsible for assigning addresses and assuring the uniqueness of the assigned addresses within the domain. The addressing authority also determines the rules for specifying addresses within the domain (*abstract syntax*), provides a human-readable representation of addresses that could be obtained from directories (*external reference syntax*), and describes how the addresses are to be conveyed in network-layer protocols (*encoding*).

The concept of network addressing domains is recursive; an individual network addressing authority may further subdivide its domain into subdomains, and each subdomain would again have its own authority to determine rules of address syntax and administer addresses. An example of a domain hierarchy is shown in Figure 13.17.

ISO/IEC and CCITT jointly administer the global network addressing domain. The initial hierarchical decomposition of the NSAP address is defined by ISO/IEC 8348 and is illustrated in Figure 13.18. The standard specifies the syntax and the allowable values for the high-order part of the address—the *initial domain part* (IDP), which consists of the *authority and format identifier* (AFI) and the *initial domain identifier* (IDI)—but specifically eschews constraints on or recommendations concerning the syntax or semantics of the *domain specific part* (DSP).

The value used for the authority and format identifier establishes both the format of the initial domain identifier (its syntax and the numbering plan or identifier assignment system on which it is based) and the abstract syntax (the syntax in which values are allocated by a registration authority, as opposed to the concrete syntax in which the value might actually be represented in a protocol header or in a table within an OSI

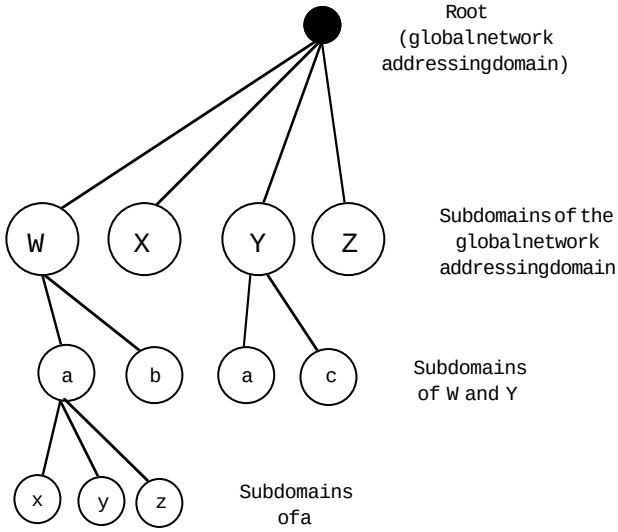


FIGURE 13.17 Hierarchies (Inverted Tree Diagram)

system) of the domain-specific part. The two-digit authority and format identifier is always encoded as a single binary-coded decimal (BCD) octet. The AFI specifies

- The addressing authority that governs the allocation of the initial domain identifier (i.e., the authority that administers this branch of the address tree), such as CCITT's X.121 public data network numbering plan.[20]
- Whether or not leading 0s in the encoding of the initial domain identifier are significant.[21]
- The abstract syntax of the domain-specific part (discussed later in this section).

20. The "authority" identified by the value of the authority and format identifier is typically a numbering plan or other coding system, defined by a specific standard or set of standards, rather than a person or agency. The authority is the standard that specifies the numbering plan or coding system, not the organization that developed the standard or the agency that administers it.

21. Some of the initial domain identifier formats are based on existing numbering plans that allocate numbers of varying length. In these cases, it is necessary to distinguish between numbers for which leading 0s are significant and those for which leading 0s are not significant, so that leading 0s can be included in or excluded from the standard encoding scheme that produces binary encodings of the decimal-digit parts of the NSAP address. In Table 13.7, where two authority and format identifier values are shown in a single table cell, the first value is used when leading 0s in the initial domain identifier field are not significant, and the second value is used when leading 0s in the initial domain identifier field are significant.

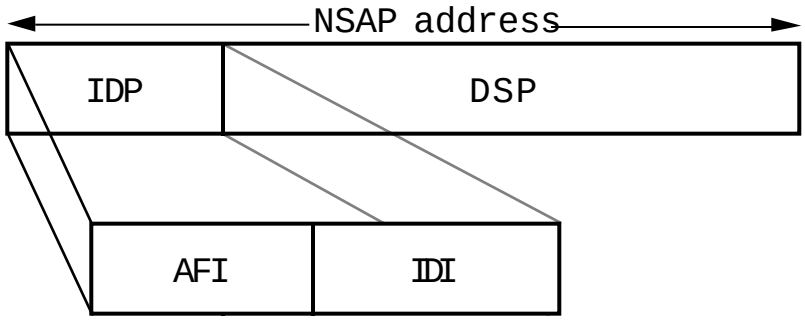


FIGURE 13.18 Structure of the NSAP Address

The available initial domain identifier formats, and the authority and format identifier values that select them, are shown in Table 13.7, which also shows the domain specific part syntax (decimal digits, binary octets, ISO 646 characters, or a nationally standardized character set) selected by each of the possible values.

