Microprocessing and Microprogramming 38 (1993) 733-739
North-Holland

Providing Circuit Service over a High Speed Deflection Network

E. Pagani and G. P. Rossi

Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano
via Comelico 39, 20135 Milano, Italy

Abstract

Deflection Networks, i.e. networks that use Deflection Routing on Mesh Topologies, have gained considerable attention in the last years. Their throughput and reliability behaviours indicate that they are an effective alternative to Metropolitan Area Networks (MANs) in extending the LAN services to larger geographical areas. As shown by recent investigations, Deflection Networks naturally provide a simple datagram service on top of which added value services can be created. In this paper, we describe and compare a set of protocols that provide, on the top of a basic Deflection Network, a service with guaranteed bandwidth and maximum delay, referred in the sequel to as *circuit service*. This is obtained by slightly modifying the basic access mechanism and makes Deflection Networks an easy to implement solution capable to integrate multimedia traffic and to support modern network applications even in wide areas.

1 Introduction

Deflection Networks, i.e. networks that use Deflection Routing on Mesh Topologies, have gained considerable attention in the last years [1, 2]. They are an effective alternative to Metropolitan Area Networks (MANs) in extending the LAN services to a larger geographical area and to support the communication needs of the modern network applications. The use of mesh topologies provides multiple paths between sources and destinations and guarantees great traffic handling capability and high reliability. Further, their throughput performance increases as new nodes and links are added to the network, thus allowing a gradual and modular growth. The association of deflection to normal routing resorts into flexibility and fast adaptation to the changes of traffic conditions. Packets are routed to the preferred link from source to destination and deflections from optimal routing have a limited impact on the end-to-end delay and throughput performance. As a consequence, node buffering is not needed and the network does not suffer from internal congestion. Packets are accepted as long as there is room in the network and this makes the behaviour of the network very similar to that of LANs.

*This work has been carried out under financial support of CNR, the National Research Council, Telecommunication Project, 1992.

As shown by recent investigations [3, 4], Deflection Networks naturally provide a simple datagram service on top of which added value services can be created. In this paper we describe and compare some original protocols that provide, on the top of a basic Deflection Network, a service with guaranteed bandwidth and maximum delay, referred in the sequel to as *circuit service*. This is obtained by slightly modifying the basic access mechanism and makes Deflection Networks a solution capable to integrate multimedia traffic and an effective alternative to the ATM solution, even in wide area networks.

The paper is organized as follows: in Section 2 we describe the basic Deflection Network behaviour and the advantages of the approach. In Section 3 we briefly discuss the overall protocol architecture to which the Circuit Protocol belongs, while in Section 4 a more detailed description of the protocol is provided. In Section 5 we describe the circuit service and the way it is accessible from the adjacent upper layer.

2 Deflection Networks

In this Section, we briefly describe the behaviours of basic Deflection Networks, or DNs, to which we will refer in the sequel of this paper.

Each DN node has the number of input links that equals the number of output links. Links have the same transmission speed.

For every input link, I/O operations are performed as those usually encountered in slotted rings. A packet addressed to the node is extracted. A packet in the local user queue enters the network if an empty slot is available, i.e., a slot has been received empty or a packet has been extracted.

Switching and transmission processes proceed in time slots. At a locally synchronized instant, each node switches to the preferred link the packets it has in input buffers according to address and local routing information. Should two or more packets be routed to the same link, they collide. Collided packets are *deflected* over non preferred links and they are not locally buffered. Deflections may waste the network capacity and reduce the maximum throughput attainable under ideal conditions. However, it has been shown that this waste can be very limited and that its impact decreases as the network size increases [4]. As an example, in torus networks with M nodes the achievable throughput is very close to $8\sqrt{M}$.

The routing function assumes that nodes know their shortest-path distances to all possible destinations, when each of their output links is used. Since no particular topology is assumed, a mechanism must be introduced to dynamically estimate the required distances. The Backward Learning Technique, firstly presented in [5], is well suited to this purpose as it is very easily implemented in a distributed fashion with an amount of processing at each node that can be performed within the time limits required by the high speed operation of the network. An implementation of this technique has been proposed in [2] where also the performance under time-varying topologies have been evaluated.

