SNA routing: Past, present, and possible future

by J. M. Jaffe
F. H. Moss
R. A. Weingarten

This paper reviews the evolution of routing mechanisms in IBM's Systems Network Architecture (SNA) since its inception in 1974 to the present. Routing mechanisms are related to changes in the application and communications environment. Also discussed are possible evolutionary paths that may be taken in the future to address the problems of large heterogeneous networks.

IBM's Systems Network Architecture (SNA)[1,2] is a layered structure that provides a set of formats and protocols to enable various systems to communicate with one another. At the lowest layers of the architecture are protocols to handle the transmission of messages on physical data links. The path control (or routing) layer provides for transmission over routes that may traverse multiple nodes. The higher layers (session layers) handle the end-to-end communication between two users of the communication system. The routing layer in particular tends to be very sensitive to the changing user, application, and communication environments. These changes cause the routing function to evolve continuously over time to account for many factors, including changing user needs, new applications, growth in network size, network interconnection, and new transmission facilities such as satellites and local area networks.

This paper traces the evolution of SNA routing from the introduction of the architecture in 1974 to the present and speculates on possible directions that may be taken to meet future needs. In addition to describing the routing mechanisms that are present in each release, the authors discuss the rationale underlying various design choices.

The paper first reviews the changing environment that has contributed to the evolution of SNA routing, as well as the projected environment for the mid-to-late 1980s, where it is emphasized that large, dynamic, and heterogeneous network structures will provide new challenges. A description of basic routing principles is then presented, followed by a historical perspective of SNA routing. The discussion spans the period from the announcement of the capability to route on host-based "trees" in 1974, through the multihost Advanced Communication Function (ACF) announced in 1976 with additional functions announced in 1978, to the SNA Network Interconnection technique announced in 1983. The paper concludes with a speculative discussion of possible evolutionary paths that may be taken to meet the challenge of large, dynamic networks expected in the mid-to-late 1980s.

Changing environment

Prior to and into the early 1970s, two types of networks were in general use, the interactive and

the batch.[3] The interactive network environment was characterized by terminals attached to data processing machines supporting a single major application program. This program contained specific device support to control usage of the terminals. These terminals were attached to teleprocessing lines which were also controlled by the same application program. Messages were simply routed to a line address and forwarded to the attached terminal. Several large networks were in use during this time. These pioneering networks were mainly in the airline[4] and banking industry[5] sectors.

The initial batch network environments consisted of point-to-point transmission between paired paper tape devices, card readers/punches, followed by magnetic tape devices. This batch mode of transmission changed to on-line batch because of the development of the concept of spooling and creation of higher-speed batch terminals. Since batch transmission was point-to-point, sophisticated routing mechanisms were not needed. Teleprocessing lines needed for batch data transfer were separate from those used for the interactive traffic. Line sharing was not possible since the line control for most of the teleprocessing lines, at that time, was handled in the application programs themselves.

It was generally recognized in the early 1970s that systematic network architectures would be needed to allow for greater consistency in terminal attachment and line control protocols, for sharing of expensive teleprocessing lines by both batch and interactive traffic, and for clean separation of application and network functions. The need for routing mechanisms also became apparent. Since terminal sharing was a major goal of the architecture, application programs could no longer be directly connected to a terminal device. It was necessary to route messages to different devices from different application programs on an as-needed basis. Routing mechanisms were needed to provide the structure to forward those messages to the appropriate devices.

Several network architectures emerged in the first half of the decade to meet these requirements, including ARPANET[6] from the U.S. Advanced Research Projects Agency, SNA from IBM, and DECNET[7] from the Digital Equipment Corporation.

As networks using these early architectures were installed during the mid-1970s and experience was gained in their operation, several major problems related to routing surfaced. First, the availability of communication paths between nodes depended on the availability of the underlying physical elements within the paths. Failure of one element would cause disruption of network traffic between the nodes. Thus, methods were needed to allow for alternate routes to bypass failures and to allow traffic to continue to flow. Also, the trend towards sharing between interactive and batch traffic caused performance problems, since both classes of traffic contended for the same scarce resources of link capacity and nodal buffers. Thus, mechanisms were needed for assigning priority to different traffic types on the same route, as well as for allowing the separation of different traffic classes onto different multiple routes through the network. These capabilities were introduced into network architectures in varying degrees, with various solutions, during the late 1970s.

Every network entity that is to communicate must be assigned a network address.

During the beginning of the 1980s, there was a dramatic increase in reliance on the use of computers, particularly in the business world. Also, systems employing microprocessors allowed for the distribution of intelligence closer to the end user, providing more reliable service and faster response time, as described by Scherr.[8] These factors caused a number of changes in the network environment.

The number of devices being incorporated into a single network was expanding rapidly, and a need developed for the interconnection of networks employing like and different architectures. Public networks employing interconnection standards such as X.25[9] grew in size, and multinational corporations began the creation of worldwide private networks. Transmission bandwidth requirements were being met by increased use of satellites and the introduction of fiber optic links. All of these devel-

opments implied the need for new mechanisms for routing messages within and between networks. In the future, proliferation of personal computers, as well as devices built on videotext, will play a major role. New transmission facilities, i.e., local area networks that may require greater routing flexibility to rapidly add additional network entities, will be developed. Routing methods that allow the inclusion of extremely large numbers of devices with potentially different architectures in a dynamic, nondisruptive methodology will be needed.

Basic routing principles

This section discusses basic routing techniques, concentrating on the principles behind end-to-end routing. Prior to this discussion, a general description of a network address and routing mechanisms will be provided.

Every network entity, such as terminals, application programs, network control points, etc., that is to communicate must be assigned a network address. In most networks there is a relatively small number of major data processing and communication multiplexor nodes and a much larger number of network entities or minor nodes (e.g., terminals, displays, application programs, etc.) associated with each major node. This association makes it convenient to have a structured address of the form major_node . minor_node. This form is analogous to area_code . phone # for telephones or zip_code . street address for mail.

Routing is the mechanism whereby messages are directed through the network, possibly across intermediate nodes and links, from their origin address to their destination address. The two basic mechanisms that can be used to achieve this directing are either node-by-node routing or end-to-end routing. In node-by-node routing each node receiving a message decides independently which node the message is to be sent to next. In end-to-end routing, the path to be traversed by a message is determined before the message is sent, and the message proceeds in a systematic fashion from origin to destination along this predetermined path.

The choice of node-by-node routing versus end-to-end routing depends on many factors, including the nature of higher-level protocols and the philosophy of network control and management. Node-by-node routing tends to be the method of choice in networks employing datagram protocols at the higher level, whereas end-to-end routing is most often used in networks employing end-to-end virtual circuits and session protocol at the higher levels, such as TYMNET[10] and SNA. The remainder of this paper focuses on end-to-end routing.

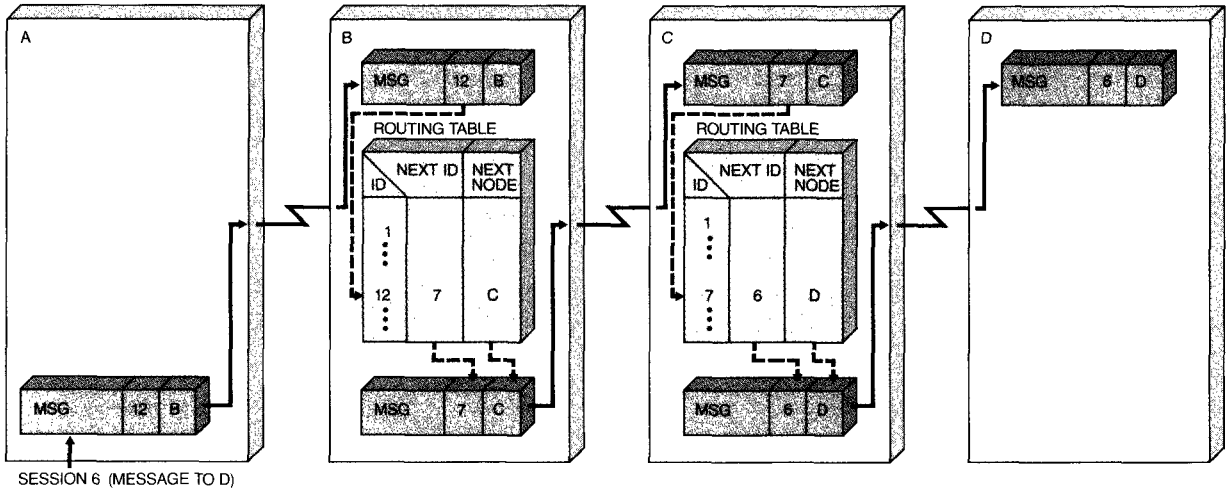
An end-to-end static session routing mechanism has been selected for SNA.

