Dynamic Burst Transfer Time-Slot-Base Network

Kohei Shiomoto and Naoaki Yamanaka, NTT Network Service Systems Laboratories

ABSTRACT This article proposes a new high-speed network architecture called dynamic burst transfer time-slot-base network (DBTN). The DBTN network is based on circuit-switched network technology. A routing tag is attached to a burst at an ingress edge node and the burst is self-routed in a DBTN network, the circuit of which is created dynamically by the routing tag. The routing tag, which is called time-slots-relay, consists of link identifiers from the ingress to the egress nodes and is used to create the circuit. Subsequent data is switched over the circuit being created in an on-the-fly fashion. Each link identifier is loaded into the address control memory (ACM) of each circuit switching node, and thereby the circuit to the destination is created dynamically. Subsequent user data follows immediately after the time-slots-relay and is sent over the established circuit. Thus short-lived fairly large data transfers such as WWW traffic are efficiently carried. A circuit between adjacent nodes is created and released dynamically so bandwidth efficiency is improved compared with conventional circuit-switched networks. Time division multiplexing of the circuit-switched network is utilized so there is no delay jitter or loss within a burst. We address the performance of DBTN switches and report the experimental system.

Internet traffic continues to grow exponentially [1, 2]. The World Wide Web (WWW) is one of the most popular applications of the Internet, and it has been reported that the volume of WWW traffic is 65 to 85 percent of the total Internet backbone traffic [3]. WWW traffic is characterized by large amounts of data transfer and short response time requirements.

Several high-speed networking technologies have so far been proposed such as burst switching [4, 5], fast circuit switching [6], and asynchronous transfer mode (ATM) [7]. However, these technologies are not well suited for the above-mentioned traffic because they are connection-oriented. The session duration of this kind of traffic is too short to carry using a connection-oriented protocol because it is wasteful to set up a connection before sending data. In addition, because circuit switching allocates static bandwidth to each connection, bandwidth efficiency degrades. Even though ATM carries bursty traffic efficiently, cell transfer delay jitter and cell loss may occur in bursty data. (In addition, because ATM is connection-oriented, setup delay is not efficient for short-lived sessions.)

In this article we propose a new communication network architecture called *dynamic burst transfer time-slot-base network* (DBTN). This is based on a burst-by-burst circuit setup in a circuit-switched network. A routing tag is attached to a burst at an ingress edge node and the burst is self-routed in a DBTN network, the circuit of which is created dynamically by the routing tag. The routing tag, which is called time-slots-relay, is a series of link identifiers to the destination. The routing tag is resolved from the network address for the destination at the edge node. The time-slots-relay consists of link identifiers from the ingress to the egress nodes and is used to create the circuit. Subsequent data is switched over the circuit being created in an on-the-fly fashion. A link identifier in the *time-slots-relay* is unique within each node. Each link identifier is loaded into the address control memory (ACM) of each circuit-switched transit node, and thereby the circuit to the destination is created dynamically. Subsequent user data follows immediately after the time-slots-relay and is sent over the established circuit. Thus, short-lived fairly large data transfers such as WWW traffic are efficiently carried.

Table 1 compares DBTN and conventional transport techniques such as circuit switching, packet switching (datagram), and ATM. In the table, we address the following items:

- Space overhead.
- Efficiency for bursty data transport.
- Quality of service support.
- Effectiveness for short-lived sessions.

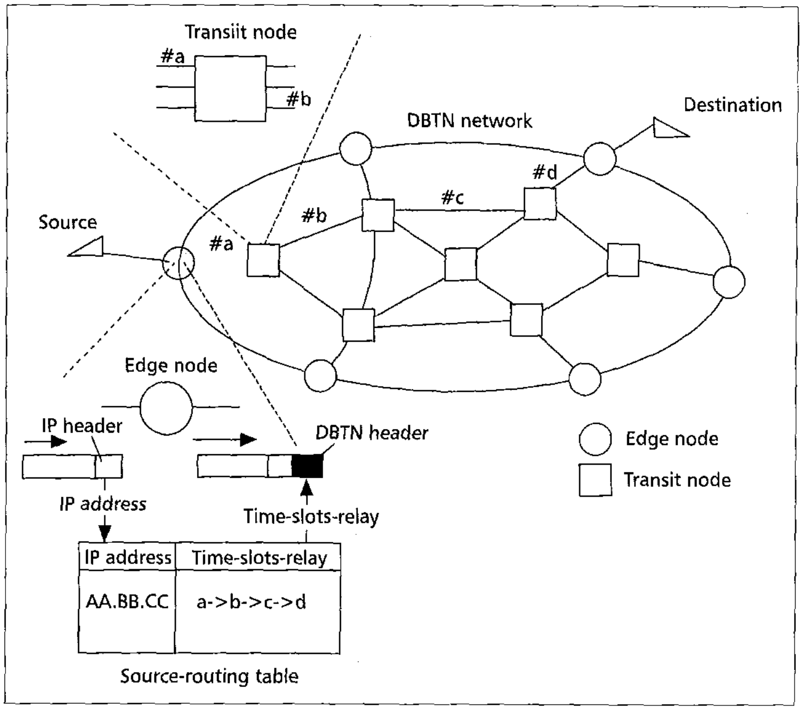
Space Overhead — The space overhead is low for DBTN and for circuit switching, because they use time division switching, in which switching is performed by exchanging time-slots between input and output links, so *no packet header* is needed. In contrast, the space overhead for ATM amounts to about 10 percent (=100 x 5/53) even if a continuous stream is transported. Taking into account packet length distribution, the space overhead of ATM, which is also known as cell tax, amounts to 25 percent [2].

Bursty Data Transport — DBTN, packet switching, and ATM efficiently carry bursty traffic. In contrast, circuit switching does not because it holds bandwidth for a connection while there is no data to send. In a DBTN network, connections are created and released dynamically on a burst-by-burst basis. A connection is established only when a packet is sent. In contrast, circuit switching holds a connection whether a packet is present or not, so the bandwidth efficiency is low for bursty traffic.

Quality of Service — DBTN and circuit switching provide good quality of service because they are based on time division multiplexing. In contrast, ATM and packet switching are based on statistical multiplexing, so we cannot avoid cell/packet loss and transfer delay jitter.