In general, the presented architecture assures many attractive properties which are typical of LANs. The network behaviour is stable under any load, i.e. the throughput cannot collapse due to the saturation of some resources. In fact, packets cannot be accumulated into nodes neither be blocked awaiting for the availability of transmission buffers. Packets are kept travelling in the network until they reach their destination. A positive throughput (often the maximum throughput) is assured even in saturation conditions, i.e. when packet leaving the network is immediately replaced by a new one; no flow control is needed to avoid congestion and there is no need to control the sharing of buffering resources to avoid deadlocks; under a uniform traffic matrix, the network assures the same throughput between any pair of nodes. In addition, and differently to current LANs and MANs, the network throughput increases as new nodes and links are attached to the network.

3 The Protocol Architecture for the Deflection Network Project

The very attractive potential features of Deflection Networks led the Italian National Research Council to found a project to build an experimental network using the available low cost technology and off the shelf components and able to provide an aggregate throughput of tenths of Gb/s. In this Section we briefly describe the overall protocol architecture to which the circuit protocol belongs.

Although the DN basic services are attractive, delivery, error correction and sequencing are not guaranteed. These must be provided by added functions, as can be obtained by a complete stack of protocols that has been devised for the DN prototype (Figure 1).

The transport protocol has lightweight behaviours and provides for error recovery, flow control and sequenced delivery of transport packets. It has three Modes of operations:

Mode a : is suitable for reliable transfer of large files without specific requirements of bandwidth assignment or time constraints. Flow control is obtained by rate control and window mechanisms, i.e. transport packets are transmitted under restriction of the rate control algorithm as long as the window remains open. The Mode a entity utilizes the connection-less service offered by the underlying layer;

Mode b : is suitable whenever real time constraints are associated to the transport traffic and is oriented to transport both large bulk of data and small packets according to the application needs. The protocol exploits the Circuit Services offered by the Network Layer and offers a guaranteed rate transfer. Error recovery is not provided;

Mode c : has error control, a simple flow control and accepts both unicast and multicast addressing. It is based on implicit connection set up and is suitable for request-reply interactions.

The network layer provides both connectionless service and circuit service, i.e. a *pipe* from source to destination with guaranteed bandwidth and maximum delay. Multicast addressing capabilities are also under study and are addressed to support the communications amongst the processes of a *group* [6]. Routing protocol is based on backward learning to define the shortest-path distances to all possible destinations.

Network services are created over the Data Link Layer. This provides the access mechanism according to a priority based policy that has been

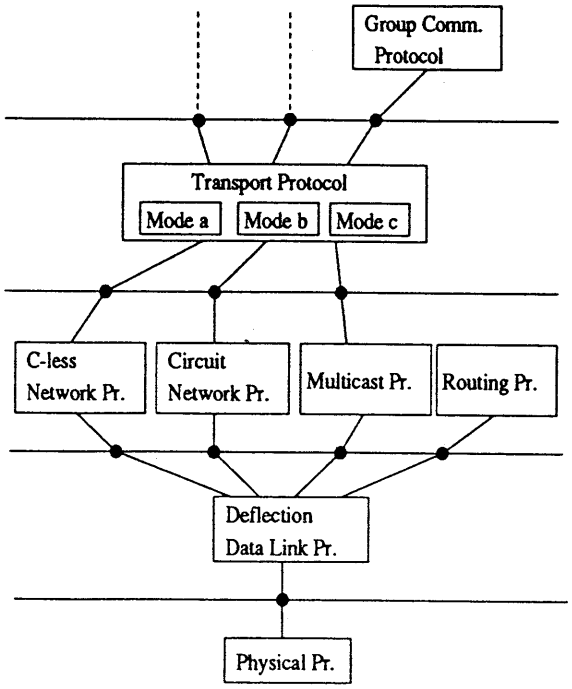

Figure 1: Deflection Network architecture.

obtained by slightly modifying the schema described in Section 2.

4 Guaranteed Access and Circuit Protocol

The major problem in providing guaranteed bandwidth in a distributed environment, are due to the fact that any node must be able to know whether or not the resources to accommodate a new connection are available and, when available, to reserve them. Further, the sharing of resources with the datagram traffic must not affect circuit traffic.

In classical packet switching networks, the connection mechanism must control network resources, such as bandwidth on link and buffers at nodes. Its design is very critical as a misbehaviour may cause unavailability of resources, throughput collapse and very long delays (congestion).