An end-to-end routing mechanism should support multiple routes between nodes if there are multiple physical paths between the nodes. This support is desirable because there may be multiple service classes or traffic types (e.g., batch and interactive), changing traffic patterns, and the ability to provide high availability through "back-ups" at failure.

There are several possible mechanisms for implementing end-to-end routing in a network. The approach that is generally employed is to specify at the origin a route identifier in the message header, and to specify the next node along the route by routing tables contained in each network node. These routing tables may be defined statically as described by Maruyama[11] or dynamically as discussed in this paper. When a message is received at a node, the route identifier serves as an index into the routing table to determine the next appropriate node. There are several possible choices for the form of the route identifier and the routing tables.

An example is provided in Figure 1. Session 6 at node A is transmitting a message to D. Associated with this session is a route end point that specifies the next node (B) that must receive the message and an identifier (12), by which that node understands the route. This identifier is used as a direct index to B to very rapidly determine the next node and identifier. This process continues until the message arrives at the destination node where the route end point routes the message to the relevant end user or minor node.

Figure 1 Routing example



Historical perspective

As is well known, SNA utilizes an end-to-end static routing mechanism. This mechanism was evolved from the initial announcement in 1974 through the present. The following section describes the evolution that has occurred in SNA during this time. The dates being used are the announcement dates. The actual product shipment dates vary up to 18 months from the announcement date.

SNA routing, circa 1974. When SNA was announced in 1974,[12,13] the routing mechanism utilized two physical addresses called the origin and destination network addresses. Each of these network addresses was divided into two parts, the *subarea*, also referred to as major node, and the *element*, or minor node, fields. The subarea field was used to route a message to a subarea node, such as a System/360 or System/370 or an IBM 3704/3705 Communications Controller. The element field was used by that subarea node to route the message to the appropriate local resource associated with that node. Figure 2 illustrates this division. Note that the System/360 and IBM 3705 are subarea nodes, whereas the local resources associated with the nodes are considered elements, such as host application programs or IBM 3705 attached terminals.

Table 1 Possible combinations

Subarea/Element Bits	Subarea/Element Combination
1/15	2/32,768
2/14	4/16,384
3/13	8/8,192
4/12	16/4,096
5/11	32/2,048
6/10	64/1,024
7/9	128/512
8/8	256/256

The subarea and element fields are contained within the *Transmission Header*, which precedes the user message as illustrated in Figure 3. These network addresses become fixed after the initiation of a session and remain the same until session termination. Sessions are established by an SNA service called the System Services Control Point (SSCP). The SSCP provides the correlation between the network names used by SNA end users and their network addresses. Four Format IDentifiers (FID) were introduced to distinguish different routing capabilities. The FID0 and FID1 were used to denote routing between subarea nodes. The FID0 was used for pre-SNA device types, and the FID1 was used for SNA devices. The FID2 and FID3 were used for routing from the subarea node to either a cluster controller, such as the IBM 3274, or a terminal node, such as the IBM 3270. The FID2 and FID3 used a simplified version of the network address called a *local address*, which is only understood locally by the subarea node.

From the FID1 header of Figure 3, we note that each network address, origin, and destination consist of 16-bit fields. This allows an absolute maximum of 65 536 individual addressable units, called *Network Addressable Units* (*NAU*), to be defined within an SNA network. The selection of the 16 bits for the implementation of the SNA address fields was not arbitrary. It was based on several factors which included the following:

1. Line costs were critical. The objective was to reduce the number of bits considered as "overhead" in transmitting over a teleprocessing line. The network addresses and control functions were considered necessary overhead.
2. The maximum of 65 536 network addressable units appeared ample for current and future usage. In 1974, SNA routing consisted of a single-system tree network which limited the number of subarea nodes contained in any SNA network. Even when future network needs, i.e., meshed networks, were considered, the 16-bit structure appeared sufficient.
3. A 16-bit address structure had already been implemented in the IBM 3704/3705 Communications Controller for pre-SNA routing in March 1972. This structure was selected to allow easy and fast manipulation of control blocks and tables since the communications controller was a 16-bit machine. It appeared that a 16-bit SNA network address was a natural match for the addressing and instruction usage structure of the controller as well.

The combination of the subarea and element field was called the network address. The boundary between the number of bits assigned to the subarea and the element fields could be specified by the user but was fixed within a given network. Initially no limits were set on the boundary placement, but in 1976 this condition was changed to specify that the subarea field had to be from one to eight bits.

Table 1 illustrates the boundary bit split and the number of subarea nodes and elements that could be addressed based on the user-selected split. For example, if a user selected a 6/10-bit split, then it would be possible to address up to 64 subareas with each subarea node containing up to a maximum of 1024 elements.

The routing structure announced in 1974 remained unchanged until the announcement of the multiple-route function in 1978. The routing mechanism

Figure 2 Subarea-element distribution

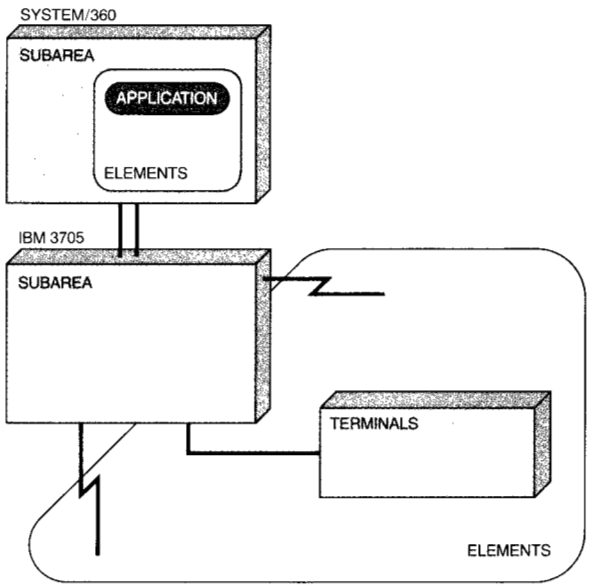
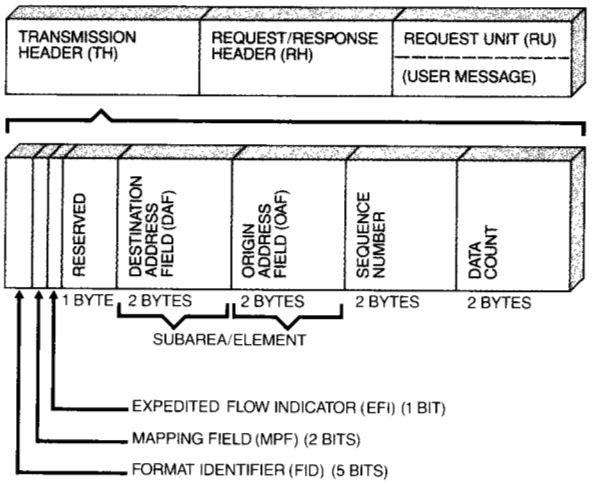


Figure 3 Message and header



used in an SNA network, called *destination routing* or *source independent routing*, also remained unchanged from 1974 through 1978.

Destination routing. Routing was based only on the destination subarea field regardless of the origin or source. Each subarea node contained a routing table unique to that node which specified how the mes-

Figure 4 NCP outbound path control routing table

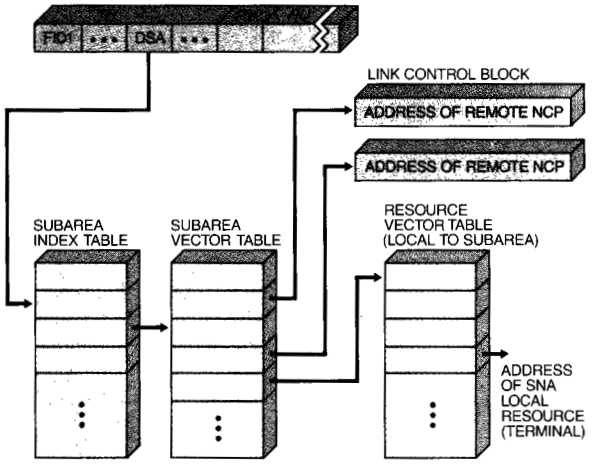
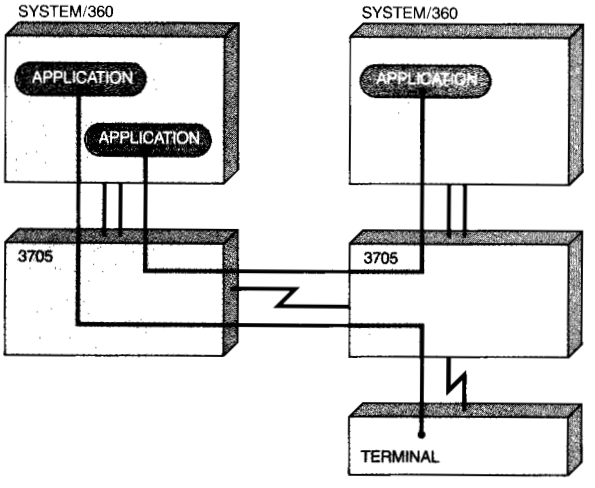


