The NBBS access node

by N. Budhiraja
M. Gopal
M. Gupta
E. A. Hervatic
S. J. Nadas
P. A. Stirpe
L. A. Tomek
D. C. Verma

Networking Broadband Services (NBBS) is an advanced wide-area-networking architecture that provides a rich set of services to support emerging applications for which quality of service and bandwidth reservation are critical, as well as legacy applications. The advanced features provided by the NBBS architecture require that each network node maintain a fully replicated topology database. As the size of the network increases, the network control costs at each network node become prohibitively high. The NBBS access node is a low-cost end node that introduces a hierarchical structure to the NBBS architecture, holding down network control costs while allowing the network to increase in size. The access node contains a subset of NBBS services. It is restricted to the periphery of the network and does not perform intermediate-node switching functions. It provides access to the NBBS backbone network by negotiating a relationship with an immediately attached network node (server network node) to obtain full NBBS services. In this paper, we discuss the requirements for the NBBS access node, examine the relationship between the access node and the server network node, and provide an overview of the access node architecture.

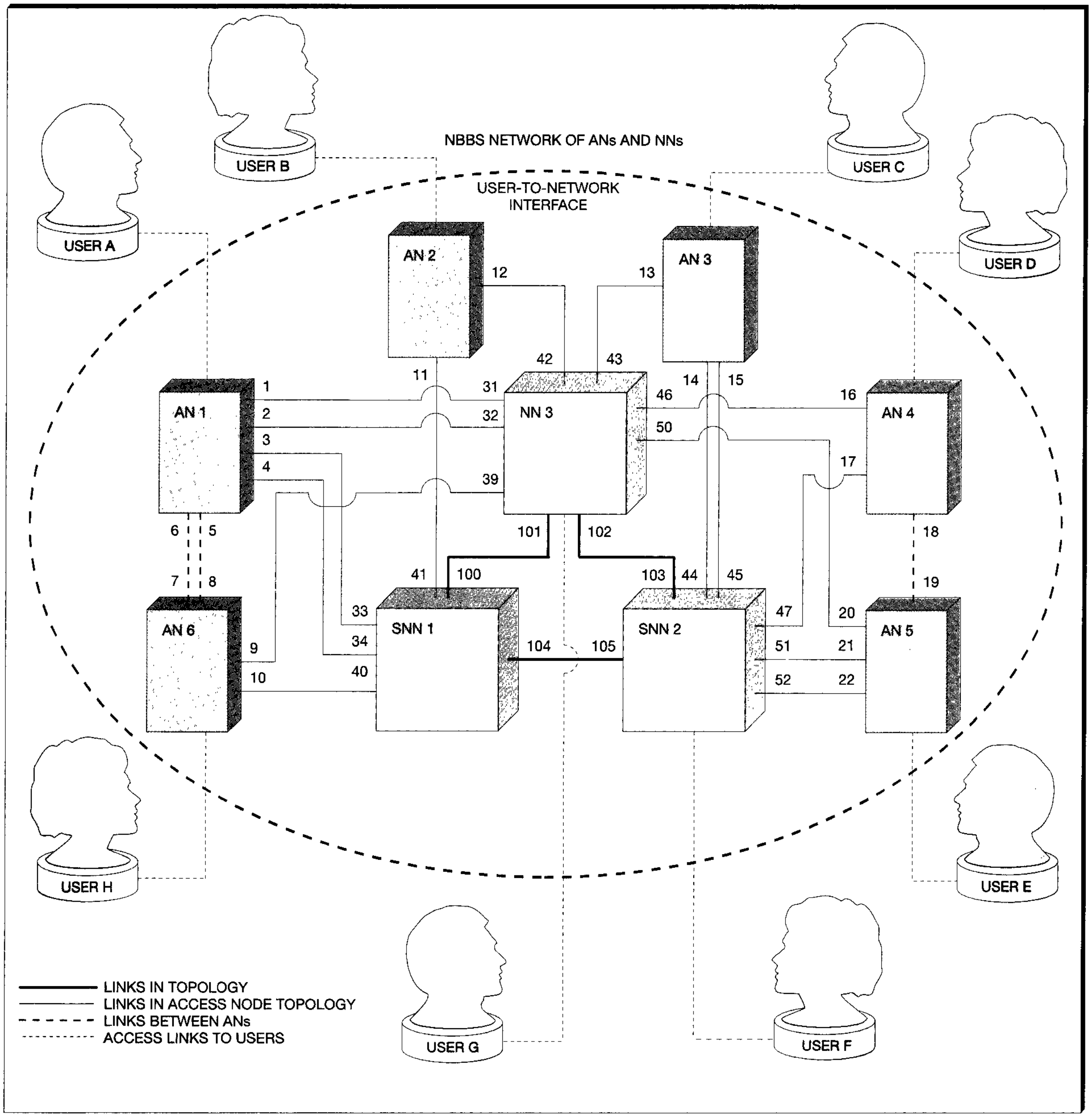
The Networking BroadBand Services (NBBS) architecture provides a rich set of advanced networking features to efficiently support emerging quality-of-service- and bandwidth-critical applications, as well as legacy applications. These features include bandwidth management for the efficient utilization of network resources, guaranteed quality of service (QOS) to support, for example, multimedia applications, set management to manage groups of users, nondisruptive path switching to reroute connections after failures, and connection preemption to insure that the higher priority connections receive service when network resources become scarce.[1,2]