	Circuit switching	Packet switching	ATM	DBTN
Space overhead	x			x
Bursty traffic		x	x	x
Quality of service	x			x
Short-lived session		x		x

Table 1. *Comparison of DBTN with conventional methods.*



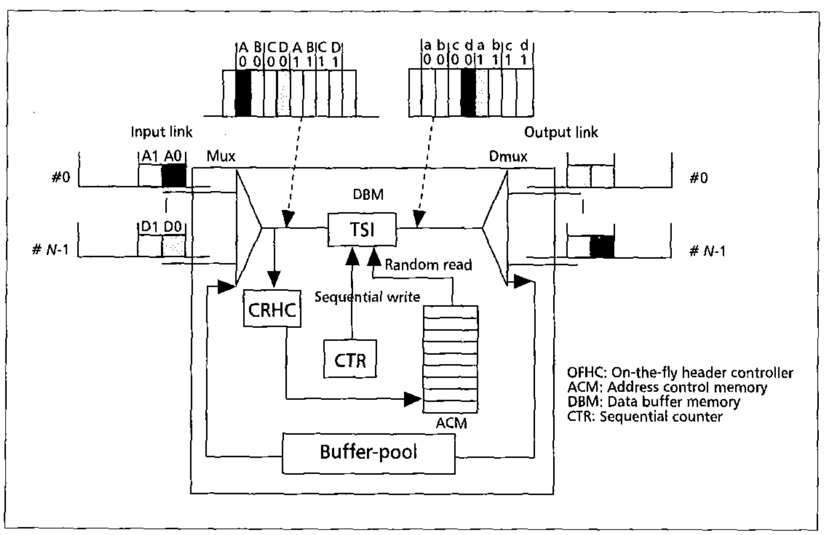
■ **Figure 1.** *The DBTN concept.*

Short-Lived Sessions — DBTN and packet switching are suited for short-lived fairly large data transfer such as WWW traffic because they do not require a pre-established connection. By contrast, ATM and circuit switching require control delay and round trip time for setting up the connection.

Figure 1 shows the concept of DBTN. A DBTN network consists of edge nodes and transit nodes. Each edge node caches the time-slots-relay associated with a destination address. When a source edge node receives a packet, it resolves the time-slots-relay from the network address for the destination and then attaches the time-slots-relay to the packet and injects it into the network. Connection is set up while the time-slots-relay is en route to the destination. Subsequent data follows immediately after the time-slots-relay.

Each node is an STM switch, which translates the link identifier in the time-slots-relay, determines appropriate output time-slots, and configures a connection by setting the association between input and output time-slots into an address control memory (ACM). Thus the connection is configured at the transit node. This procedure takes place at every transit node between the source and destination edge nodes.

In this article, we address the design and the performance of DBTN switches and report the experimental implementation system. The rest of the article is organized as follows. In the second section, we describe the design of the DBTN switch. In the third section, we investigate its performance. The fourth section reports the experimental DBTN switching node system. In the fifth section, we overview related work and compare it with work on DBTN. The last section summarizes the article.



■ **Figure 2.** *The DBTN switching node.*

DBTN Switches

Principle behind DBTN Switches

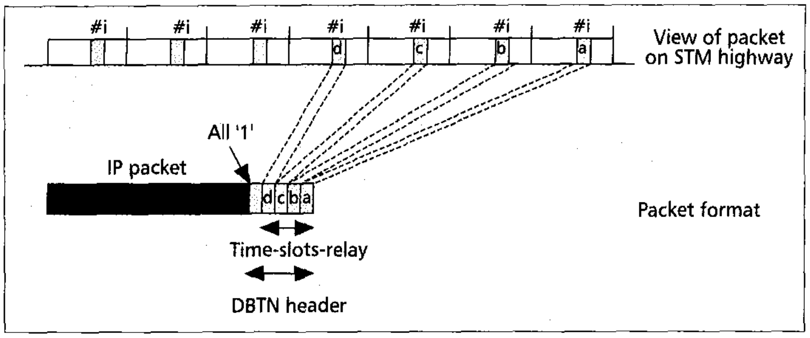
The DBTN switch is based on the conventional STM switch. It sets the address control memory (ACM) and creates the circuit dynamically by using the time-slot-relay in the burst header. To this end, the DBTN employs mechanisms for:

- Delineating the burst.
- Configuring the ACM.

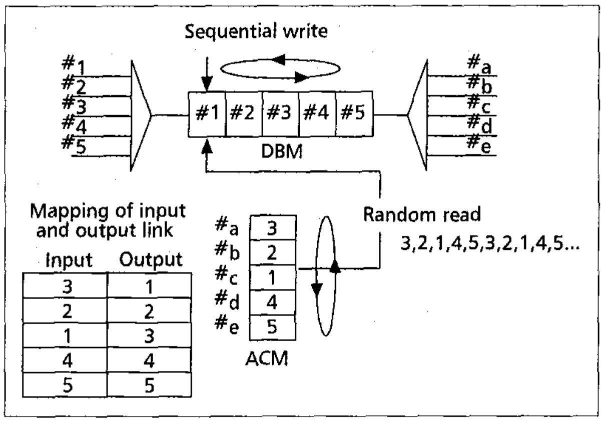
Below we give an overview of the DBTN switch and describe those mechanisms.

Figure 2 shows an $N \times N$ switch fabric of a transit node. Time slots within a frame over the highway are exchanged by a time-slot interchanger (TSI). By exchanging time-slots, a circuit is switched. The TSI is implemented by using a data buffer memory (DBM) and an address control memory (ACM). The time-slots-relay, which is a part of the DBTN header, is carried in the time slot associated with the input link. The link identifier in the time-slots-relay is loaded into the ACM and the association between the input and output links is made. The output link is designated by the link identifier in the time-slots-relay.

We need a packet delineation mechanism because the time-slots-relay is carried in-band. The packet delineation mechanism and packet format are detailed below. The packet format used in DBTN networks is shown in Fig. 3. Because a time-slots-relay is transferred in an in-band fashion in DBTN networks, a packet delineation mechanism is needed. For



■ **Figure 3.** *Packet format.*



■ **Figure 4.** *An STM switch.*

this purpose, a preamble pattern is attached to the front of a packet. The time-slots-relay field follows the preamble pattern. If a preamble pattern is detected at a node, the time-slots-relay field is extracted. The length of the time-slots-relay field is variable, so its last time slot is marked by all "1's." When an end-of-packet pattern is detected by the presence of an idle pattern, which is sent out by the source terminal, an ACM entry is cleared.