The initial domain identifier identifies the addressing authority that allocates the bytes of the domain-specific part. It consists of a value selected from the numbering plan or other coding system specified by the value of the authority and format identifier; for example, if the

TABLE 13.7 AFI Values and IDI Formats

DSP Syntax / *IDI Format*	*Decimal*	*Binary*	*Character (ISO 646)*	*National Character*
X.121	36, 52	37, 53	—	—
ISO DCC	38	39	—	—
F.69	40, 54	41, 55	—	—
E.163	42, 46	43, 57	—	—
E.164	44, 58	45, 59	—	—
ISO 6523-ICD	46	47	—	—
Local	48	49	50	51

authority and format identifier value specifies the "ISO data country code (DCC)" format for the initial domain identifier (AFI value of 38 or 39), then the values of the initial domain identifier are selected from the ISO standard for data country codes (ISO/IEC 3166)—e.g., "840" (which is the ISO/IEC 3166 country code for the United States).

The composition of the domain specific part is the responsibility of the individual authority identified by the value of the initial domain identifier (in the preceding example, it would be the responsibility of the authority associated with the ISO/IEC 3166 data country code "840" [United States], which is the American National Standards Institute [ANSI]). Although some of the initial domain identifier formats use up more of the available octets in an NSAP address than others, the upper bound of 20 octets for the length of an NSAP address ensures that there is ample room in the domain-specific part for authorities to create further subdivisions (subhierarchies) if necessary (see Table 13.8).

The ANSI Standard NSAP Address Structure The American national standard for the structure and semantics of the domain-specific part of the subset of NSAP addresses for which ANSI is responsible (see the preceding section) provides an example of the way in which authorities may decide to define the domain-specific part (which is deliberately not specified by any ISO/IEC or CCITT standard). American National Standard X3.216-1992, *Structure and Semantics of the Domain Specific Part of the Network Service Access Point Address*, applies to those NSAP addresses that use the "ISO DCC" format, in which the value of the authority and format identifier is (decimal) 39 and the value of the initial domain identifier is (decimal) 840.

ISO/IEC 8348 does not require that the domain-specific part of the

Table 13.8 Maximum DSP Lengths

DSP Syntax / *IDI Format*	*Decimal (Maximum Length of DSP in Decimal Digits)*	*Binary (Maximum Length of DSP in Binary Octets)*
X.121	24	12
ISO DCC	35	17
F.69	30	15
E.163	26	13
E.164	23	11
ISO 6523-ICD	34	17
Local	38	19

NSAP address be specified by any international or national standard. The objective of ANS X3.216-1992 is to facilitate the operation of OSI routing functions, by enabling those functions to recognize a relationship between the hierarchical structure of the domain-specific part of the NSAP address and the hierarchical structure of routing domains and areas. This capability is particularly valuable when intradomain routing information is exchanged among intermediate systems (ISs) using the *Intermediate System to Intermediate System Intra-domain Routeing Information Exchange Protocol for Use in Conjunction with ISO/IEC 8473*,[22] defined in ISO/IEC 10589 (see Chapter 14).

The domain-specific part structure specified by American National Standard X3.216-1992 is illustrated in Figure 13.19. The specific way in which this format is used to facilitate routing is covered in great detail in Chapter 14. X3.216-1992 also specifies the way in which the numeric form of the "organization name" allocated and assigned by the ANSI-administered U.S. Registration Authority (see Chapter 5) is used as the "org" component of NSAP addresses constructed according to the standard.

For the purposes of OSI routing, the individual fields of the domain specific part illustrated in Figure 13.19 are interpreted as follows:

- *IDP*: The two subfields of the *initial domain part* are allocated and assigned according to ISO/IEC 8348 and are not affected in any way by X3.216-1992.
- *DFI*: the *domain specific part format identifier,* which specifies the version of the X3.216 standard. The value of this field is binary 1000 0000 (shown in Figure 13.19 as decimal 128).
- *org*: an *organization* name allocated and assigned by the ANSI-administered U.S. Registration Authority for OSI organization

IDP		DSP						
AFI	IDI	DFI						
39	840	128	org	res	rd	area	system	sel
1	2	1	3	2	2	2	6	1

FIGURE 13.19 ANSI Structure of the DSP

22. "Routeing" is in fact the internationally accepted spelling of the word when it is used in the context of computer networking (see Chapter 14).

names (see "The USA Registration Authority," in Chapter 5). The U.S. Registration Authority allocates numeric organization names as decimal numeric values. The value of this field is the binary value obtained by encoding the numeric organization name according to the "network address encoding" procedure defined in ISO/IEC 8348.