Many of these functions rely on information contained in the topology database. This fully replicated database contains a description of all the *network nodes* (NNs) and the bandwidth-reservation and link-utilization state information of the entire network. As the number of NNs and links in an NBBS network increases, the network control costs associated with the utilization and maintenance of the topology database can become prohibitive. The costs include the network control traffic used to maintain replication of the topology database at each NN, the computational costs of path selection, and the storage requirements imposed on each NN.

A hierarchical approach is introduced that holds down the network control costs while allowing the NBBS network to increase in size. A new type of node called an *access node* (AN) is added to the NBBS architecture to implement this hierarchy. An AN is a reduced-cost node that contains only a subset of the NBBS services found in an NN. Those ser-

0018-8670/95/$3.00 © 1995 IBM

Figure 1 Example NBBS network with AN-NN links, AN-AN links, and NN-NN links



vices not resident in the AN are obtained from an adjacently attached NN called a *server network node* (SNN).

Figure 1 shows an example configuration of ANs in an NBBS network. Here the NBBS backbone consists of three NNs, two of which are SNNs. The six ANs reside at the periphery of the network and are attached to their respective SNNs. As we see here, an AN can be attached to more than one NN, via NBBS trunks, for increased availability and reliability. This allows the AN to reestablish network connections on an alternative path, in the event of a failure of a link between the AN and an NN. Furthermore, it gives the AN the potential to switch to another SNN in the event of loss of connectivity to its current SNN.

AN-to-AN one-hop connectivity is provided for point-to-point and point-to-multipoint connections.

The access node can be only the origin or the destination of a network connection.

In Figure 1, user A and user H can communicate over the one-hop link that connects the two adjacent ANs. This provides efficient utilization of network resources, avoiding the NBBS backbone network for one-hop AN-to-AN connections when capacity exists on the AN-AN link.

The AN architecture allows an NBBS network to become significantly larger (an order of magnitude more nodes) while holding down network control costs. The AN architecture permits large, geographically distributed networks by allowing ANs to remotely access their respective SNNs. Finally, the AN architecture permits ANs to attach to the NN backbone using lower speed links, if desired, in order to reduce network operation costs.

A primary requirement of the AN architecture is to provide *functional transparency* to the resources that are attached to the AN. In NBBS terminology a resource is an entity that utilizes the NBBS services at network end points, thus a user is an instance of a resource. The same access services are available to users whether they are attached to ANs or to NNs. The AN transparently provides these access services to its users by interacting with its SNN.

An AN contains a subset of the switching capabilities present in an NN. The AN can be only the origin or the destination of a network connection, and does not act as an intermediate node. This allows the hardware and software requirements of an AN to be substantially less than those of an NN, and allows the AN to provide low-cost access to the NBBS network.

The NBBS architecture is not the first to benefit by differentiating between end nodes and intermediate, or switching, nodes. Earlier IBM networking architectures introduced low-entry networking (LEN) end nodes (nonintermediate nodes) and Advanced Peer-to-Peer Networking* (APPN*) end nodes.[3]

Like an NBBS AN, an APPN end node depends on an adjacent node for network services. Once the network-node-to-server relationship is established, the APPN end node may request full network services. The role of the AN topology database is similar to that of the APPN local topology database. In both cases, each end node maintains topology information about directly connected links and nodes. ANs, however, cache remote AN topology information. The main difference is in the services that the underlying networks provide. APPN does not provide QOS guarantees, bandwidth reservation, or multicast services. It uses broadcasting for distribution of control information, rather than the spanning tree approach of NBBS.[1]