In conventional STM switches, the circuit between the input and the output links is made by loading the time slot number of an input link into the ACM entry associated with the time slot number of the output link. In Fig. 4, the circuit between the input link 1 and the output link 3 is set up. In conventional STM networks, the contents of ACM are configured via an out-of-band setup signaling message. This takes several seconds because of the complicated software control to set up at each node. In contrast, in DBTN networks, the ACM contents are configured via an in-band message, i.e., the time-slots-relay. The time-slot corresponding to the node in the time-slots-relay is extracted. This time slot indicates the destination output link identifier of the switch. Suppose that the time slot stands for the a-th output link. The switch finds an idle time slot associated with the output link and then sets the time-slot number for the input link into the a-th entry of the ACM.

Time Slots-Relay Cache

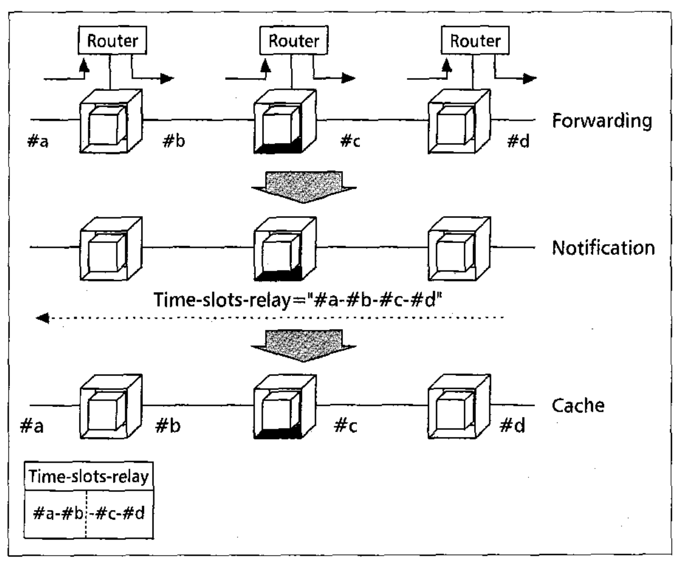
In DBTN networks, however, the source edge node resolves the time-slots-relay from the network address for the destination. We resolve the time-slots-relay from the destination address in the packet header. The association between the destination address and the time-slots-relay is kept in a cache table at the source edge node.

In some cases, however, the destination terminal address might not be found in the table due to limitations of the cache table. To cope with this case, we must perform forwarding, notification, and caching, as shown in Fig. 5. When we encounter a miss-cache, the packet is forwarded to the next node in a *forwarding* mode. Each node determines the next node by consulting the routing table with the network address for the destination like IP routers. An output link identifier at each node is collected as the packet is forwarded to the destination edge node. Once the packet reaches the destination edge node, the time-slots-relay, which consists of the output link identifier from the ingress to the egress nodes, is sent back to the source edge node by traversing the time-slots-relay backward. The time-slots-relay is stored in the cache table at the source edge node. The cached time-slots-relay is used for the subsequent burst transmission.

Buffer-Pool

Because burst data follows immediately after the routing information is sent in-band, if sufficient time slots are not available, the circuit cannot be established and the subsequent burst data are lost. To avoid loss of burst data, we introduce a *buffer-pool.* When sufficient time slots are not available at the destination output link, the connection request is blocked and diverted to the buffer pool. This buffer pool is connected with the STM switch fabric via input and output links as shown in Fig. 2. The buffer pool is shared among all the output links.

The diverted connection is queued in a packet-by-packet fashion. The connection setup request is retried after a back-off timer expires. The controller of the buffer pool manages individual queues for each diverted burst. On receiving the diverted burst, it starts the timer associated with the burst. The timer expiration time is randomly determined. When the timer expires, the associated burst is selected for the retransmission.

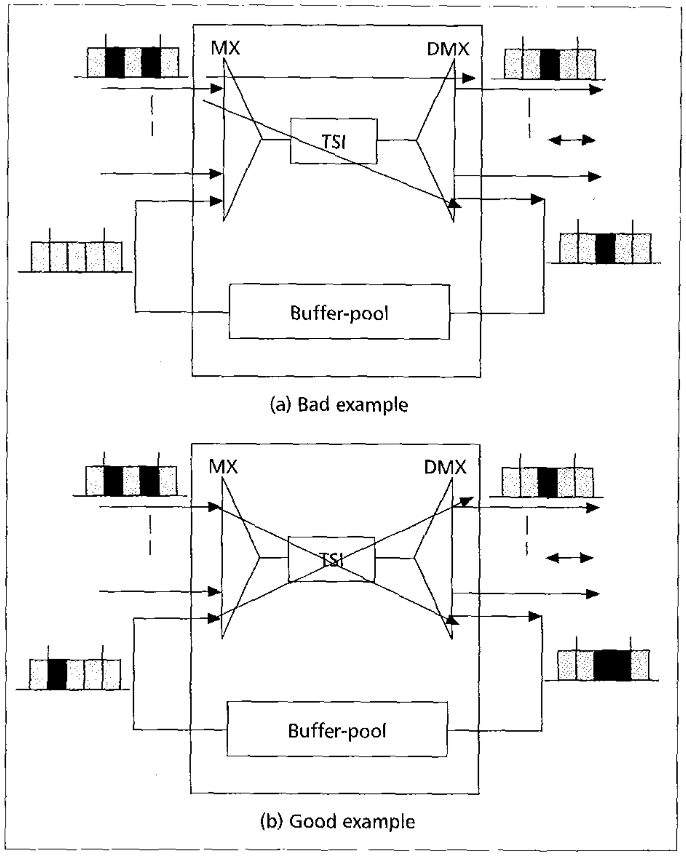


■ **Figure 5.** *Time-slots-relay cache.*

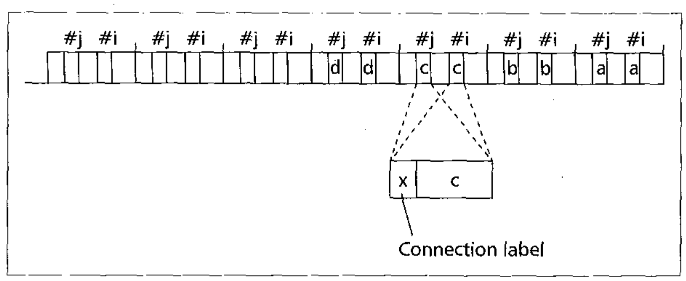
The retry request is treated in the same way as other newly arriving requests at the output port. In this way, the communication overhead between the buffer pool controller and all the output links is eliminated. The advantage of this retry mechanism is that it isolates the buffer-pool management from the time-slots manager. We might maximize time-slot utilization if we could allocate idle time slots *immediately after* they were freed. To do so, the buffer-pool controller would need to manage the bandwidth of all the output links of the switch. A communication mechanism between the buffer pool and the time-slots manager would be required, which would not be scalable as the number of switch output ports grew. A problem may arise due to the high communication overhead as the switch size grew.