The environment of deflection network is completely different. The network behaviour is stable under any load conditions as no internal congestion may occur, even if the network is working in saturation. Therefore, assuming the network delay much smaller than the end to end delay, the only resource to be measured is the rate of the network accesses being granted to a node.

4.1 The Circuit Protocol

To provide circuit service, it is sufficient that each node is able to determine, during the circuit set-up phase, whether or not input resources are available at the node and, once available, allocate them to the circuit. This capability is achieved by means of a distributed algorithm that allocates bandwidth to the circuit up to the requirements, thus avoiding waste of network capacity.

The protocol uses two types of packets: ***signal packets***, that are used during the circuit set up phase to measure and allocate resources; ***circuit packets***, that are used to transport user data over the circuit. The flows of these types of packet form different traffic patterns that share the network resources with datagram traffic.

During the set up phase, the circuit protocol uses signal packets to measure the capability of the network and to allocate the required bandwidth. The measure we need on the network is both *local* and *global* to guarantee the required resources from source to destination as long as the circuit remains open. The local measure can be performed just by controlling the length of the queue associated to the Data Link Connection End Point (CEP) in which the source protocol entity places signal packets at the rate r_{send}. Of course, should the queue grow up, the local resources are not enough to accommodate the traffic at the required rate and the circuit cannot be accepted. The global measure is obtained by forcing the circuit destination entity to report the observed receiving rate of signal packets. Whenever the destination circuit entity receives a signal packet, it compares the specified rate with the receiving rate r_{rec} that it measures. Periodically, it sends back to the source the value r_{rec}. When the source observes that r_{rec} approximates r_{send}, it decides to switch signal traffic to circuit traffic. If $r_{rec} < r_{send}$, it refuses the circuit request. If several circuits are open at different rates r_{send}, the signal phase is lasted to cover the interarrival time of the slowest circuit.

The described signalling phase does not lead to a stable resource allocation without a proper access mechanism provided by the underlying data link protocol. In fact, the capability to access the network depends on the availability of free slots. We need that high priority traffic patterns have granted access to the measured network resources.

4.2 The Access Protocol

The access protocol exploits the link *preemption* over different traffic patterns. Preemption indicates the capability of high priority traffic (i.e. circuit over signal and signal over datagram) to preempt a link to a lower priority one.

Different strategies can be applied to deal with preempted packets:

1. preempted packets are removed from the network; in this case a fired signal packet will never reach its destination, thus reducing the r_{rec} value, while a fired datagram packet resorts to a low degree of network reliability. The loss of datagram packets should be recovered by transport entities that can renegotiate the packet flow over the involved transport connections. The reduction of datagram traffic automatically reduces the probability of further packet preemption;

2. signal packets are fired, while datagram are entered a temporary queue Q_t. This approach produces packet starvation and dropping because of Q_t overflowing;

3. to avoid starvation and to prevent queue overflowing, that may occur when the approach 2) is used, the *free token booking* mechanism can be applied to empty the temporary queue Q_t. The booking mechanism was originally introduced to provide fair access and avoid node lock out on deflection networks [4]. Nodes that need empty slots *piggyback* their request on the packets passing by. Once these packets are removed from the network, the empty packets, or *tokens*, are forwarded to the requesting node, according to a priority based mechanism, to provide access opportunities. By associating a sort of aging zone to the queue Q_t, the same mechanism may be activated whenever some age limit is reached and stopped when Q_t is emptied. This guarantees that packets do not stay awaiting in Q_t for a long time and that the token draining mechanism is activated just when needed. In Figure 2, we report the relationship between the number of packets in Q_t and the amount of generated tokens against the simulation time assuming a 10×10 torus network that operates with 100 established circuits, at rate $r_C = 1/2$ *pck/slot* and background datagram traffic capable to saturate the network. As shown, when the amount of queued packets reaches a given threshold (set to 6 during the simulations) at any node, the booking mechanism starts to drain tokens from the surrounding nodes until Q_t is emptied. Datagram packets may only experiment long delay to reach their destination but they are never dropped for queue overflowing.