Figure 5 Session capabilities (1976)



sage was routed in that node. The routing table was organized by destination subarea number and indicated the "next leg of the journey" to which the message should be passed on its way to the destination subarea. For SNA networks created from 1974 through 1978, the next leg was either a System/360 or System/370 channel (called by SNA a link), a Synchronous Data Link Control (SDLC) link, or the subarea node itself. This is illustrated with a representation of the Network Control Program/Virtual Storage (NCP/VS) routing table[14] in Figure 4. The destination subarea was obtained from the transmission header for processing in the SNA *path control* layer of the node. In this layer, a table look-up was performed using the Subarea Index Table to the Subarea Vector Table, to identify to which leg the message should be queued next. A message that did not have the same destination subarea as the NCP/VS node itself would be queued to the link queue identified by the pointer in the Subarea Vector Table. A message destined for this subarea node would be processed further via another table to identify to which element it should be sent. In the NCP/VS node, the element portion was processed using the Resource Vector Table that pointed to the local resource being addressed. Similar processing was contained within the host access method with differently named tables, with one exception. A host subarea node did not allow routing through that host to another subarea node from messages received from the network.

Each of the subarea routing tables was statically created by the user, i.e., system administrator or system programmer, via a system generation process on a node-by-node basis. All that was needed for the routing table was a list of potential destination subareas and an indication as to which outbound link queue should be used for the message routing function. No subarea node needed to know the complete path used between two subarea nodes for an SNA session except perhaps for network management purposes. The routing mechanism was nonadaptive to changes in network topology. Topology changes required regeneration of the routing tables and reloading of subarea nodes to utilize the changes.

The destination routing technique was used in 1974 for a single-system tree network that could contain multiple IBM communications controllers either locally or remotely (1975) attached to other communications controllers. Routing of session traffic was only allowed between host application programs and terminals attached to any of the communications controllers in the network.

Routing, circa 1976. In 1976, the Advanced Communication Function of networking was announced.[15] The networking function enhanced the routing facility by allowing the establishment of sessions between host application programs and other host application programs or attached host terminals, as illustrated in Figure 5.

Although multiple host nodes were allowed in a network, routing between them was only allowed via

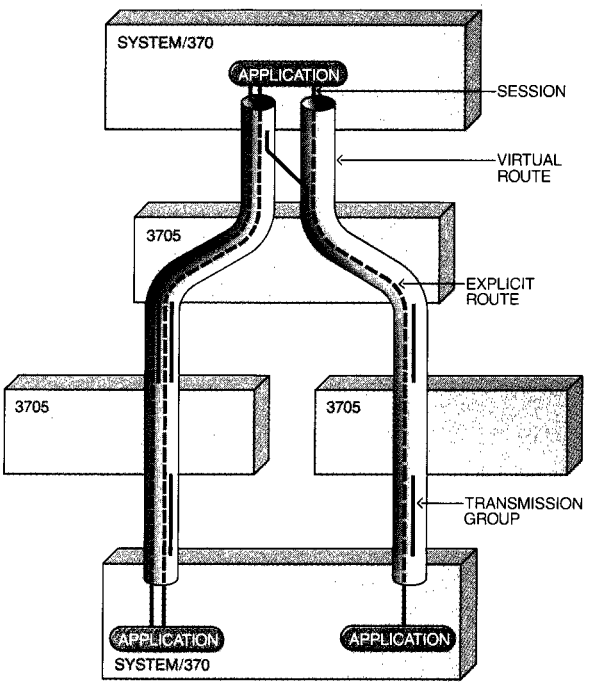
links between their locally attached communications controllers and not through channel-to-channel connections. Also allowed was the capability of host nodes to act as intermediate network nodes.

This topology has been called a mesh of trees or a grafted tree network. As in the 1974 announcement, subarea routing continued to use the FID0/FID1 transmission header as well as the destination subarea routing mechanism. Although the routing mechanism remained unchanged, the method used to obtain an SNA resource network address changed. A new function of the SSCP called the *Cross Domain Resource Manager* (*CDRM*) was created. Each CDRM had an understanding of the network name-to-address correlation for resources that would participate in cross-domain sessions under control of its SSCP. Communication between CDRMs was used to obtain the network address pair to be used for the establishment of an SNA session.

Routing, circa 1978. In 1978 the networking functions of SNA were further enhanced by the announcement of the capability to establish multiple or alternate routes between two subarea nodes.[16,17] This new function made possible a meshed network topology by eliminating the distinction between the local and remote 3705/NCP/VS. This support was announced to satisfy several requirements such as network load distribution, better route selection for better service needs, and the capability to circumvent network component failures. The route between two subarea nodes was called an *explicit* or physical route. An explicit route defined an ordered set of nodes and transmission groups from one subarea to another. A transmission group denoted a user-designated set of parallel links defined between two subarea nodes which was viewed as one logical link by the path control layer. Message traffic for a specific session assigned to a path using a transmission group would be queued for transmission over any available link within the specific transmission group. To ensure that message order was maintained at the receiving session end, the receiving end of each transmission group would reorder any out-of-order messages.

Eight of these explicit routes were allowed to be defined, again statically via a system generation process, between any two subarea nodes. A virtual or logical route was used to manage an origin-to-destination subarea protocol without being concerned with the explicit route in between. Virtual routes consisted of two parameters, a virtual route

Figure 6 Virtual route assignments



number (up to eight) and a transmission priority level (up to three). A virtual route number was mapped at activation to an explicit route number with a transmission priority associated with each virtual route, thereby allowing for 24 virtual routes. Multiple virtual routes could be mapped to the same explicit route. SNA sessions were assigned to the same or a different virtual route as portrayed in Figure 6.

The multiple-routing function did not increase the number of subarea nodes or network elements that could be addressed but changed the method used for network routing purposes. To provide this multiple-routing capability, several internal SNA changes were needed. First, the basic destination routing mechanism required modifications, since the subarea node receiving messages needed a method to determine to which of eight "next legs" or explicit routes the message should be sent. Since the explicit route identified the physical path, the explicit route identifier was added to the routing table to be used in conjunction with the destination subarea number as an index. Figure 7 illustrates this concept. Viewing this figure, we can see that a combination of the destination subarea and the

Figure 7 SNA 4.2 routing

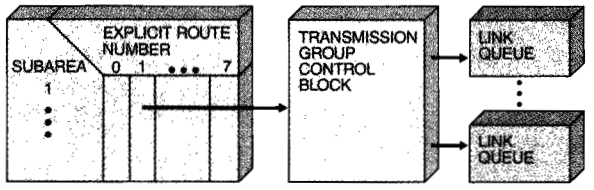
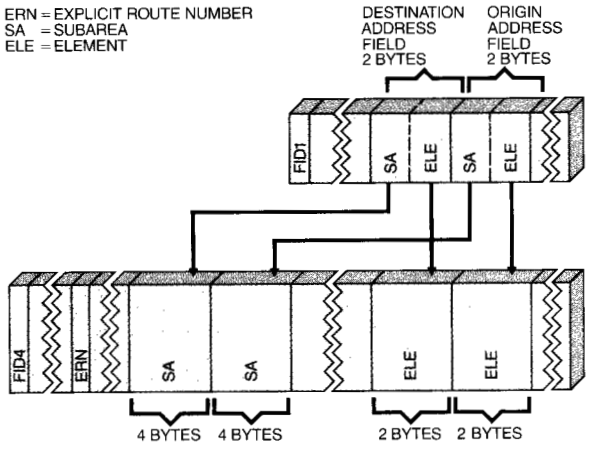


Figure 8 FID1 to FID4



explicit route number indicates to which transmission group the message is to be queued. Note that the transmission group points to link queues which can be used to transport the message from node to node.

Second, the transmission header created in 1974 did not have sufficient room for growth to contain the fields needed to address and control the multiple-route function. Therefore, a new header called the FID4 was introduced. This header, illustrated in Figure 8, contained ample space for fields needed for the multiple-route function, as well as for fields reserved for future use. The specific fields added for routing purposes were the explicit route numbers, one in each session direction. Other new fields were added, and fields that appeared in the FID1 header were redistributed as well. Note from Figure 8 that the subarea and element fields were separated in this FID4 header but that the total 16 allowable bits remain the same.