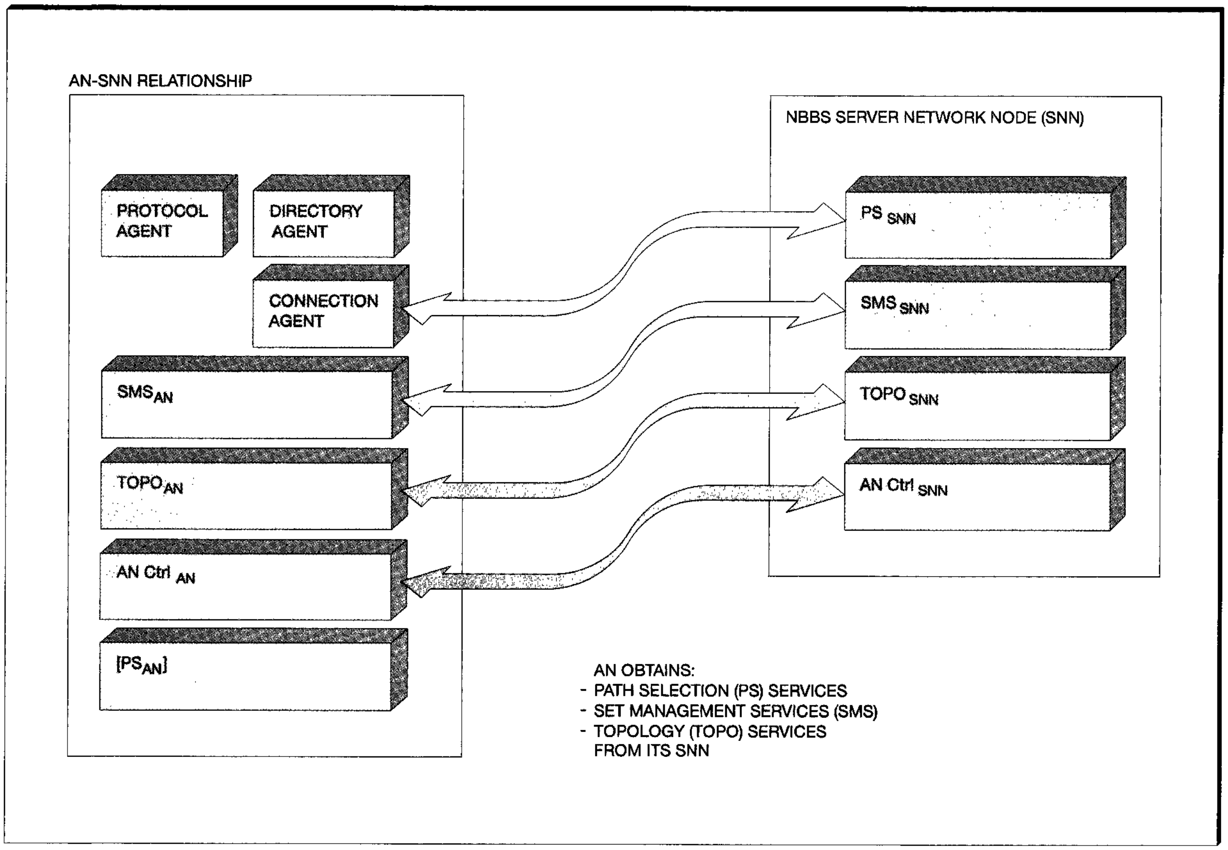
The LEN end node is less capable than either the APPN end node or the AN. It maintains topology information about adjacent links and nodes, but can only establish connections to resources that have been statically configured at the end node itself. The capability to request full network services makes the AN and the APPN end-node architectures significantly more powerful.

This paper is structured as follows. We begin with an overview of the NBBS services resident in an AN and those obtained from an adjacent NN. Next we explain how an AN obtains services from an adjacent NN. The NBBS services obtained by an AN through its SNN are then described: AN topology services, set management services, path selection services, and topology services. Finally, we discuss the costs involved with network control and show how the AN architecture reduces these costs.

Service placement

An NBBS NN provides three types of services: access services, transport services, and network control services. Access services[4] provide multipro-

Figure 2 ANs obtain network control services from SNNs



tocol support across the NBBS backbone network and maintain the negotiated user traffic rates (by policing and smoothing the traffic). Transport services[5] provide end-to-end data transport and are used by the access services to establish user connections. Network control services, comprised of topology,[5] set management,[6] and path selection services,[7] are used to manage NBBS network resources.

An AN provides only access services and transport services and obtains network control services from its SNN. The AN contains network control functions that operate in concert with their counterparts in the SNN, providing full, transparent NBBS network control services to AN users. Figure 2 illustrates the AN and corresponding SNN control functions. Before an AN can obtain network control services, it must establish a relationship with an adjacent NN. AN $\mathrm{Ctrl}_{\mathrm{AN}}$, in the AN, interacts with AN $\mathrm{Ctrl}_{\mathrm{SNN}}$, in the SNN, to establish and manage the AN-SNN relationship.

Although ideally any NN in the network would be able to act as an SNN, the NBBS architecture does not require it. For an NN to act as an SNN, enhancements are required to the topology, path selection, and set management services defined by the architecture for an NN. An AN must obtain topology (TOPO) and path selection services from its SNN. The $\mathrm{TOPO}_{\mathrm{AN}}$ function on the AN interacts with $\mathrm{TOPO}_{\mathrm{SNN}}$ on the SNN to provide topology services to AN resources. Furthermore, the AN obtains full set management services (SMS) from its SNN, although some subset of set management services may be provided by the AN as a result of previous interactions with the SNN; for example, $\mathrm{SMS}_{\mathrm{AN}}$ may maintain a cache of set membership information. The AN control functions provide transparent NBBS network control services to AN resources.

AN control services

In this section we describe the AN control function that establishes, validates, and terminates the AN relationship with the SNN. The AN retains responsibility for determining the server from which it receives its network control services. The AN determines whether its current SNN is providing adequate services, or whether it should select another adjacent NN to be its SNN. A mechanism is also provided for an AN to notify network management if it cannot find an SNN.