Multiple Time-Slots Rate

How to implement a connection requiring multiple time slots is detailed below. In DBTN networks, a variable transmission rate is implemented by using multiple time slots within a frame. The time-slots-relay should be handled in such a way that all time slots associated with a connection are assigned to the same output link. If some time slots were partially assigned to the buffer-pool link while the others were assigned to the output link, such a connection would be meaningless due to a lack of order integrity of the time slots, as shown in Fig. 6. To associate multiple time slots with the connection, a connection identifier sub-field is introduced in a time-slot field of the time-slots-relay, as shown in Fig. 7. The connection identifier sub-field has only to be unique within an input link and an output link. We may change its value hop-by-hop for it to be unique within a link like a VCI.



■ **Figure 6.** *Problem in multiple time-slots connections.*



■ **Figure 7.** *Connection label.*

Idle-Time-Slots Manager

Multiple time slots are associated with an output link. The mechanism for translating link identifiers into time slots is detailed below. The translation is handled by an idle-time-slots manager, whose block diagram is shown in Fig. 8. Idle-time-slot identifiers of the output link are maintained in a chain. Once an output link has been determined, time-slot identifiers are selected from the chain associated with it. Those time slots are assigned to a new connection. This chain-based management achieves a quicker response than a table-based method, which stores the states of all time slots (busy or idle). When the circuit is released, the time slots allocated to the circuit are freed and their identifiers are attached to the end of the chain.

Performance Evaluation

In this section, we address the performance of DBTN switches. We examine the performance of the shared buffer pool.

Performance Analysis of the Shared Buffer Pool

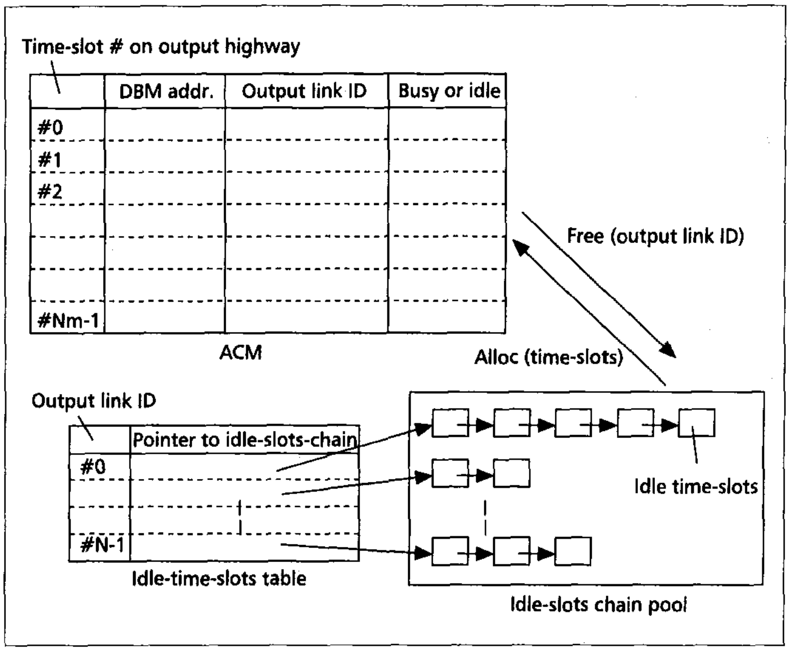
We compare two buffer-pool configurations, separate and shared (our proposed method) (Fig. 9). In the following, we investigate the economies of scale with respect to bandwidth and buffer size.

Bandwidth — For the separate method, a buffer is deployed at each output link. Excess burst is stored there. To avoid burst loss caused by the lack of bandwidth between the time-division switch and the buffer pool, we need to speed up the link between them so that it is faster than the output link speed.

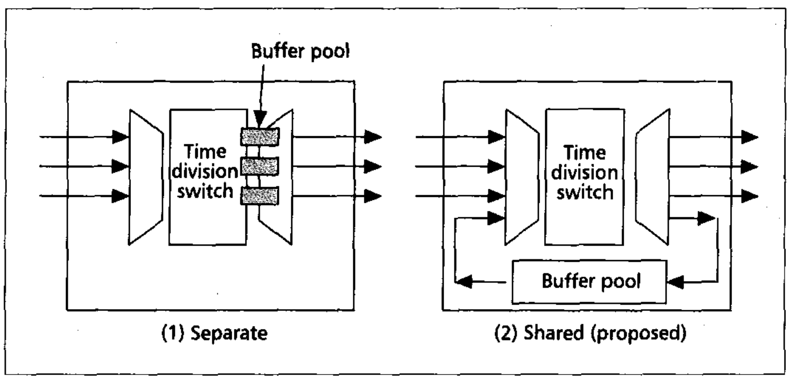
In contrast, for the shared method, the buffer is connected via the normal input and output ports of the TDM switch. The link speed between the time-division switch and the output link is equal to the output link speed. (The link speed between the time-division switch and the buffer pool is also equal to the output link speed.) As the number of switch ports increases, the number of buffer pools increases accordingly. That is, the number of switch ports for the buffer pools increases.

The required bandwidth for the buffer pool is a performance measure here. Let C' denote the bandwidth required for the buffer pool in the separate buffer pool approach. We define the required bandwidth ratio as:

$$(N_{\text{buffer-pool}} + N)C'/N, \quad (1)$$



■ **Figure 8.** *Translation of link identifiers into time slots.*



■ **Figure 9.** *Two alternatives for buffer-pool configuration.*

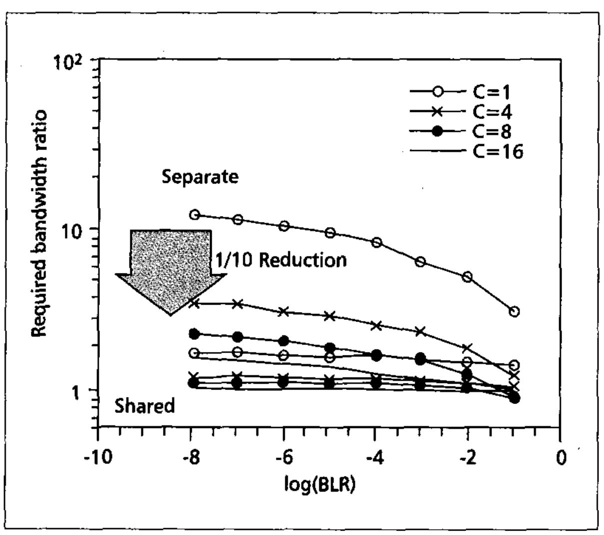
where N and $N_{\text{buffer-pool}}$ denote the number of output links and the buffer-pool.