Third, a new mechanism was needed by the SSCPs to establish a session, since the user could now request that the session be assigned to one of many potential virtual routes. A new concept called the *Class of Service* was devised. It allowed the user to specify an ordered list of virtual routes. The session was assigned to the first available virtual route in the list.

Although a new transmission header, the FID4, was announced, it was added in an evolutionary manner. To allow back-level nodes to connect to a network where several nodes supported the multiple-routing function, migration support was provided. Any route whose origin, destination, or path traversed a subarea node that was back-level had to be identified as a migration route and be assigned the same characteristics as a route created prior to this multiple-route release. It was the responsibility of the newly released nodes, ones attached to the back-level nodes, to convert the FID4 to a FID0 or FID1 header prior to forwarding the message.

When the multiple-routing host software was initially released, the host intermediate network node (INN) function was not supported, although the communications controller (IBM 3705) did provide that support. The host INN function was announced in 1981 as the host channel-to-channel support.[18]

With the introduction of the multiple-route function, the major problems associated with network availability were reduced but not eliminated. As in the past, the routing tables were statically created, although a package called the Routing Table Generator[19] was provided to aid the network administrator. These routing tables were again nonadaptive to network changes. Additions and deletions of network nodes required regeneration and reloading of the subarea nodes prior to their use. Although an alternate routing function was provided, user involvement was still required to restart a failed session. With a need to provide greater network availability, additional requirements were being voiced in the area of the creation, operation, and management of larger SNA networks. The most prominent large network requirement dealt with the capability to allow routing between multiple SNA networks that could have duplicate network addresses or different addressing splits. To satisfy this requirement, a technique called SNA Network Interconnection was announced in 1983.

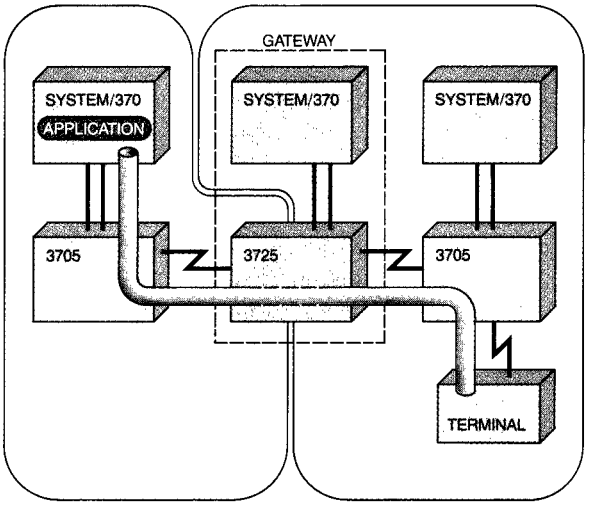
Routing, circa 1983. The SNA Network Interconnection technique[20] allows SNA sessions to be established between resources that could span multiple

SNA networks, as illustrated in Figure 9. Network routing for these internetwork sessions still utilizes the destination subarea and explicit route number technique. The difference is that as an internetwork session proceeds from network to network, the destination subarea and explicit route numbers change. These changes take place in the ACF/NCP/VS[21] gateway nodes. In the ACF/NCP/VS gateway node, the routing table structure remains unchanged as defined in the multiple-routing SNA function, although several copies of the routing table, one per attached network, exist. As a message within an internetwork session arrives at a gateway node, its destination subarea field is used to scan the routing table. If this destination subarea is the gateway, the element field is used to identify the local resource. In the gateway case, the local resource[22] is actually a control block that specifies the routing data of the next network. This routing data is swapped with the data found in an incoming FID4 header, and the message is transferred within the gateway to the path control associated with the next network for routing purposes. The method used to create the routing tables is the same as in the previous release. Subarea numbers, explicit route to outbound Transmission Group (TG) queues, are still specified for the ACF/NCP/VS node, although several additional items have been added to establish network identities, multiple subarea addresses for a gateway ACF/NCP/VS, and other gateway fields.

Further SNA routing changes will be necessary due primarily to network growth and the need for more dynamic changeability of network configurations.

These updates to both the ACF/NCP/VS and the ACF/VTAM[23] (Advanced Communication Function/Virtual Telecommunications Access Method) program products have again been done in an evolutionary manner. SNA products that do not provide this support[24] can participate in an internetwork session, provided that the appropriate gateway products are contained in the network.

Figure 9 SNA Network Interconnection configuration



A large number of configurations of network interconnections are permitted. They include two networks interconnected at one or multiple gateway nodes, two or more networks interconnected to the same gateway nodes, and cascaded interconnected networks. Although each individual network generates its own static routing table, changes to one network are masked from changes in other networks.

Table 2 contains a summary of the major evolutionary steps in network routing thus far presented. Figure 10 illustrates examples of allowable network topologies.

The possible future

The previous section described the evolution of the SNA routing mechanism as the applications environment and communications environment changed. If we were to project the changing environment into the late 1980s, we would conclude that changes will continue in two generic ways:

1. Networks will continue to become larger because of decreased cost and increased usage of distributed systems.
2. This growth will imply the need for more dynamic changeability of network configura-

tions which will be accelerated with the growth of faster switching facilities such as PBXs, local area networks, X.25 networks, etc.

Below we discuss the issues that arise from the above two observations. We then speculate on the potential approaches that SNA could use to address these issues.

Issues in large networks. The key issues arising from large networks are the current addressing structure and the cost of maintaining large numbers of statically defined routes. The addressing structure permits 256 subarea nodes. Fewer nodes are permitted if a large number of elements need to be associated with any one subarea, as can be seen from Table 1. SNA Network Interconnection alleviates this problem to some extent by allowing each attached network to utilize the maximum addressability within its boundaries, yet create sessions with resources in other SNA networks. However, it is not necessarily a complete solution in a single network. It would seem that enlarging the capabilities for addressability is necessary in any method that allows for large networks.

The main issue that arises with dynamic networks is the need or desire to change routing definitions more easily than is possible today.

The cost of maintaining route definitions is related primarily to table size. The table size issue is having an SNA intermediate network node (e.g., ACF/NCP/VS) maintain tables that are linear in size with network size. As networks grow, the resultant storage required at each node grows proportionately. In addition, whenever routes fail or are repaired, network traffic is generated to inform affected nodes. The amount of route status traffic handled at a node is roughly proportional to the number of table entries and therefore roughly proportional to net-

Figure 10 Examples of allowable network topologies: (A) Tree; (B) Grafted Tree; (C) Meshed