The architecture supports nodes with different implementations and therefore with different failure scenarios. The AN control function (AN $Ctrl_{AN}$) is a central place within an AN to package complex error-recovery actions. This permits the control functions residing in the AN, for example, SMS_{AN}, to support only relatively simple recovery actions, avoiding duplication of complex recovery code.

As part of AN initialization, AN $Ctrl_{AN}$ performs several functions. It locates a potential SNN, establishes and validates the AN-SNN relationship, and marks predefined network addresses utilized by the NBBS network control services. If unsuccessful, it notifies network management that the AN is isolated.

Establishing an AN-SNN relationship. The first responsibility of AN $Ctrl_{AN}$ is to find an adjacent NN that can provide network control services to the AN and to establish the *AN-SNN relationship*. As the AN activates links to adjacent NNs, it determines which of its adjacent NNs are *potential servers*, i.e., NNs with SNN capability.

The AN selects a potential server and sends a request for network control services. The potential server determines whether it can support the request. If it has SNN capability, it checks constraints supplied by network management for load balancing and insures that this AN name is unique across the network. If the potential server replies negatively, the AN selects another potential server and attempts again to establish the AN-SNN relationship.

When a potential server replies positively, the AN-SNN relationship is established. AN $Ctrl_{AN}$ then triggers the establishment of transport connections between peer network control functions in the AN and SNN, over the same link that was used to establish the AN-SNN relationship. These transport connections are used to communicate network control messages: path requests and replies, topology information, and directory query requests and replies. The AN and SNN mark addresses for SMS, TOPO, AN Ctrl, and path selection (PS) functions on the link utilized for AN control functions.

Validation of the AN-SNN relationship. NBBS control functions within an AN use transport connections to their counterparts in the SNN to obtain network control services. When one of these transport connections fails, the AN owning the transport connection attempts to reestablish it over the same link. In parallel, the AN validates that the server is still providing services to this AN. This is done by sending the SNN a message requesting network control services, using any link that connects the AN to the server. The difference between validation and establishment of an AN-SNN relationship lies in the way that the server replies to this request.

When a potential SNN determines that it can provide services to an AN, it checks whether it is already doing so. If it is, its positive reply message includes that additional information. The AN checks whether the server's reply validates the relationship. If so, and if the request for network control services was made on a different link than that used to establish the initial AN-SNN relationship, the AN triggers the establishment of transport connections to the server over the new link. The services are then available for use.

Terminating the AN-SNN relationship. The AN may determine that its SNN is no longer able to provide services, perhaps because of insufficient storage, and terminate the AN-SNN relationship and select another potential server. Network management may also determine that an SNN should no longer serve an AN or that an AN should terminate its current AN-SNN relationship.

If an AN attempts to establish an AN-SNN relationship with all of its potential servers and none agrees to be its SNN, the AN must notify network management of the problem. It sends an alert to any potential SNN, which relays the message to network management.[8]

AN topology

In an NBBS network, each NN maintains a consistent, replicated topology database that contains information about NNs and the *NN-NN links* that connect adjacent NNs. In an NBBS network that contains ANs, the complete topology is defined as the union of the NN topology, all the AN topology information, and the AN-AN links. As discussed earlier, the complete topology is too large to be stored and maintained in each NN. A key requirement of the AN architecture is that ANs are not burdened with NN topology information, and NNs are not significantly burdened with AN topology information. This is accomplished by not including AN topology information in the NN topology database, and not requiring each AN or NN to know the complete network topology. In this section the AN topology and the mechanisms that provide AN topology exchange are described.

In NBBS, a node maintains ownership of the *unidirectional links* that emanate from it to adjacent nodes.[6] Unidirectional links are logical links; two or more unidirectional links may share a common physical link. For example, in Figure 1, NN3 maintains ownership of unidirectional links 101 and 102. The topology database contains link state and bandwidth utilization information about SNN1, SNN2, NN3, and unidirectional links 100, 101, 102, 103, 104, and 105. In contrast, ANs consistently maintain only their own topology information.

An *AN topology* is defined as the combined information of the node identifier and characteristics for the AN itself and link state, bandwidth reservation, and utilization information for *AN-NN links* between an AN and its adjacent NNs. An AN owns the *AN-NN unidirectional links* that emanate from it to adjacent NNs and an NN owns the *NN-AN unidirectional links* that emanate from it to adjacent ANs. However, the AN topology includes the link state, bandwidth reservation, and link utilization information for NN-AN unidirectional links.

In Figure 1, the topology of AN1 consists of links 1, 31, 2, 32, 3, 33, 4, and 34, plus the NBBS node identifier of AN1 itself. Links 1, 2, 3, and 4 are AN-NN unidirectional links and links 31, 32, 33, and 34 are NN-AN unidirectional links. Links 6, 7, 5, and 8 are AN-AN links. AN1 owns and maintains unidirectional links 1, 2, 3, 4, 5, and 6. SNN1 owns unidirectional links 33 and 34; NN3 owns unidirectional links 31 and 32. The AN topology does not include AN-AN links, since AN-AN link information is always used for one-hop AN-to-AN connections and is not used by remote ANs.