- *res*: *reserved*. The value of this field is 0.
- *rd*: the portion of the NSAP address up to and including the "rd" (routing domain) field constitutes a *routing domain prefix*, which summarizes information about some or all of the area addresses within a routing domain. This enables the border intermediate systems in a routing domain to advertise one or more prefixes, in lieu of an enumeration of the corresponding individual area addresses, for the purposes of interdomain routing information exchange (see Chapter 14).
- *area*: an identifier for the *area* within a routing domain to which the NSAP address belongs. The portion of the NSAP address up to and including the area field constitutes an *area address* in the sense in which the term is used (and defined) by ISO/IEC 10589. In the ISO/IEC 10589 context, level-1 intermediate systems report only area addresses, and level-2 intermediate systems exchange among themselves only area address information (and thus route only on the basis of area addresses).
- *system*: an identifier for the individual end system (in the case of an NSAP address) or intermediate system (in the case of a network entity title) with which the NSAP or network entity title is associated. The internal structure, value, and meaning of the *system* field are not specified or constrained by X3.216-1992.
- *sel*: the "NSAP selector," which serves to differentiate multiple NSAP addresses associated with the same network entity. The value of the *sel* field in a network entity title is 0.

A Real-World Example of OSI NSAP Addresses What in the real world are NSAP addresses? In UNIX implementations, under a socket address family "ISO," an NSAP address could be viewed as the equivalent of a TCP/IP "Internet address." As described in Chapter 12, the C data structure *sockaddr*, OSI-style, looks like this (again, from ARGO 1.0 RENO):

```
struct sockaddr {
    short      sa_family;      /* address family, ISO */
    char       sa_data (14);   /* space for the address */
    }
```

```
struct sockaddr_iso {
    short      siso_family;    /* address family is iso */
    u_short    siso_tsuffix;   /* the TSAP Address  */
    iso_addr   siso_addr;      /* the NSAP Address */
    char       siso_zero[2];   /* unused */
    }
```

Many systems, especially routers, will find it necessary to accommodate multiple NSAP address formats. These might be encoded in the following manner:

```
struct iso_addr {
  u_char     isoa_afi;   /* authority/format identifier */
  union {
     struct addr_37       addr_37;       /* x.121 */
     struct addr_39       addr_ansi;     /* ISO DCC IDI *
     struct addr_47       addr_6523icd; /* ISO 6523 ICD*/
  }
  isoa_u;    u_char
isoa_len;    /* length */
```

in which "addr_ansi" from the earlier example would appear as follows:

```
struct addr_ansi {
  u_char   ansi_dfi;      /* DSP format identifier    */
  char     ansi_org (3);  /* organization code        */
  char     reservd (2);   /* not used yet             */
  char     domain (2);    /* routing domain           */
  char     area (2);      /* area within domain       */
  char     systemid (6);  /* possibly a 48-bit
                             IEEE 802 MAC address     */
  u_char   NSAPsel;       /* NSAP selector            */
};
```

OSI NSAP Addresses in the Internet With so many NSAP addressing formats to choose from, which should be used in a multiprotocol Internet supporting (at least) OSI and TCP/IP? *Guidelines for OSI NSAP Allocation in the Internet* (RFC 1237) describes the administrative requirements for obtaining and allocating NSAP addresses and for distributing the assignment authority among the Internet's backbone and midlevel networks. It also explains how to construct and assign the various parts of the NSAP address to achieve efficient OSI and dual-stack (OSI and IP) routing and how to apply NSAP addressing to achieve the OSI equivalents of existing IP topological entities—backbones, regionals, and site and campus networks—in the Internet.

Conclusion

Because of the inability of the OSI standards community to decide between a network-centric or host-centric model of global networking, the OSI network layer is really several layers trying to look like one, as if for the sake of architectural appearances. *The Internal Organization of the Network Layer* is a convenient fig leaf, but the international networking community would be much better served by a forthright recognition of the internet/subnet split within the network layer. The TCP/IP architects avoided this problem entirely—their network layer is actually called the "internet" layer, and it sits on top of clearly distinguished subnetworks ("network interfaces").

Although this is often characterized as a "failure" on the part of the OSI standards developers, it is important to realize that the host-centric/network-centric conflict was not played out in purely abstract terms. Network-centric arguments make more sense in the real-world context of national regulation, monopolistic common carriers, and the economics of public data networks and their tariffs. The host-centric argument has always been that diversity—in networks as in everything else— is an inescapable fact of life and that no amount of regulating (or standards making) can or should limit the range of choices from the menu of networking technologies that makes internetworking essential. The conflict has by no means been resolved within OSI, but few people today would bet on anything other than internetworking as the most likely model of the future global data network.

The apparent incompatibility of the connection-oriented and connectionless network services in OSI is not an interworking problem and would not be solved by the invention of a magic gateway that could convert one type of service into the other. It is a fundamental architectural problem, reflecting two very different models for interconnecting a world's worth of networked hosts. The TCP/IP community has been fortunate to avoid this issue long enough to firmly establish TCP/IP's connectionless-internetworking architecture as the unchallenged centerpiece of the global Internet.