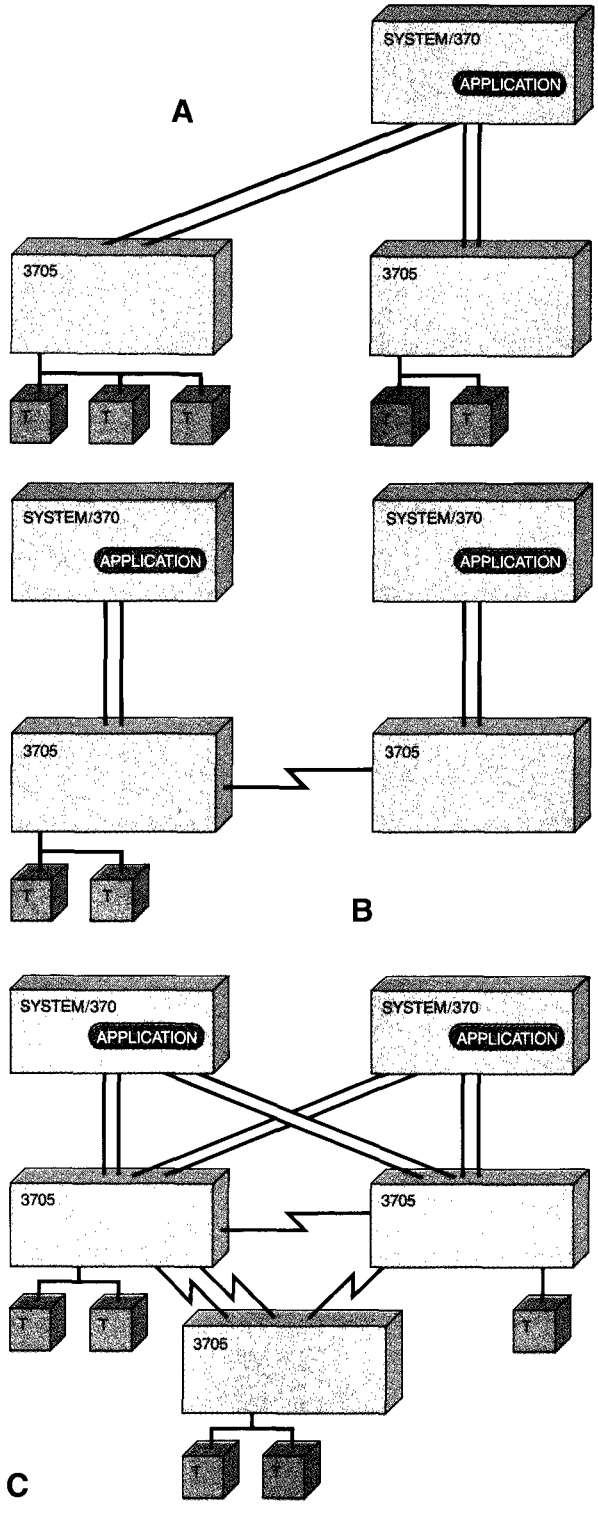
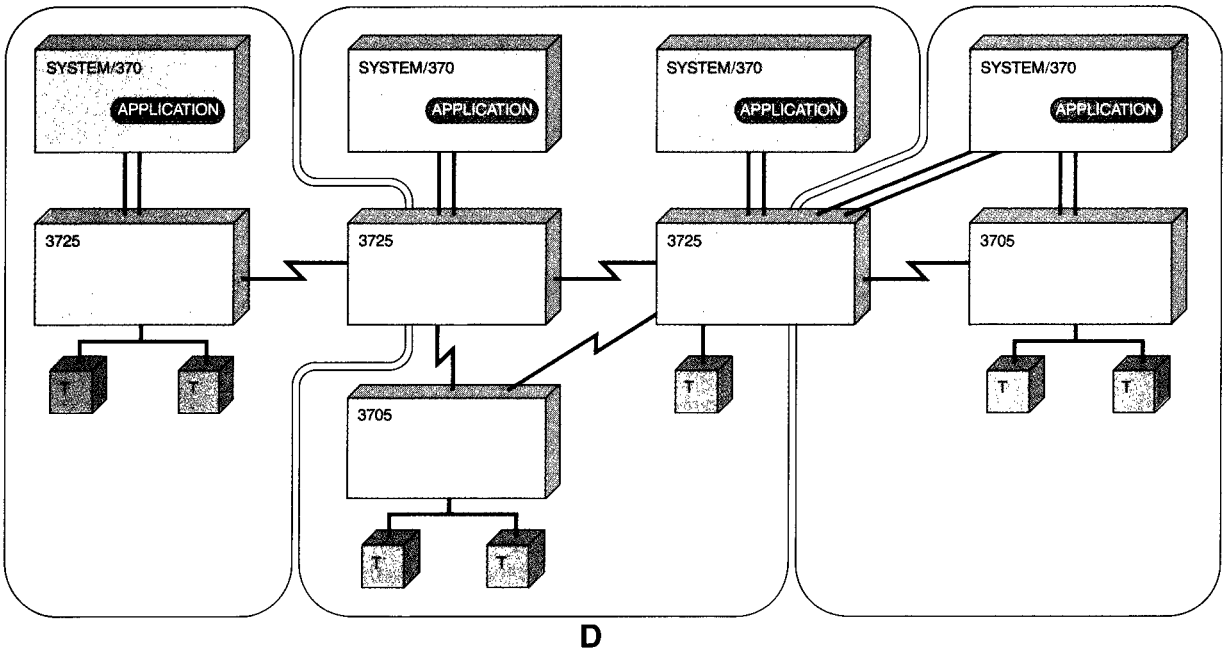


Figure 10D Mesh of meshes example



work size. Thus, the quantity of network control traffic is also a large network problem. These problems will be mitigated if storage and bandwidth become cheaper at a faster rate than the rate at which networks grow.

Issues in dynamic networks. The main issue that arises with dynamic networks is the need or desire to change routing definitions more easily than is possible today. If a new subarea node or transmission group is added to an SNA network, there is no automatic mechanism for adding or deleting routes in a network node without disruption.

Other important issues are availability and performance. The availability issue is that no static mechanism with a fixed number of routes will *always* provide a route when physical connectivity exists. The performance issue is concerned with wanting to assign sessions to routes dynamically[25] to provide better load balancing.

To continue in its evolutionary pattern, SNA will be expected to find ways to meet new challenges. The following subsections describe three potential approaches. The first is specifically oriented toward large networks—without too much concern for dynamicity. It is followed by a set of approaches that are primarily concerned with dynamicity, but address large network problems as well.

Current techniques. Currently there are several available techniques that can be used to aid in the creation and maintenance of large networks. These techniques provide for varying degrees of dynamism in the current SNA routing structure.

Preplanning. Although changes in the subarea topology may occur frequently, it is usually possible to predict relatively far in advance what changes are to be made. In the *preplanning* technique, when a network administrator plans routes (for example, on January 1), the network administrator uses the projected maximal topology for July 1. Although certain routes may be inoperative for many months (since links or subarea nodes are not yet in the network), SNA will permit those routes that are operative to function normally. Moreover, the fact that SNA allows a large number of routes (eight)

Table 2 Major steps in network routing

Year	Topology	Routing Technique (Announced)	Routing Indicators
74	tree network	single path	destination subarea
76	multitree, grafted tree, mesh of trees	single path	destination subarea
78	mesh	multiple path	destination subarea plus explicit route number
83	mesh of mesh networks	multiple path through multiple networks	destination subarea plus explicit route number

implies that even without routes that are not yet operative, there is still plenty of "routability" in the network. When the new subarea nodes and links are added into the network, the routes can become operative without disruption to the rest of the network, since they have previously been defined within the routing tables of other subarea nodes.

Using current SNA techniques, many approaches are available to address large and dynamic networks.

Maintaining old route definitions. Whether or not preplanning is used, when a new, unplanned route needs to be added, it may often be done without major network disruption. Typically, new routes that are added near a new subarea node or link are local to that subarea node or link. In addition, if new routes are *added* without *deleting* or *changing* old route definitions, they may be gradually added to the routing tables of the affected subarea nodes without any need to change *all* tables involving a route at once. A route then simply becomes operative after each subarea node has had an opportunity to change routing definitions.

Route number reservation. The technique of not changing old routes when new routes are created has the limitation that once eight routes have been defined between two subarea nodes, it is no longer possible to add additional routes between them without deleting old routes. One may plan for this eventuality, however, by defining only those routes that are really necessary initially and reserving route numbers for later uses. In a typical scenario, on January 1 one may design routes for the projected July 1 network using route numbers 0, 1, 2, and 3. On July 1, route number 4 may be used to plan until the next January 1, etc. and may be gradually added to the routing tables in the network. In this way, dynamic change capability is guaranteed for a number of years.

Four route numbers at a time. Although route number reservation allows the dynamic addition of routes, it does not allow addition indefinitely. Permanent dynamic addition is permitted by the technique of using only four route numbers at a time. In this technique, one specifies routes using numbers 0, 1, 2, and 3 in the original configuration of a network. Then, when a collection of routes need to be added, a new set of routes is specified using numbers 4, 5, 6, and 7. These routes will generally include duplicates of most of the original routes as well as the new routes. Routes with numbers 0, 1, 2, and 3 are used until network nodes are (gradually) regenerated with both routes 0 to 3 and 4 to 7. Once the procedure is complete, a switchover to routes 4 to 7 is initiated, and the entire network begins using the new routes. During subsequent regenerations, routes 0 to 3 are removed from tables so that after some time no node has route numbers 0 to 3 in use. Then, when new routes need to be added, numbers 0 to 3 may be used in an analogous way. Thus, one

may permanently switch between 0 to 3 and 4 to 7. Note that there may be a need to reserve explicit route 0 for migration purposes.

Hierarchical routing table organization. This method entails creating an addressing hierarchy within the subarea field. An addressing hierarchy is advantageous for several reasons. Among these are economy of routing table sizes, reduction of network route status information flow, and minimization of the impact of topology changes. This technique is based on several assumptions about the distribution of network nodes in large networks. Consider a typical large network which spans the entire United States. Typically, several large data processors are grouped in major industrial or commercial areas with teleprocessing lines connecting terminals in outlying areas. Additionally the major group of data processors are attached to the other centers via high-speed teleprocessing lines from a specific node in that center. Taking an IBM internal network as an example, one can see from Figure 11 that they have four large communication centers.

Let us take a subset of this network using the four centers with 18 nodes and view the routing table at subarea node 1 for explicit route 1. The subset network is illustrated in Figure 12 with the routing table in Figure 13. This routing table is consistent with tables that are produced in a multiple-routing environment. Rather than specifying Transmission Group (TG) queue control blocks as the table entries, we will use the notation that identifies the outbound TG from the network figure. For example, a message destined for destination subarea 11 will be placed on an outbound TG which will initially be sent to subarea 3, i.e., TG 1-3. Then the routing table in subarea 3 will forward the message on the TG specified in its table. From the routing table of Figure 14, we can see a definite pattern of TG usage in relation to specific sets of subarea nodes.