The described preemption strategies have different behaviours as far as the implementation and the impact on datagram traffic are concerned. The implementation of the booking mechanism (approach 3) is not as simple as the basic access scheme, although it allows a graceful sharing of

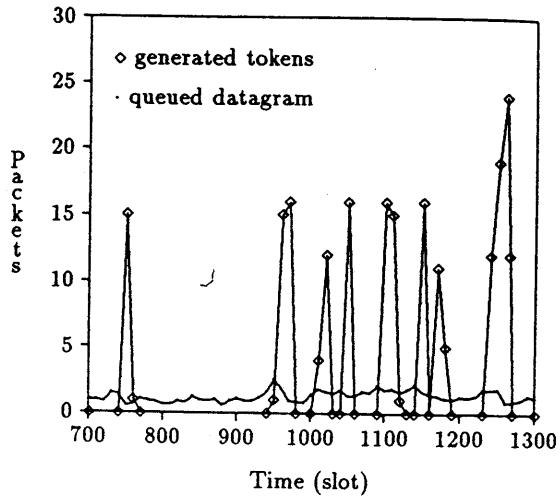

Figure 2: Average amount of packets in Q_t and drained tokens vs. simulation time.

the network resources among different traffic patterns.

The first described approach is suitable for a simple implementation and potentially guarantees a high circuit throughput since it is not affected by datagram packets. Nevertheless, the datagram traffic results to be highly unstable and unreliable, suffers from unfairness and the recovery performed at transport layer slowly adapt to the changing network conditions. Indeed, a better solution would be achieved by regulating the datagram traffic according to the circuits needs. The tuning of the datagram traffic can be performed in two directions: 1) by avoiding that accepted datagram packets are routed towards nodes with high circuit traffic. Network zones in such conditions are dodged round by deflecting the packets to suboptimal paths, and, 2) by reducing the amount of accepted datagram packets when the resources being required by circuits grow up. A simple algorithm allows to measure the circuit resources and to control the datagram flow thus leading to a fourth policy:

4. when, in order to allocate resources to a circuit, a node resorts to preempt (remove) a datagram packet, it tries to stop the datagram flow coming from adjacent nodes. To this purpose, it piggybacks the request (by setting a boolean variable) on the packets being ready to output (if packets are not available the problem does not arise). Upon receiving such a request, the adjacent nodes stop routing datagram packets to the neighbour node until the lock condition is released.

Should it experiment the same condition, the lock signal is propagated as a sort of back pressure. To cope with transient bursts of network traffic, each node could maintain an internal queue, say Q_t, where it temporarily saves the datagram packets being dropped to allocate circuit packets. The lock condition is generated when a given threshold is reached and released when Q_t newly becomes empty. The nodes that observe lock conditions do not accept new user datagram packets thus reducing the amount of circulating packets. Of course, the higher the circuit traffic, the longer is the end to end delay that datagram packets may experiment. This depends on a higher amount of time spent into the user queues and on longer paths to reach the destination because of deflections.

4.3 Analysis of the protocol

In this Section we want to analyse the circuit protocols independently of the implementation and we use simulation results to clarify the effects that the different access policies have on the circuit establishment.

We investigated by simulation the torus network performance assuming that each node generates background datagram traffic T_D, sufficient to saturate the network resources. The circuit protocol attempts to establish up to 200 circuits at the rate $r_C = 1/5$ *pck/slot* each to generate the traffic T_C. In the following, we describe the results obtained when both T_D and T_C are uniformly generated over a 10 × 10 DN. In Figure 3.a), b), c), the datagram (S_D), circuit (S_C) and global (S) throughputs are reported against the offered circuit traffic T_C for the 1st, 3rd and 4th protocols, respectively. We observe that the protocols perform similarly, the throughput $S_C = T_C$ grows up towards the maximum throughput by consuming the datagram throughput and about 160 circuits out of 200 requests are established. This indicates that the signal mechanism properly measures the network resources and does not allocate more circuits than allowed.

The first protocol kills many datagram packets (12.6% of them are killed when 160 circuits are open), thus leaving several empty slots that are not immediately filled up by newly accepted packets. As a consequence, lower S_D values are achievable. The booking mechanism (Figure 3.b)) is not able to guarantee values $S_C = T_C$ and about the 7% of throughput is lost. This derives from the fact that circuit packets do coexist with tokens and datagrams without extracting or blocking them. As a consequence, circuit packets are often deflected to suboptimal and longer paths.