ANs and NNs discover target AN topologies when their resources establish network connections with resources attached to these ANs. In particular, when a node attempts to locate a target resource that is attached to an AN (or a set of resources with one or more members attached to ANs), the AN topology is (or topologies are) provided. Origin and target AN topologies are used to compute end-to-end paths.

In order to inform and be informed about topology changes that potentially affect existing network connections, an AN needs a way to communicate its AN topology state to other nodes and to obtain topology information about other nodes.

- The AN requests topology monitoring services from its SNN to receive link state information about NN-AN unidirectional links in its AN topology.
- The AN receives utilization updates for NN-AN unidirectional links in its AN topology from the adjacent NNs owning the NN-AN unidirectional links.
- The AN requests topology distribution services from its SNN to forward link state, bandwidth reservation, and utilization information about its own AN-NN unidirectional links, on the control point (CP) spanning tree,[1] to communicate changes in its AN topology to other nodes that have established connections to this AN and to set leaders[7] whose membership contains one or more resources attached to this AN.
- The AN requests topology monitoring services from its SNN to allow its connection agent (CA) to monitor for link failure and for changes in the target AN topology.

Once a network connection is established to one or more target resources attached to an AN, the originating CA monitors not only the unidirectional links along the path(s), but also the AN topology of all target resources, to keep abreast of target AN topology changes. This allows the originating CA to perform fast nondisruptive path switching (NDPS), using recent target AN topology information, when a target AN link along the path fails. Additionally, the target AN supports fast NDPS by optionally sending bandwidth reservation and link utilization information about its AN topology on the

CP spanning tree. Receiving the AN-NN bandwidth reservation and link utilization information allows the originating CA to provide to path selection services the most recent utilization and configuration information.

The network control messages that contain AN topology are:

- Explicit query reply. If the target resource is attached to an AN, the AN topology is returned to the node in the explicit query reply.
- Implicit query reply. If the target resource is attached to an AN, the AN topology is returned to the origin node in the connection setup reply.
- Join request. If a resource attached to an AN joins an open set, the AN topology is included in the join request.
- Join remote resource reply. If the target of a join remote resource request is a resource attached to an AN, the AN topology is included in the join remote resource reply.
- Query set information reply. If a set member sends a query set information request, and one or more set members are attached to ANs, the query set information reply includes the topology for each attached AN.
- Path selection services. When an AN requests that a path computation be performed by the path selection services in the SNN, the AN must include the source AN topology, and if the target resource is attached to an AN, the target AN topology must also be included. An AN may include link state and utilization information for AN-AN links to allow the SNN to consider AN-AN links when computing a one-hop path.

Topology services

In this section we describe the topology distribution and monitoring services provided by the AN topology function. $TOPO_{AN}$ provides two kinds of services that parallel the services provided by $TOPO_{SNN}$:

1. Distribution of local AN topology to the remote NNs and ANs
2. Monitoring of links, remote NNs, and AN topologies for the various functions on the local AN

$TOPO_{AN}$ interacts with $TOPO_{SNN}$ to accomplish the above two functions. When AN $Ctrl_{AN}$ indicates to $TOPO_{AN}$ that a server relationship has been established with a neighboring NN, $TOPO_{AN}$ establishes a network connection with $TOPO_{SNN}$ to provide the above services. The following sections discuss these services in more detail.

Distribution services. Whenever the state of any of the links that are owned by an AN changes, $TOPO_{AN}$ sends the topology update to $TOPO_{SNN}$ and also to the $TOPO_{AN}$ functions on all the adjacent ANs. $TOPO_{SNN}$ then reliably broadcasts[1] the update on the CP spanning tree.

$TOPO_{AN}$ also provides the local topology to various AN control functions, whenever needed. These include the connection agent (CA), the directory agent (DA), and set management services. As described earlier, when the CA sends a path computation request to PS_{SNN}, it includes up-to-date local AN topology, which it received from $TOPO_{AN}$. Similarly, the DA requires up-to-date topology to respond to a directory query, and SMS_{AN} needs to include this topology on all join requests that it sends to SMS_{SNN}.

The local topology provided by $TOPO_{AN}$ to the control functions contains state information about local links (e.g., AN-NN unidirectional links) and their link duals (the corresponding NN-AN unidirectional links). Clearly, $TOPO_{AN}$ has up-to-date information about its local links from the link management services. As we describe next, $TOPO_{AN}$ gets information about the corresponding NN-AN unidirectional links from its SNN.

Monitoring services. In addition to the distribution services described above, $TOPO_{AN}$ monitors links and nodes and remote AN topologies on behalf of the control functions on the local AN. For example, the CA needs to monitor all the links over which it has initiated connections. Similarly, the DA needs to monitor the network for access agents that utilize the CP spanning tree rather than default distribution trees[6] for sending undirected explicit queries. By knowing which type of multicast tree a given directory group utilizes, a DA can determine which multicast tree is required to initiate an undirected explicit query to locate resources that may reside within a group of DAs.