We conducted computer simulations to obtain the required bandwidth for both methods. Figure 10 shows the required bandwidth ratio as a function of the burst loss ratio (BLR). Clearly, we can reduce the bandwidth dramatically by using our method. For example, the required bandwidth was reduced by an order of magnitude using our method when BLR = 1.0e-8. The reduction effect was significant when the link bandwidth C was small compared with the burst bandwidth. If we used $C = 1$ for the separate approach, we would need more than 10 times the output link speed for the buffer-pool bandwidth, which would not be practical.

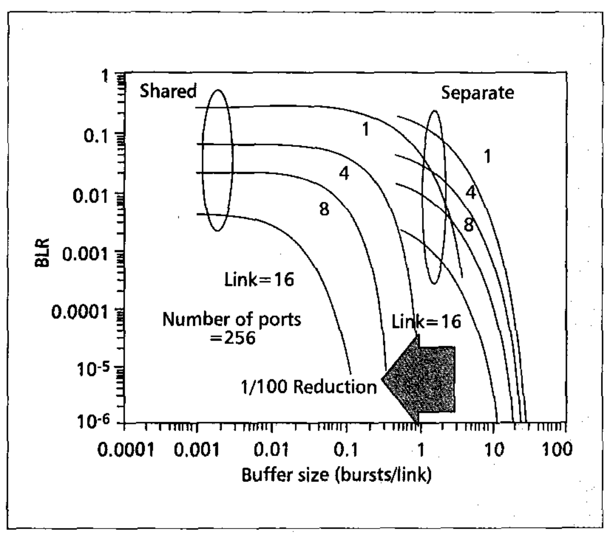
Buffer Size — For the separate control, each output link is equipped with a buffer pool, while in our approach the buffer pool is shared by all the output links. Thus we can expect the economies of scale with respect to the buffer size. Here we compare the buffer size required to support the target BLR. The buffer size is normalized by the burst size and the number of output links. Figure 11 shows the BLR as a function of the buffer size. We set the number of switch ports to 256 and the offered load $\rho = \lambda / C\mu$ was 0.6. The burst bandwidth changed from 1/16 of the link capacity to the entire link capacity. Our proposed shared method clearly reduces the required buffer size to 1/100 that of the separate method. This provides significant economies of scale.

EXPERIMENT

We constructed an experimental DBTN network (Fig. 12) that connects a server host and three client hosts. The server host



■ **Figure 10.** *Burst loss ratio as a function of buffer-pool bandwidth.*



■ **Figure 11.** *Burst loss ratio as a function of buffer size.*

and three client hosts were connected. The packet generator generated the IP packets. The time-slot-relay transported the IP packets based on the DBTN principle. In our first trial, we confirmed the feasibility of the DBTN network. The performance is being measured using the experimental network.

We have implemented a trial DBTN switching node using four field-programmable gate arrays (FPGAs). The detailed block diagram of the trial DBTN switch is shown in Fig. 13. The multiplexer, TSI, demultiplexer, and buffer pool are each implemented using a single FPGA. Figure 14 shows a photograph of our implemented DBTN switching node. The buffer pool collects and reports statistics on the total number of packets and the number of lost packets.

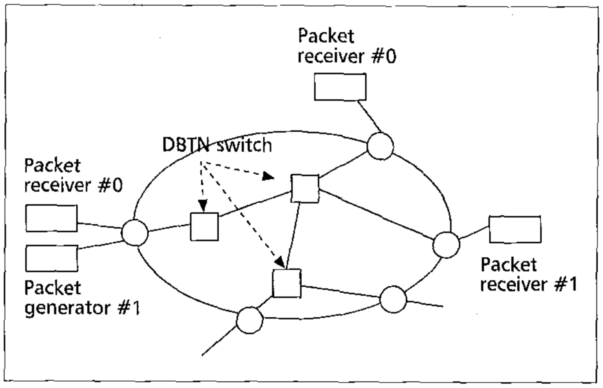


Figure 12. *The experimental DBTN network.*

RELATED WORK

In this section, we review previously proposed methods for high-speed burst transfer. We examined the packet switching technique because it has been envisioned to be suitable for burst transfer. Based on our examination, we compared the DBTN to other methods.

DATAGRAM AND VIRTUAL CIRCUIT

Time and Space Overhead — There are two packet transmission paradigms: datagram and virtual circuit. A routing decision is made for every arriving packet in the datagram network, while it is done only for a control packet in the virtual circuit network. A network address for the destination is used for the routing decision. Every packet carries the destination network address in the datagram network, while only the control packet carries one in the virtual circuit network. During the virtual circuit setup phase, every switching node determines a virtual circuit identifier ("label"), which will be included in the packet header of subsequent user data packets. Once the virtual circuit is set up, the user data packet is identified by the label attached in the packet header.

Let's consider space overhead in the packet header for both datagram and virtual circuit networks. When a routing decision is made, the network address is used. The network address should be long enough so that a unique network address can be assigned to all hosts in the entire network. Taking into account the inefficiency of hierarchical usage, we need to have a very long network address. For example, IP v6 uses a 128-bit address and NSAP uses an address of up to 160 bits. The label for the virtual circuit is relatively small, on the other hand, because it only has to be unique within each link. For example, Frame Relay uses a 10-bit label called DLCI, and ATM uses a 28-bit or 24-bit label, which has a hierarchy of VPI and VCI. (Most ATM switches compress these labels into a smaller one, say 16-bit in its proprietary cell header.) Thus the datagram network requires every packet to have a destination network address, and therefore it is less efficient in the space domain than the virtual circuit network. The