The 3rd and 4th protocols never remove datagram packets from the network and, although the

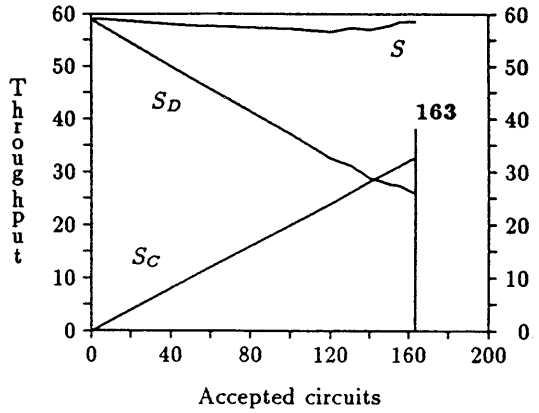

Figure 3.a

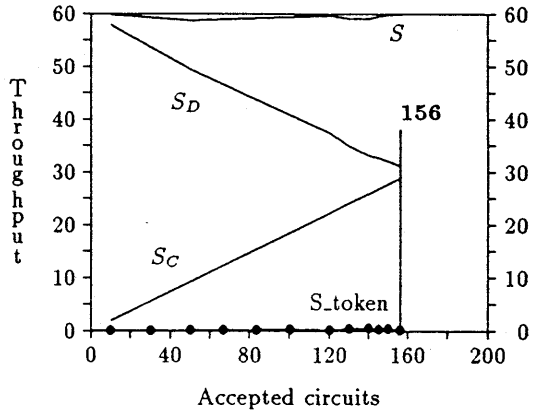

Figure 3.b

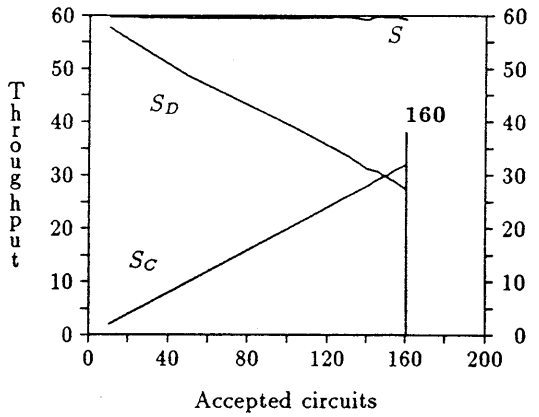

Figure 3.c

Figure 3: Different Throughputs S, S_C, S_D vs. offered traffic T_C, $r_C = 1/5$ *pck/slot* - a) 1st protocol, b) 3rd protocol, c) 4th protocol.

requested circuit rate is very high (consider that it corresponds to 20 Mbps per circuit with 100 Mbps links), they rarely resort to either booking or locking respectively. Under this condition, datagram traffic is slowly affected by circuit traffic and, when the protocol 3 is used, the fraction of network capacity that tokens waste is very little.

More stressed conditions are observed when the circuit rate grows to $1/2$ *pck/slot* (Figure 4.a), b),c)). As shown, less circuits can be established and, when the 3rd protocol is considered, the capability of regulating the access at any node and for different traffic patterns is achieved against the fraction of capacity that tokens waste. In fact, tokens cover short network travels without transporting user data and the amount of generated tokens is, generally, higher than required (see Figure 2).

The 4th protocol heavily uses the lock-unlock mechanism to regulate the datagram flow, thus confining datagram packets into the input queues. In this case, the adopted policy leaves some empty slot circulating in the network, can lock out some network zone and forces datagram packets to follow longer paths. As a consequence, the datagram throughput collapses when the amount of open circuits becomes higher.

Although the described protocols use different policies to gain and to maintain the access to the network resources, they have comparable performances. Nevertheless, datagram traffic experiments either unreliable travelling from source to destination or longer end to end delay that derives from the longer paths and times spent into the input queues. In Figure 5, we show the measured datagram delay against a datagram traffic that grows up to saturate the network resources (about 0.6 *pck/slot* for each node) for a given amount of established circuits (set to 100 during simulations). As shown the 3rd and 4th protocols impose the same delay to datagram packets because they force them into (different) waiting queues, while the first policy seems to perform better because we have not considered the killed packets to compute the delay. Anyhow, simulations indicate that datagram and circuit packets can actually share the available resources and coexist when their rates are slightly under the saturation conditions. This ought to be the most common traffic condition.