Whenever the CA or DA wishes to monitor the topology of a remote AN, it registers interest in the remote AN topology with the local $TOPO_{AN}$. The $TOPO_{AN}$ forwards the monitoring request to $TOPO_{SNN}$, which then monitors the remote AN topology. When $TOPO_{SNN}$ receives a topology update about which some TOPO_{AN} has registered interest, it forwards the update to the interested TOPO_{AN}, which then forwards it to the local CA or DA, as appropriate. Since the SNN does not store the update (it only forwards it), the size of the topology at the SNN grows only minimally as the network size increases.

Updates for links are obtained in a similar manner. TOPO_{AN} registers interest in these links with TOPO_{SNN}, and whenever the SNN receives an update about these links (over the CP spanning tree), TOPO_{SNN} forwards the update to the interested TOPO_{AN}.

Set management services

In this section we discuss how set management services (SMS)[6] are provided to users attached to an AN. As described earlier, an AN obtains set management services from its SNN.

Set management services in the server network node. SMS_{SNN} is responsible for providing the following services to users attached directly to its node as well as to users attached to any ANs for which it is the SNN.

- Set join service. A user may ask its set manager to join an open set or it may be asked to join an open or closed set by another user who is already a set member.
- Add to set service. A set member at the node where the set leader is located may ask the set leader to add a remote user to the set.
- Closed set creation service. A user may ask its set manager to create a closed set, owned by the requesting user.
- Set leave service. A set member may leave an open or closed set.
- Set member removal service. The owning user of a closed set may request that a set member be removed.
- Set destruction service. The owning user may request that a closed set be destroyed.
- Query set information service. A user may query the members and the cardinality of a set.
- Query default distribution tree information service. A user may query access information about the default distribution tree of a set.

When a service request is received from an AN, SMS_{SNN} performs the necessary protocols on behalf of the AN. For example, when a user attached to an AN wishes to join an open set, the AN sends SMS_{SNN} a join set request message. This message identifies the requesting user and the set to be joined. When SMS_{SNN} receives the join set request message, it locates the set leader and forwards to it the join request message. Upon receiving the join request message, the set leader adds the requesting user to its set membership list and sends a join reply message back to SMS_{SNN}. Upon receiving the join reply, SMS_{SNN} adds the new set member to its local set membership list and informs SMS_{AN} that the join set request was successful.

Set management services in the access node. Although the SNN performs all of the set management services, an AN is required to maintain some set information. SMS_{AN} maintains a local set membership list containing sets for which its users are members. It also maintains a local cache of target sets containing membership and default distribution tree information.

By maintaining a local set membership list, an AN can support multicast network connections without interacting with its SNN. By maintaining a cache of target set information, the AN can obtain this information without communicating with its SNN and the SNN does not have to cache this information.

Path selection services

In this section we discuss how access nodes were incorporated into the path computation services.[7] As we described earlier, path computation services for an AN are provided by an SNN. The SNN, however, has information about the topology of network nodes only. In order to incorporate the AN topology of the source and destination, the path computation algorithm was enhanced.

When an AN needs to determine a path, it sends a path request message to the PS component in its SNN. This message contains the destination resource(s), the type of path required (e.g., point-to-point or point-to-multipoint), the QOS parameters, the local AN topology, and possibly the destination AN topology. PS_{SNN} computes an end-to-end path or a tree[6] and returns the computed path to the AN. The connection agent uses the path to send the connection setup messages.

Enhancements to path selection services. The source or destination of a network connection (or both) may be attached to an AN. If so, the path selection algorithm needs to include the appropriate AN topology (or topologies) in its computation. AN topology information is provided by the connection agent for the path computation algorithm.

When a path through the NN topology is required, path computation must be performed at the SNN. When the network provider uses policy-based routing, the paths must be computed by the SNN. Even though a path may exist between two adjacent ANs, the network policies may prohibit the use of it; for example, the link may not meet security needs of the connection. The cost of maintaining network topology in the AN for path selection and the expectation of more powerful processing capability in the server also argue for path computations to be performed in the server.

The access node architecture reduces the amount of bandwidth required in several ways.

When the connection is between two adjacent ANs and the network provider does not enforce policy-based routing constraints, it may be more efficient for an AN itself to compute the path, rather than sending a path request to its SNN. If policy-based routing is not required, when a path computation is requested the AN checks to see if it is for a connection between two adjacent ANs. If so, and a path with sufficient bandwidth is available, it is selected locally. Otherwise, the path computation request is forwarded to the SNN.

This enhancement of the path computation services reduces the load on the SNN and results in a faster response time for path computation requests for network connections between adjacent ANs.