From Figure 14 we note that messages destined for subareas 15 to 18 are placed on the same outbound TG queue 1-4. By combining other subareas into groups, hereafter called *clusters*, a new routing table identified as Table A can be created. This table is not yet usable for routing purposes since there is no mechanism that can be used to describe a cluster for routing purposes associated with the table. Proceeding one step further, we can create a cluster table, denoted as Table B, which contains a hierarchical addressing structure that could be used for routing purposes.

Figure 11 The IBM Consolidated Communications Data Network (CCDN)

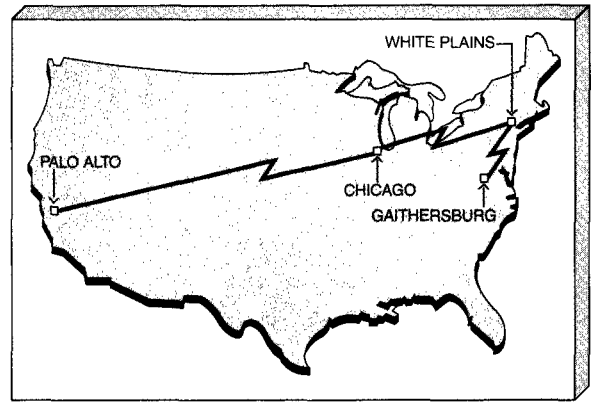
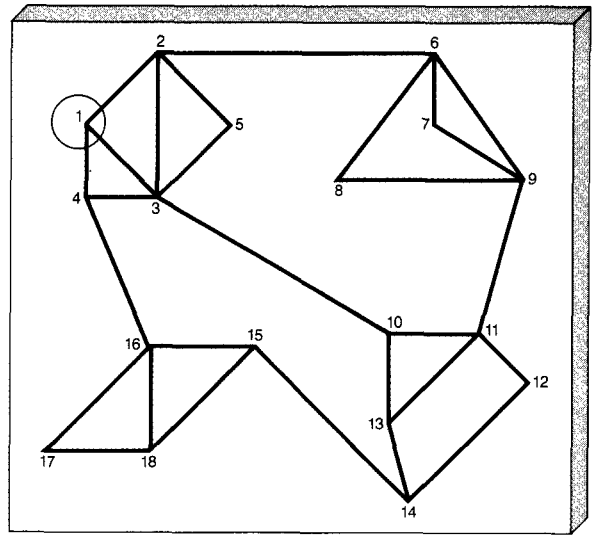
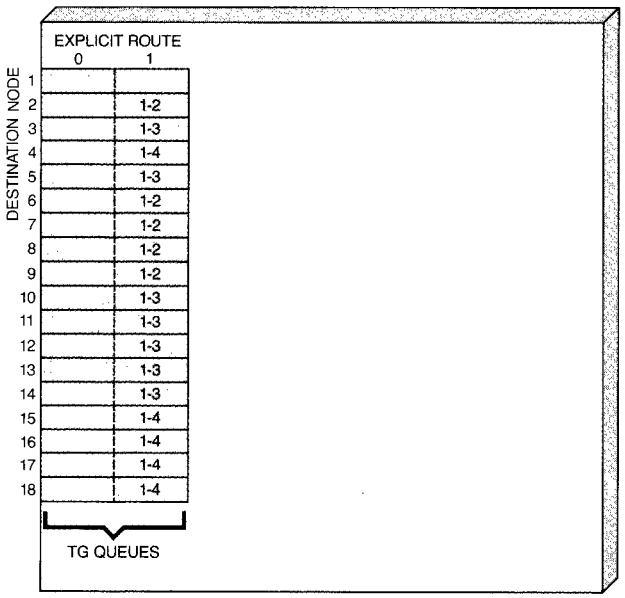


Figure 12 Subset of CCDN with nodes



The routing mechanism would change slightly from the one in use today. Currently, the entire destination subarea coupled with the explicit route number is used as an index into the subarea routing table to locate the outbound TG queue. In the hierarchical routing scheme, the cluster identifier, part of the destination subarea, coupled with the explicit route number, would be used as an index (Table B of Figure 14) into a route table to identify the outbound TG until the message encounters the node in

Figure 13 Routing table of network in Figure 12



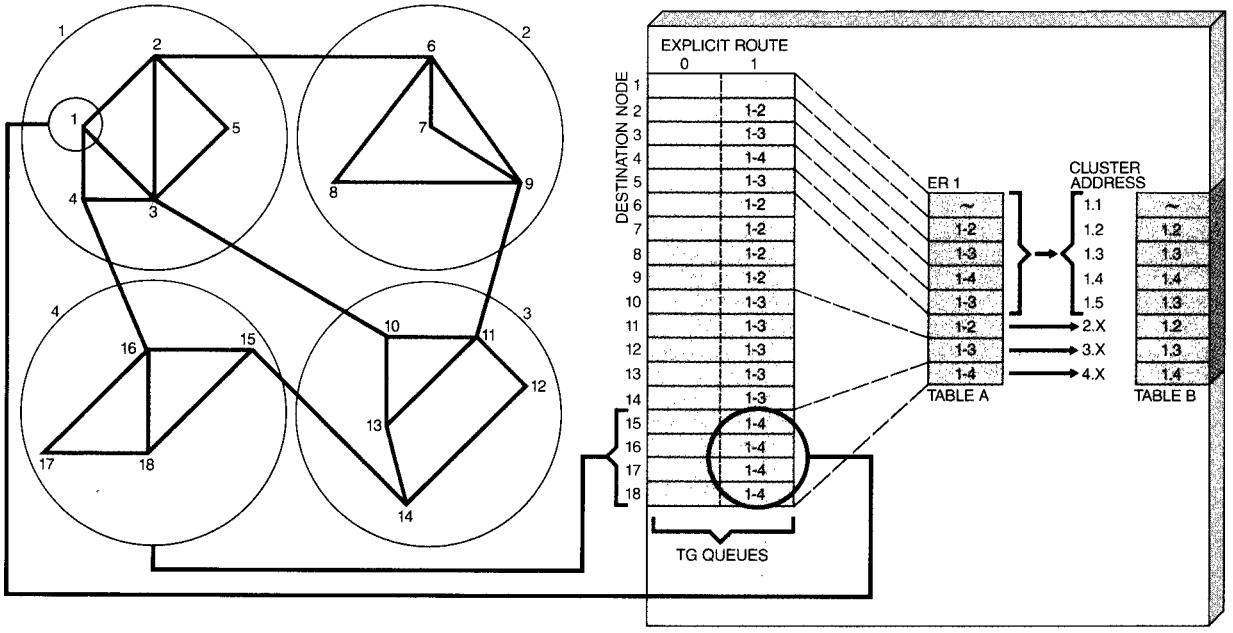
the destination cluster. Then, in that first node, routing would continue as it is done today on a node-by-node basis until the destination subarea is reached.

This clustering effect would allow the reduction of the network status message since data relating to individual nodes would not be needed. The entire cluster status could be provided. For example, if links 2-6 and 9-11 failed, a single notification that cluster 2 could not be accessed could be given instead of several messages stating that nodes 6, 7, 8, and 9 were unavailable.

Also, since routing tables use cluster numbers for routing purposes, the addition of nodes within a cluster only causes local routing tables within that cluster to be changed and reloaded. This feature would minimize the disruption caused by updates to a cluster when new nodes are included in a network topology rather than affecting the entire network. Naturally, when a new cluster is added, routing tables in other clusters must be modified and reloaded to reflect this change.

Dynamic routing. Since the inception of packet-switching networks in the late 1960s, the issue of dynamic routing has received a great deal of attention in the literature, and a number of schemes have been implemented. ARPANET,[26] DECNET,[7] and TYMNET[27] all implement various forms of dynamic routing. The schemes differ in a number of ways, including the level of dynamicity, whether the

Figure 14 Combining subareas into groups



routes are calculated in a centralized or distributed fashion, and the level of integrity and reliability provided to higher-level protocols. However, they all share the feature that the manipulation of the routing tables is taken out of the hands of humans and placed into intelligent network nodes, where adjusting smoothly to frequent dynamic change in a large network environment is thus possible.

One potential evolution for SNA routing is to provide dynamicity while preserving the predictability, controllability, and integrity of having sessions assigned to end-to-end routes which do not change during the lifetime of the session (except, of course, where failures in the route occur or if the session is of long duration compared to the frequency of network change). This way would avoid problems often associated with dynamic routing, such as message looping, lost messages, and ping-ponging of traffic. But dynamic mechanisms would allow the end-to-end routes to be automatically generated on line without human involvement, avoiding the system generation burden and network availability problems often associated with static routing schemes.