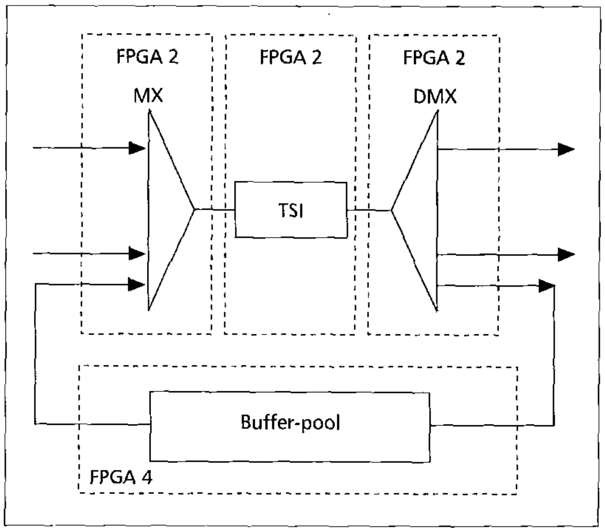


Figure 13. *A block diagram of the implemented DBTN switching node.*

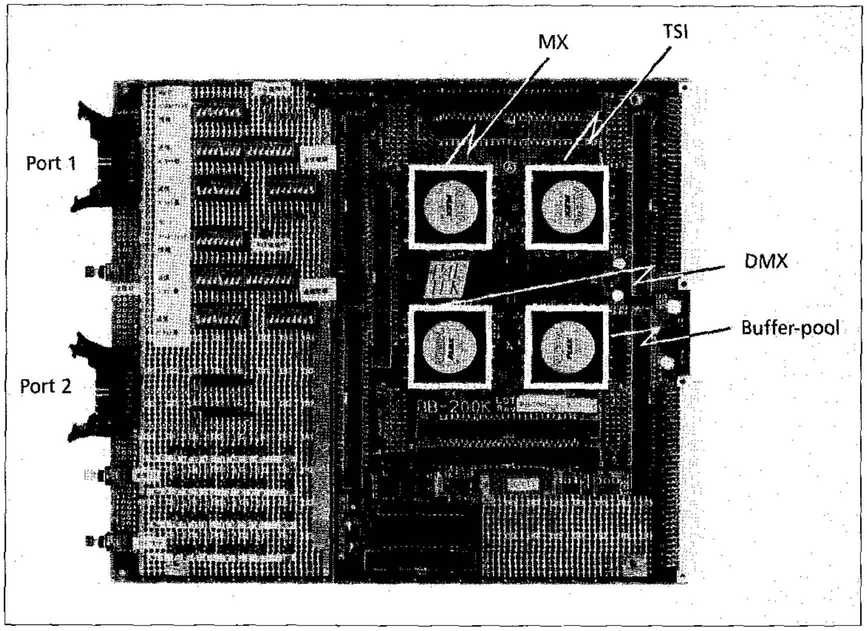
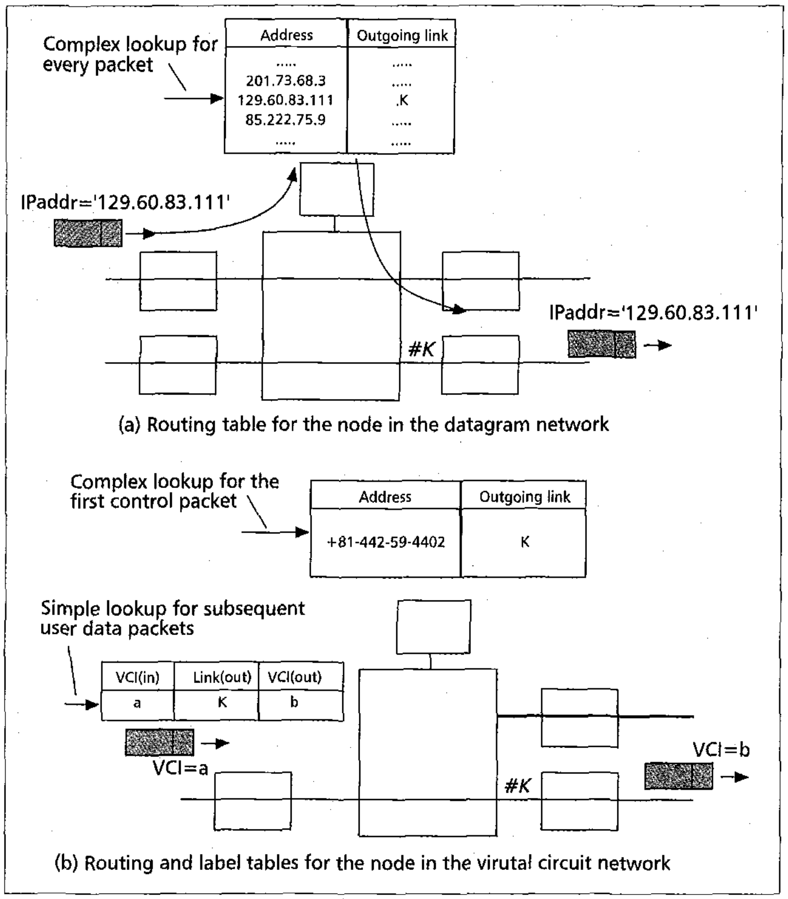


Figure 14. *A photograph of the implemented DBTN switching node.*



■ **Figure 15.** *Routing and label tables.*

datagram and virtual circuit are packet switching techniques where a header is attached for each packet. We should note that circuit switching does not require a header because a circuit is established by using a signaling message. As far as space-domain overhead is concerned, circuit switching is more efficient than packet switching, including datagram and virtual circuit.

Let's consider time overhead in setting up the virtual circuit. As mentioned earlier in this section, the virtual circuit needs to be set up before user data packets are sent. Control packets are used to set up the virtual circuit. It takes at least the round trip time for the control packet to travel between source and destination nodes. In addition, a processing delay at each node is required. The control packet is used to communicate information elements such as label values used at each hop, bandwidth, QoS profile, and so on. It takes considerable time (presumably, more than milliseconds) to parse the control packet and perform adequate actions at each node. We should note that when a connection-oriented TCP session is set up, intermediate nodes do not intervene in the session establishment procedure (they just transport IP packets). It takes a much longer time to set up a virtual circuit than to set up a TCP session. Thus the virtual circuit network is less efficient in the time domain than the datagram network in that it requires a virtual circuit to be established in advance.