Network control costs

The network control costs of an NBBS network include bandwidth utilization for network control messages, storage requirements for network state information, and topology-dependent processing requirements for path computations. The network state information, which includes the network configuration as well as the status and capability of different network entities, is stored in the topology database, maintained at each NBBS NN. Each NN executes a distributed algorithm that utilizes a logical CP spanning tree to exchange topology, link state information, and bandwidth reservation and utilization information. The CP spanning tree, a multipoint-to-multipoint connection that interconnects all NNs in an NBBS network, is utilized to efficiently distribute control information.[1]

The first network control cost of particular concern in a large network is the bandwidth requirement for network control messages. Bandwidth is a costly commodity in wide area networks and should be used conservatively for network control traffic. For example, we estimate that in an NBBS network containing 100 NNs, with each NN supporting 5 T1 (1.54 million bits per second) trunks connected to other NNs, the bandwidth utilized on a given CP spanning tree link just to exchange bandwidth reservation and link utilization information is 1.3 percent of the total trunk bandwidth. A network containing 1000 NNs similarly connected would require 13 percent of the T1 trunk bandwidth to exchange bandwidth reservation and link utilization information. Bandwidth reservation and link utilization information messages are sent by each NN periodically or when thresholds are crossed by changes in reserved bandwidth.

The AN architecture reduces the amount of bandwidth required to support users attached to an AN in several ways. First, the bandwidth utilized on the CP spanning tree links is reduced. AN link state information is sent via the CP spanning tree in the same manner as state information for links between NNs. However, bandwidth reservation and utilization messages for AN-to-NN or AN-to-AN links are not carried on the CP spanning tree. Instead, the bandwidth reservation and utilization information on AN-to-NN links is piggybacked onto other control messages so that it is selectively available, on a best-effort basis, to ANs with interest in the information. A second way in which the AN architecture reduces the bandwidth requirement for control traffic involves one-hop connections. ANs that are directly connected to each other can establish one-hop point-to-point network connections not routed through the NN backbone, reducing the load on the links that connect NNs. Finally, the AN architecture permits ANs to be attached to the NN backbone using lower speed links, thus reducing network operating costs for NBBS connectivity to lower capacity, geographically distributed locations. To that end, the AN architecture reduces the amount of NN backbone control traffic that is carried on AN-NN and NN-AN links by selectively obtaining key services from an SNN. For example, the NN topology database is maintained at the SNN. The network control traffic utilized to maintain the fully replicated NN topology database is not carried on the AN-NN and NN-AN links.

The second network control cost that must be addressed is the storage capacity of each NN. The size of the topology database grows linearly as the size of the network increases (either by the addition of links, nodes, or subnodes). Since the topology database is maintained in each NN, the size of the network can be constrained by the storage capacity of each NN. The AN architecture allows the AN topology, which is the network information about an AN and the links that connect it to adjacent NNs, to be transmitted and stored separately from the topology database of NNs. Topology information for a given AN is provided and maintained for each NN or AN with a network connection to it. The other NNs and ANs in the network are not required to store that AN topology information. This significantly reduces the memory requirements of ANs and NNs in the network, thus facilitating a larger NBBS network size.

The third network control cost to be considered is the processing requirement on each NN. The NBBS path selection algorithm, which computes the source-to-target route of network connections, is heavily dependent on network topology size and has stringent performance requirements. The path selection algorithm is an efficient graph-search algorithm based on a modified version of the Bellman-Ford algorithm.[9] Although this algorithm has been optimized in the NBBS architecture, its complexity is dependent on the number of nodes and links in the network. The AN architecture reduces the processing requirements for path selection by reducing the size of the network on which the path selection algorithm operates. That size is proportional to the number of NNs in the network and the links that connect them. ANs are periphery nodes, permitted to be origins and destinations of network connections. If a path computation is requested that has a source or destination attached to an AN, that AN topology is provided during the path computation. Since a network that efficiently uses ANs may contain an order of magnitude more ANs than NNs, the computational savings in path selection can be substantial.

Summary

The NBBS AN architecture is a natural hierarchical extension to NBBS. The AN architecture permits large, geographically distributed NBBS networks. Larger networks are feasible as a result of localizing AN topology information at the AN itself, reducing the flow of bandwidth reservation and link utilization information in the network, and allowing each AN to remotely access its SNN. The AN contains a subset of the services provided by an NBBS NN. The additional NBBS services necessary to provide full, transparent NBBS functionality are obtained at an adjacent SNN. Such services include end-to-end bandwidth and quality-of-service guarantees, multicast group management, and efficient bandwidth utilization.

Acknowledgments

The authors would like to acknowledge John Drake for his technical contributions to the AN architecture. Furthermore, we thank Gerald Lebizay, Denis Chevalier, and Eric Levy-abegnoli for their contributions toward the initial AN design direction. Rachel Bodner and Jean-Paul Chobert contributed their NBBS topology expertise to the AN topology.