The approach in principle is quite simple. Suppose that the network of Figure 15A is operating and at the moment there does not exist a route between nodes A and D suited to a given Class of Service, such as "Interactive." There may exist other routes between A and D, but they are tailored for use by other Classes of Service, such as a "Batch" route traversing a satellite. Now suppose that a user at A desires to establish an interactive session with an application at D. A control message, which we call "ROUTE-SETUP," is sent from A to D along the path best suited to handle interactive traffic. The ROUTE-SETUP is directed along this path by an "oracle," which is described later. Now, as the ROUTE-SETUP message traverses the best path, steered by the oracle, each node along the path makes an entry in its routing table to represent the explicit route being established. When the message reaches D, a reply is sent back to A along the reverse of the path traversed on the way from A to D. When the reply reaches A, the explicit route is considered active, and message flow can begin on the new session between A and D after a virtual route is established. Subsequent interactive sessions between A and D can also be assigned to this route. The route may later be deleted when the last such session terminates. Route deletion is achieved by a control message flowing along the route, deleting entries in its wake.

Figure 15A Dynamic route setup along the best path

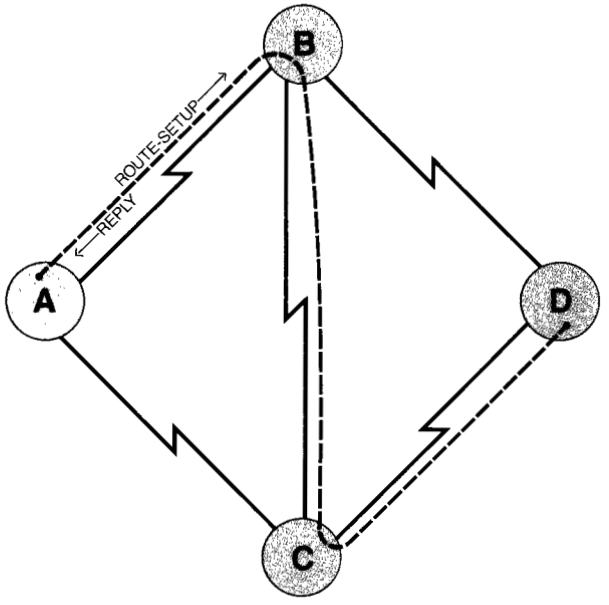
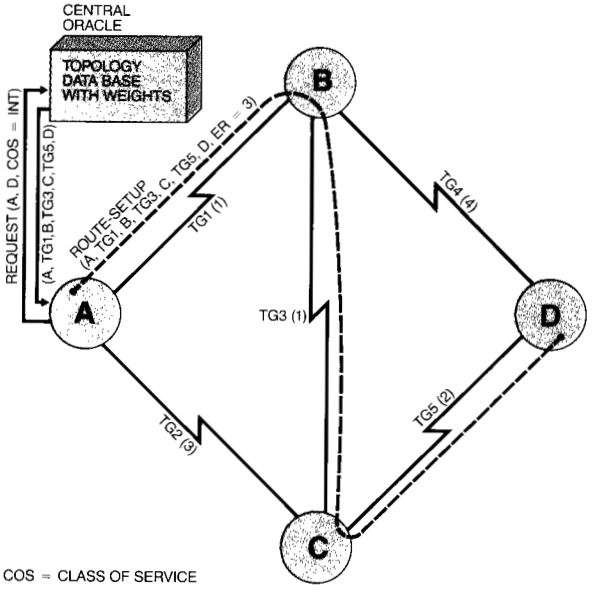


Figure 15B Centralized oracle with global information



There are many alternatives for the placement, form, and function of the oracle. One possibility is a centralized oracle that contains a data base of the

Figure 15C Distributed oracle with local information

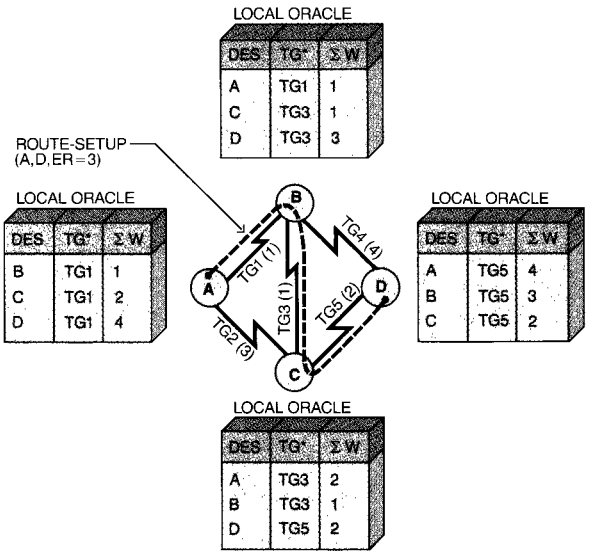
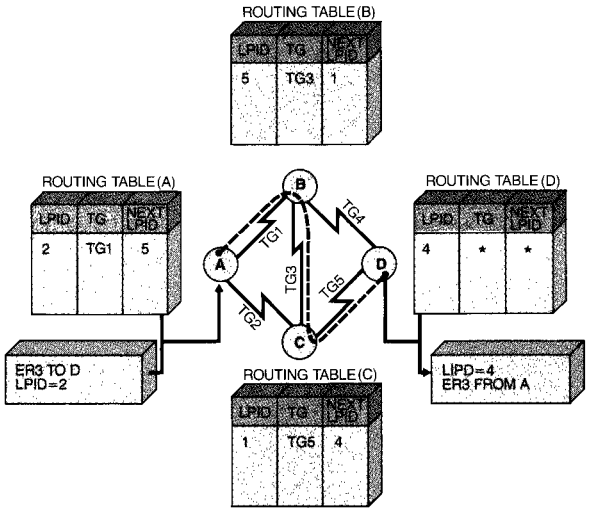


Figure 15D LPID swapping



global topology of the network and is continually updated as network topology changes. The topology data base may also associate "weights" with each TG that represent the cost of traversing the TG for a given class of service. For example, the weight of a TG related to an interactive class of service may be predicted as average delay. In the example of Figure 15B, node A would issue a request to the oracle for the best route between itself and D for the interactive class of service. The oracle would run an algorithm to calculate the path between A and D to minimize the sum of the weights corresponding to the interactive class of service (weights are in parentheses). In the example, this path is A, TG1, B, TG3, C, TG5, D with a total weight of four. The oracle would return this information to A, which would insert it into the ROUTE-SETUP message along with an explicit route (ER) number, chosen as 3 in the example. Each node receiving the message would make the appropriate routing table entry and forward the message on the TG indicated in the ROUTE-SETUP message.

Another possibility is a distributed oracle, which may take several forms. One is simply that each network node contains the global topology data base with weights as described above. Route setup would proceed in an identical fashion to that described above, where the origin node calculates the entire best path and inserts it into the ROUTE-SETUP message. Another possibility is a distributed oracle in which each node knows only local information, that is, for each destination (DES) the next TG to be taken on the best path (TG*) along with the sum of the weights on the path, W, as depicted in Figure 15C. Each node receiving the ROUTE-SETUP would make the appropriate routing table entry and consult its local oracle to find the appropriate TG on which to forward the message.

Algorithms for updating the oracle as the network topology and weights change have been extensively studied and are of considerable interest but will not be discussed here. See Reference 27 for a description of an algorithm for updating centralized oracles as implemented in TYMNET. Reference 28 describes the latest ARPANET algorithm for updating distributed oracles with global topology, where the oracles are used to route data messages directly without previous route setup. Algorithms for updating distributed oracles with local information are described in References 29 and 30.

We now discuss the form and function of the routing tables contained in each node as well as the mechanisms by which data messages are routed. As was the case with oracles, numerous possibilities exist. For example, a simple approach is source-dependent routing, whereby each entry contains source, destination, an explicit route number, and the appropriate outbound TG. Source-dependent routing is practical when routes are dynamically set up

and deleted since, on the average, the number of routes in a table at any given time is small (unlike static routing where all routes are always in the table). Creation of the routing tables and data message routing using source, destination, and explicit route numbers as a routing index are straightforward with this method.

Local Path ID is a form of source-dependent routing that could be used as a dynamic network routing mechanism.

An interesting form of source-dependent routing, Local Path ID (LPID) Swapping, for keeping routing table size small, is described in Reference 31 and could be applied here. The idea is that each explicit route is identified by a single number, the LPID, at each node through which it passes and that the LPID may be different at each node. The routing table at a node indicates the LPID for the route at that node and the appropriate outbound TG, as well as the LPID by which the route is known at the next node (see Figure 15D). For example, during the route setup process, each node independently selects the LPID for the route being set up and informs the previous node of its selection. (See Reference 32 for a detailed description of the route setup procedure.) For message routing, the LPID is carried in the transmission header and used as a routing index at each node, then swapped to the next LPID before the message is forwarded. At the origin node, the destination and ER number are mapped to the first LPID, and at the destination, the final LPID is mapped to the origin and ER number, as in Figure 15D.