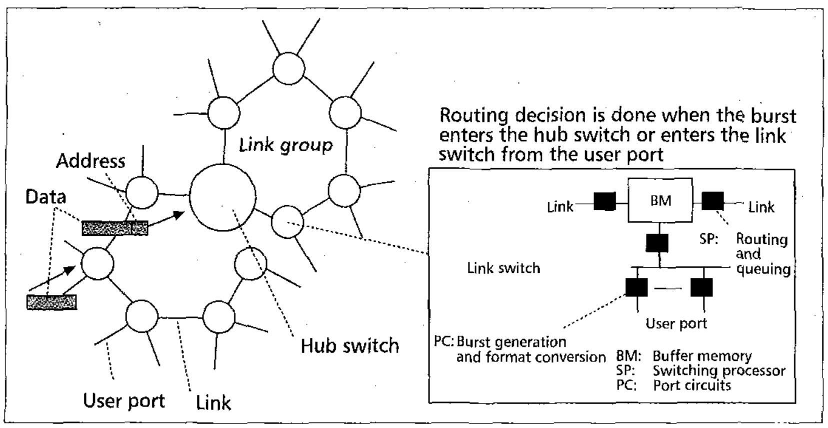
Scalability Issue — A node in the datagram network has a routing table while, a node in the virtual circuit network has both routing and label tables, as shown in Fig. 15. A node in the datagram network has a routing table to determine the outgoing link, which keeps the association between the destination network address and the appropriate outgoing link. A node in the virtual circuit network has both a routing table and a label table, which keeps the association among the incoming and outgoing labels and the appropriate outgoing link. The label table keeps this association for all virtual circuits being established.

The scalability problem of the virtual circuit network should be addressed. As mentioned earlier in this section, the virtual circuit network requires each node to have both routing and label tables (Fig. 15). As the link capacity increases, the number of virtual circuits increases. The scalability of the virtual circuit network is limited: the node in the virtual circuit network needs to maintain the state information for the huge number of virtual circuits. To support some sort of QoS, we need to set up a virtual circuit along which bandwidth is reserved. But we do not need to guarantee the bandwidth for all communications over networks.

COMPARISON TO RELATED WORK

Below we review high-speed burst transport methods and highlight differences between DBTN and these methods.

Burst Switching — Burst switching was proposed by Amstuts [4]. Figure 16 shows a typical burst switching node. The burst switching node consists of a hub switch and link switches. A link switch receives voice and data signals from user ports and converts these signals into bursts. The burst has a header that includes the destination port identifier. Several link switches

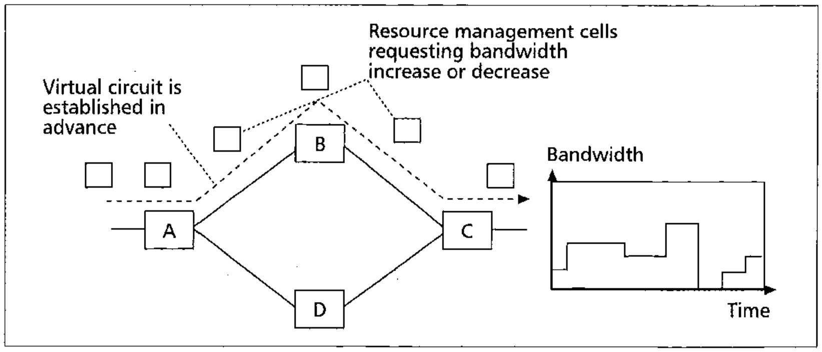


■ **Figure 16.** *A burst switching node.*

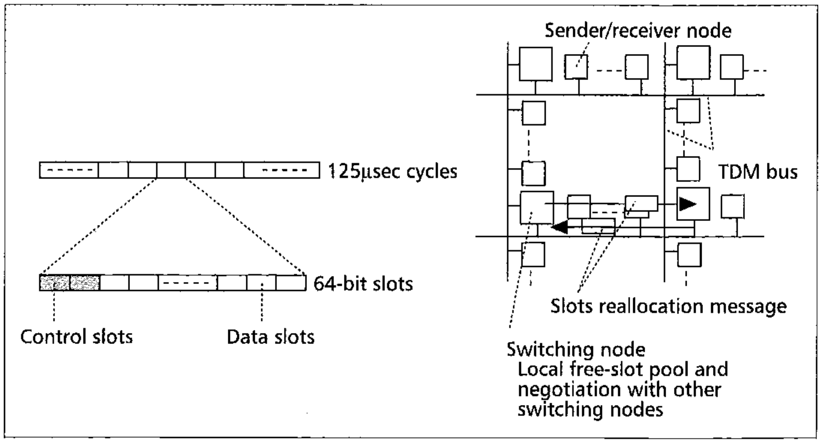
are connected in a ring fashion. The ring network of link switches is interconnected via the hub switch.

The link connecting the link switches is a TDM bus. When a new burst arrives at the ingress link switch, the circuit is dynamically set up and the burst is switched over the circuit. After the burst is finished, the circuit is immediately torn down. Both the DBTN and the burst switch avoid the time- and space-domain overheads of the packet switching technique. They also avoid the scalability problem of the virtual circuit.

The routing decision is made at the hub switch and the ingress link switch. The destination address, which consists of the link group identifier, the link switch identifier, and the link switch port's port identifier, is used to forward the burst. The burst switching is essentially based on datagrams because each burst carries the destination address. The DBTN employs time-slot-relay, which is self-explanatory to intermediate nodes, to switch the burst while the burst switching requires the destination address.



■ **Figure 17.** *Fast resource management.*



■ **Figure 18.** *Dynamic synchronous transfer mode (DTM).*

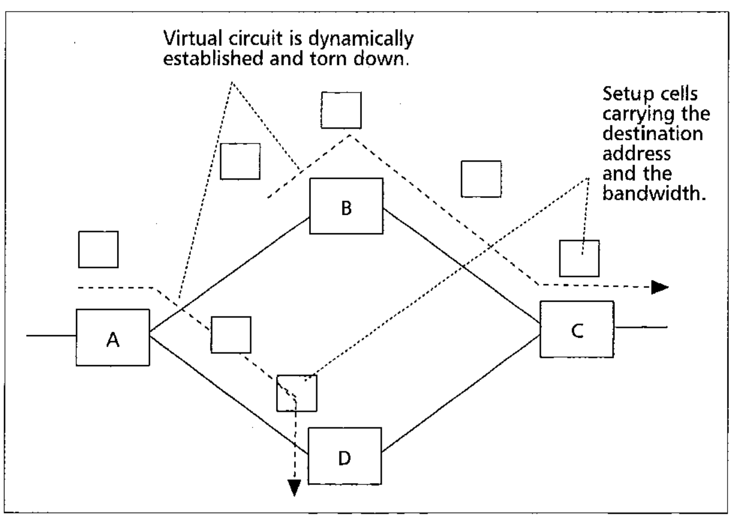
Fast Resource Management (FRM) — Fast resource management (FRM) has been proposed by Tranchier et al. and Ohnishi et al. independently [8, 9] (Fig. 17). It is based on ATM. Connection and bandwidth are controlled independently in ATM networks. In Fig. 17, the virtual connection is set up along with nodes A, B, and C but the bandwidth is not reserved. Resource management cells (RM cells) carry the bandwidth to signal the required bandwidth increase and/or decrease. FRM requires that the virtual circuit be established even when there are no bursts to send. FRM does not solve the scalability problem in the virtual circuit. DBTN is different from FRM in that FRM is based on a connection while DBTN is not. DBTN can send bursts to any destination while FRM can send bursts to only a single destination, to which the permanent connection is established.