Cited references

1. *Networking BroadBand Services (NBBS) Architecture Tutorial*, GG24-4486-00, IBM Corporation (1995); available through IBM branch offices.
2. G. A. Marin, C. P. Immanuel, P. F. Chimento, and I. S. Gopal, "Overview of the NBBS Architecture," *IBM Systems Journal* **34**, No. 4, 564–589 (1995, this issue).
3. *APPN Architecture and Product Implementation Tutorial*, GG24-3669-02, IBM Corporation; available through IBM branch offices.
4. C. P. Immanuel, G. M. Kump, H. J. Sandick, D. A. Sinicrope, and K. V. Vu, "Access Services for the Networking BroadBand Services Architecture," *IBM Systems Journal* **34**, No. 4, 659–671 (1995, this issue).
5. M. Peyravian, R. Bodner, C.-S. Chow, and M. Kaplan, "Efficient Transport and Distribution of Network Control Information in NBBS," *IBM Systems Journal* **34**, No. 4, 640–658 (1995, this issue).
6. N. Budhiraja, M. Gopal, M. Gupta, E. A. Hervatic, S. J. Nadas, and P. A. Stirpe, "Multicast Network Connection

Architecture," *IBM Systems Journal* **34**, No. 4, 590–603 (1995, this issue).
7. T. E. Tedijanto, R. O. Onvural, D. C. Verma, L. Gün, and R. A. Guérin, "NBBS Path Selection Framework," *IBM Systems Journal* **34**, No. 4, 629–639 (1995, this issue).
8. S. A. Owen, "NBBS Network Management," *IBM Systems Journal* **34**, No. 4, 725–750 (1995, this issue).
9. D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Inc., Englewood Cliffs, NJ (1987).

Accepted for publication July 6, 1995.

Dinesh C. Verma *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: verma@watson.ibm.com).* Dr. Verma is a research staff member in the Advanced Networking Laboratory. He received his B.Tech. in computer science from the Indian Institute of Technology at Kanpur, India in 1987, and his Ph.D. from the University of California at Berkeley in 1991. He is interested in network control architecture and software for high-speed networks, performance analysis, and multimedia applications.

Lorrie A. Tomek *IBM Networking Hardware Division, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (electronic mail: tomek@vnet.ibm.com).* Ms. Tomek is employed in IBM's Networking Architecture group. She received her B.S. in computer science and mathematics at the State University of New York at Binghamton. She received her M.S. degree in computer science at the University of North Carolina at Charlotte, North Carolina. She is currently a Ph.D. candidate at Duke University in Durham, North Carolina with research interests in performance and reliability analysis of fault-tolerant systems.

Paul A. Stirpe *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: stirpe@watson.ibm.com).* Dr. Stirpe has been an IBM employee for the past ten years, with work assignments in Endicott, New York, Santo Palomba, Italy, and most recently as a research staff member in the Advanced Networking Laboratory. He received his B.S. in electrical engineering from the State University of New York at Buffalo in 1985, his M.S. in computer engineering from Syracuse University in 1989, and his Ph.D. in computer science from Boston University in 1992. His research interests include performance analysis of communication networks, high-speed network architecture, and the enablement of multimedia applications on wide area networks.

Stephen J. Nadas *IBM Networking Hardware Division, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (electronic mail: steve_nadas@vnet.ibm. com).* Mr. Nadas is an advisory engineer in Networking Architecture. He joined IBM in 1978 at the East Fishkill facility and has worked in the Myers Corners and Mid-Hudson Valley development laboratories. Mr. Nadas has been a performance analyst and benchmark developer for large systems and a microcode designer in Future Processor Development in Poughkeepsie. Mr. Nadas received his B.S. in mathematics from the State University of New York at Albany in 1976, and his M.S. in computer engineering from Syracuse University in 1994.

Elizabeth A. Hervatic *IBM Networking Hardware Division, P.O. Box 12195, Research Triangle Park, North Carolina 27709 (electronic mail: hervatic@ralvm6.vnet.ibm.com).* Ms. Hervatic is an advisory programmer in ATM Market Development. She joined IBM in 1989 and has worked in Networking Architecture. Among other positions, Ms. Hervatic has been the technical leader of the NBBS base architecture team. Ms. Hervatic received her B.S. in computer science from North Carolina State University in 1989.

Manish Gupta *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: guptama@watson.ibm.com).* Mr. Gupta has been working in the Advanced Networking Laboratory since November 1992. He received his B.Tech. in computer science and engineering from the Institute of Technology, Banaras Hindu University, Varanasi, India, and his M.S. in computer science from the University of Kentucky in 1992. His research interests include high-speed network architecture and distributed systems.

Madan Gopal *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532.* Dr. Gopal joined the IBM Thomas J. Watson Research Center as a research staff member in 1985 and was a member of the Advanced Networking Laboratory until his departure in 1995. He received his master's and doctoral degrees in computer science from the University of Waterloo, Canada, in 1980 and 1985, respectively. Dr. Gopal's interests are in the area of platform-independent ATM switch software. He has published more than 20 papers in various journals and conference proceedings in the area of design, analysis, and performance modeling of network protocols. He is an active member of the Institute of Electrical and Electronics Engineers (IEEE).

Navin Budhiraja *IBM Research Division, Thomas J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, New York 10532 (electronic mail: navin@watson.ibm.com).* Dr. Budhiraja is a research staff member in the Advanced Networking Laboratory. Dr. Budhiraja received his B.S. in computer science from the Indian Institute of Technology at Kanpur, India in 1988, and his M.S. and Ph.D. in computer science from Cornell University in 1991 and 1993, respectively. Current research interests include networking, distributed systems, and fault tolerance.

Reprint Order No. G321-5589.