In summary, the mechanisms combine the predictability and controllability of explicit routing with the flexibility and adaptability of dynamic routing. The basic mechanism of sessions using end-to-end explicit routes is preserved. Finally, dynamic routing could be offered separately or in parallel with the address expansion and hierarchical routing techniques described earlier in this section.

Concluding remarks

Despite a tremendous variety of application environments and changing requirements, SNA's layered structure has provided a flexible means for the routing mechanism to evolve. Although one might expect further developments in the application environment, there is strong reason to believe that SNA will continue to flexibly incorporate routing changes that meet new requirements. We have described, for example, how SNA is poised to handle potential new issues that arise from large and dynamic networks if and when it becomes important to do so.

Acknowledgments

The authors wish to acknowledge the cooperative efforts among SNA architects, product developers, and researchers from IBM sites in Kingston and Yorktown Heights, New York, LaGaude, France, and Research Triangle Park, North Carolina, who have over the past decade worked on the evolution of the SNA routing mechanism.

Cited references and notes

1. *Systems Network Architecture Format and Protocol Reference Manual: Architecture Logic*, SC30-3112, IBM Corporation; available through IBM branch offices.
2. E. H. Sussenguth, "Systems Network Architecture: A perspective," *Conference Proceedings of the 1978 International Conference on Computer Communications*, Kyoto, Japan (1978), pp. 353–358.
3. D. R. Jarema and E. H. Sussenguth, "IBM data communications: A quarter century of evolution and progress," *IBM Journal of Research and Development* **25,** No. 5, 391–404 (1981).
4. The Semi-Automatic Business-Related Environment (SABRE) was operational in 1964 connecting 1100 agents, located throughout the country, to the system.
5. IBM announced the 1062 Teller Terminal in November, 1962, following a study with the First National Bank of Chicago to define an on-line savings system.
6. J. M. McQuillan and D. C. Walden, "The ARPANET design decisions," *Computer Networks* **1,** No. 5 (September 1977).
7. S. Wecker, "DNA—The digital network architecture," in *Computer Network Architectures and Protocols,* Editor, P.E. Green, Jr., Plenum Publishers, New York (1982), pp. 249–296.
8. A. L. Scherr, "Distributed data processing," *IBM Systems Journal* **17,** No. 4, 324–343 (1978).
9. "Public data networks," *CCITT Orange Book, Volume VIII,* CCITT Recommendation X.25, International Telecommunications Union, Geneva (1977).
10. J. Rinde, "TYMNET II: An alternative to packet technology," *Proceedings of the 3rd ICCC*, Toronto (August 1976), pp. 268–273.

11. K. Maruyama, "Defining routing tables for SNA networks," *IBM Systems Journal* **22,** No. 4, 435–450 (1983, this issue).
12. J. H. McFadyen, "Systems Network Architecture: An overview," *IBM Systems Journal* **15,** No. 1, 4–23 (1976).
13. P. G. Cullum, "Transmission subsystem in Systems Network Architecture," *IBM Systems Journal* **15,** No. 1, 24–38 (1976).
14. W. S. Hobgood, "The role of the Network Control Program in Systems Network Architecture," *IBM Systems Journal* **15,** No. 1, 39–52 (1976).
15. H. R. Albrecht and K. D. Ryder, "The Virtual Telecommunications Access Method: A Systems Network Architecture perspective," *IBM Systems Journal* **15,** No. 1, 53–80 (1976).
16. J. P. Gray and T. B. McNeill, "SNA multiple-system networking," *IBM Systems Journal* **18,** No. 2, 263–297 (1979).
17. V. Ahuja, "Routing and flow control in Systems Network Architecture," *IBM Systems Journal* **18,** No. 2, 298–314 (1979).
18. Channel-to-channel support was placed into ACF/VTAM Version 2, Release 2.
19. *Routing Table Generator, Program Description/Operation Manual*, SB21-2806-1, IBM Corporation; available through IBM branch offices.
20. J. H. Benjamin, M. L. Hess, R. A. Weingarten, and W. R. Wheeler, "Interconnecting SNA networks," *IBM Systems Journal* **22,** No. 4, 344–366 (1983, this issue).
21. ACF/NCP/VS Version 3; for more information see *Network Program Products, General Information* GC27-0657, IBM Corporation; available through IBM branch offices.
22. An ACF/NCP/VS gateway may have terminals and NCP-LUs as local resources. The local resources specifically referenced by this section deal with those used for internetwork sessions.
23. *ACF/VTAM Version 2, Release 2, General Information*, GC27-0608, IBM Corporation; available through IBM branch offices.
24. ACF/TCAM can participate in internetwork sessions as long as at least one ACF/NCP/VS Version 2, Release 2, or ACF/VTAM Version 2, Release 2, product is contained in one of the interconnected networks.
25. K. Maruyama, "Dynamic route selection in session based networks," *ACM SIGCOMM '83*, Austin, Texas (March 1983), pp. 162–169.
26. J. M. McQuillan, I. Richer, and E. C. Rosen, "The new routing algorithm for the ARPANET," *IEEE Transactions on Communications* **COM-28,** 249–296 (1980).
27. C. Tymes, "Routing and flow control in TYMNET," *IEEE Transactions on Communications* **COM-29,** 392–398 (1981).
28. J. M. McQuillan, G. Fulk, and I. Richer, "A review of the development and performance of the ARPANET routing algorithm," *IEEE Transactions on Communications* **COM-26,** 1802–1811 (December 1978).
29. J. M. Jaffe and F. H. Moss, "A responsive distributed routing algorithm for computer networks," *IEEE Transactions on Communications* **COM-30,** No. 7, 1758–1762 (July 1977).
30. W. D. Tajibnapis, "A correctness proof of a topology information maintenance protocol for distributed computer networks," *Communications of the ACM* **20,** No. 7, 477–485 (July 1982).
31. G. Markowsky and F. H. Moss, "An evaluation of local path ID swapping in computer networks," *IEEE Transactions on Communications* **COM-29,** No. 3, 329–336 (March 1981).
32. A. Segall and J. M. Jaffe, "A reliable distributed route set-up procedure," *Globecom*, San Diego (November 1983).

Reprint Order No. G321-5203.

Jeffrey M. Jaffe *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598.* Dr. Jaffe has been a Research Staff Member in the Computer Sciences Department at the Research Center since 1979. He is currently the manager of the Network Architecture and Protocols project. His work for the past several years has been in the area of network architecture and protocols, in particular in the area of distributed routing algorithms. He received a B.S. in mathematics, an M.S. in computer science, and a Ph.D. in computer science from MIT, in 1976, 1977, and 1979, respectively. In 1982, Dr. Jaffe received an Outstanding Innovation Award for his work in dynamic routing. He has also received an IBM Invention Award and two Research Division Awards for his work in network architectures and satellite communications.

Franklin H. Moss *IBM Research Division, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, New York 10598.* Dr. Moss joined IBM in November 1977 in a post-doctoral assignment at the Israel Scientific Center, Haifa, Israel. He became a staff member of the Computer Sciences Department at the Research Center in January 1978. In September of 1979 he became project leader of the Network Architecture project and was promoted to manager of the group in December 1979. He assumed his current position as manager of the Communications and Distributed Systems Department in June 1981. He directs five research groups developing advanced technology for IBM distributed system and data communication products: SNA Network Architecture and Protocols, Communications Subsystems, Communications Network Management, Distributed Systems Software Technology, and Distributed Systems Organization. In this position his responsibilities include strategy planning, technical evaluation and guidance, technology transfer to IBM product development groups, and research personnel management and development. Dr. Moss received his B.S.E. degree in aerospace and mechanical sciences from Princeton University in 1971 and his S.M. and Ph.D. degrees in aeronautics and astronautics from the Massachusetts Institute of Technology in 1972 and 1977, respectively.

Robert A. Weingarten *IBM Corporate Headquarters, Old Orchard Road, Armonk, New York 10504.* Mr. Weingarten joined IBM in 1969 in the former IBM New York Development Center, where his assignment was on the OS/360 linkage editor. He joined the U.S. Army in 1970. Upon returning to IBM in 1972, he worked on the DOS RPG II compiler. In 1974, he transferred to Kingston, New York, where he was involved in various aspects of the definition of Systems Network Architecture, including high-level systems design, systems requirements gathering and planning, and system design management for the Advanced Communication Function access methods and communication network management. In 1982, he was the control program design and development manager in the scientific and engineering processor development area. Since March 1983, he has been assigned as a consultant on the Engineering, Programming, and Technology corporate staff, concentrating on communication programs. Mr. Weingarten received his B.S. and M.S. degrees in electrical engineering from New York University in 1967 and 1969, respectively.