Dynamic Synchronous Transfer Mode (DTM) — Dynamic synchronous transfer mode (DTM) was proposed by Bohm *et al.* [6] (Fig. 18). A DTM network consists of TDM buses inter-connected in a grid fashion and bursts are switched over the network. The TDM bus generates a 125-microsecond cycle, which consists of multiple 64-bit slots. Control and data slots are separated. Control slots are used to signal circuit setup messages.

A circuit between two TDM buses is created dynamically by using control slots. DTM is based on circuit switching technique and therefore avoids space overhead for both the datagram and the virtual circuit because it has no header.

Switching nodes connecting TDM buses perform switching between TDM buses. Idle time slots over TDM buses are managed in a distributed fashion. Control information on those idle time slots is exchanged through a control communication channel between adjacent nodes.

Both DBTN and DTM perform the source routing and create the circuit dynamically. DTM has constant overhead in a 125-microsecond frame for control slots while DBTN requires control overhead only when the circuit is established.



■ **Figure 19.** *Dynaflow service.*

Dynaflow Service — The Dynaflow service was proposed by Bian *et al.* [10] (Fig. 19). The Dynaflow service is based on ATM. Connection setup and bandwidth reservation are performed in a burst-by-burst fashion. Setup cells carry the required bandwidth and the destination address to set up connections. Bursts are switched over the connection.

Because the Dynaflow service is based on ATM, which requires a cell header for every cell, the space-overhead problem remains to be solved. DBTN is different from the Dynaflow service in that DBTN uses the source-routing based on the time-slots-relay while the Dynaflow uses the hop-by-hop routing.

CLOSING REMARKS

This article proposed a new high-speed network architecture called DBTN. In a DBTN network, a circuit is set up on-the-fly by sending a series of link identifiers to the destination called *time-slots-relay*. To avoid burst loss due to lack of bandwidth at transit nodes, we introduced the share buffer-pool configuration. We investigated the performance of our buffer-pool configuration and confirmed that our method provides great economies of scale with respect to the bandwidth and buffer size compared with the separate buffer-pool approach. We implemented the experimental system and confirmed the feasibility of DBTN.

In DBTN networks, a circuit between adjacent nodes is created and released dynamically, and short-lived fairly large data transfers such as WWW traffic are efficiently carried. Because DBTN makes use of circuit-switching technology, there is no delay jitter, so traffic management is simplified. DBTN is well suited for future high-speed backbone networks.

REFERENCES

[1] http://www.mit.edu/people/mkgray/net

[2] K. G. Coffman and A. M. Odlyzko, "The size and growth rate of the Internet," Tech. rep., 1998, http://www.research.att.com/amo/doc

[3] K. Thompson, G. J. Miller, and R. Wilder, "Wide-Area Internet Traffic Patterns and Characteristics," *IEEE Network*, vol. 11, no. 6, Nov./Dec. 1997, pp. 10–23.

[4] S. R. Amstuts, "Burst Switching: An Introduction," *IEEE Commun. Mag.*, Nov. 1983.

[5] S. R. Amstuts, "Burst Switching: An Update," *IEEE Commun. Mag.*, Nov. 1989.

[6] C. Bohm *et al.*, "Fast Circuit Switching for the Next Generation of High Performance Networks," *IEEE JSAC*, vol. 14, no. 2, Feb. 1996, pp. 298–305.

[7] J. P. Coudreuse and M. Servel, "Prelude: An Asynchronous Time-Division Switched Network," *Proc. IEEE ICC '87*, Seattle, WA, June 1987.

[8] D. P. Tranchier *et al.*, "Fast Bandwidth Allocation in ATM Networks," *Proc. ISS '92*, 1992, pp. 7–11.

[9] H. Ohnishi, T Okada, and K. Noguchi, "Flow Control Schemes and Delay/Loss Tradeoff on ATM Networks," *IEEE JSAC*, vol. 6, no. 9, Dec. 1988, pp. 1609–16.

[10] Q. Bian, K. Shiomoto, and J. S. Turner, "Dynamic Flow Switching: A New Communication Service for ATM Networks," *Proc. IEEE INFOCOM '98*, San Francisco, CA, Mar. 1998.

BIOGRAPHIES

KOHEI SHIOMOTO [M] (Shiomoto.Kohei@lab.ntt.co.jp) received B.E., M.E., and Ph.D. degrees in information and computer sciences from Osaka University, Japan, in 1987, 1989, and 1998, respectively. He is a senior research engineer at NTT Network Service Systems Laboratories, Tokyo, Japan. His current interests include protocol design, traffic control, and switch architecture for high-speed networking. He joined NTT, Tokyo, Japan, in 1989, where he was engaged in research and development of ATM switching systems. From August 1996 to September 1997 he was engaged in research on high-speed networking as a visiting scholar at Washington University, St. Louis, Missouri. He received the Young Engineer Award of the Institute of Electronics, Information and Communication Engineers (IEICE) in 1995. He is a member of IEICE and ACM.

NAOAKI YAMANAKA [SM] (Yamanaka.Naoaki@lab.ntt.co.jp) graduated from Keio University, Japan, where he received B.E., M.E., and Ph.D. degrees in engineering in 1981, 1983, and 1991, respectively. He is currently a senior research scientist, supervisor, and distinguished technical member in the Broadband Network System Laboratory at NTT. In 1983 he joined NTT's Communication Switching Laboratories, Tokyo, Japan, where he was engaged in research and development of a high-speed switching system and high-speed switching technologies. Since 1989, he has been active in the development of broadband ISDN based on ATM techniques. He is now researching future ATM-based broadband ISDN architectures and traffic management and performance analysis of ATM networks. He received Best of Conference Awards from the 40th ,44th, and 48th IEEE Electronic Components and Technology Conference in 1990, 1994, and 1998, respectively; the TELECOM System Technology Prize from the Telecommunications Advancement Foundation in 1994, the IEEE CPMT Transactions Part B: Best Transactions Paper Award in 1996, and the IEICE Transaction Paper Award in 1999. He is broadband network area editor of *IEEE Communication Surveys*, editor of *IEICE Transaction*, IEICE Communication Society international affairs director, and secretary of the Asia Pacific Board of the IEEE Communications Society. He is a senior member of the IEICE of Japan.