5 The Circuit Service

The described circuit protocol provides a pipe from source to destination with guaranteed bandwidth allocation and maximum guaranteed delay. The service is described by the primitives: *N.circuit.set.Rq()* and *N.circuit.set.Conf()*, that allow to set-up the circuit from source to destina-

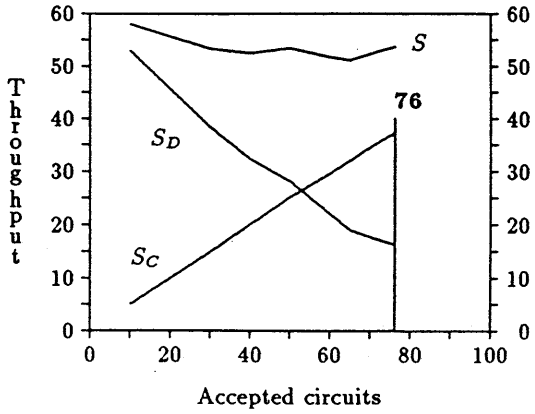

Figure 4.a

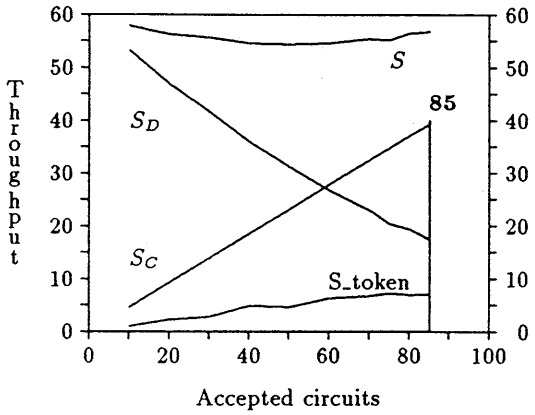

Figure 4.b

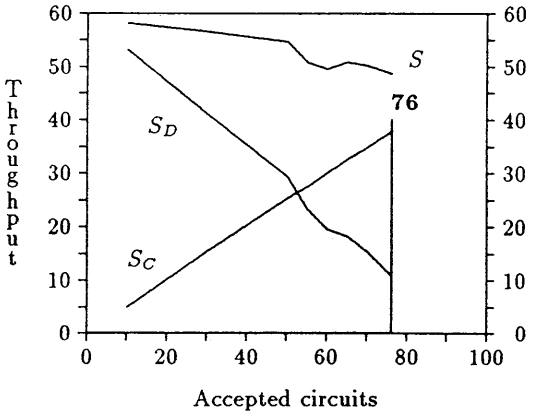

Figure 4.c

Figure 4: Different Throughputs S, S_C, S_D vs. offered traffic T_C, $r_C = 1/2$ *pck/slot* - a) 1st protocol, b) 3rd protocol, c) 4th protocol.

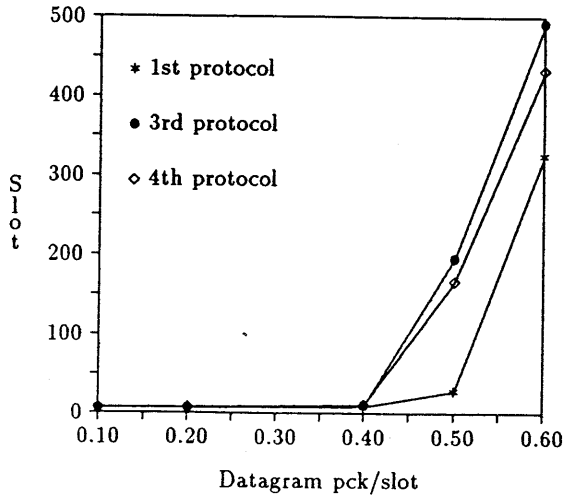

Figure 5: End to end datagram delay vs. datagram traffic with 100 open circuits and $r_C = 1/5$ *pck/slot*.

tion, *N.circuit.data.Rq()* and *N.circuit.data.Ind()* that allow to write and to read data packets into and from the pipe at the specified rate. Through the interface the upper transport entity observes a sort of implicit circuit set up mechanism, i.e. transport data can be inserted in the *N.circuit.set.Rq()* primitive. When calling the circuit establishment service, the upper entity remains blocked as long as the subnetwork measures the capability to satisfy the circuit request. When the signal phase correctly terminates, the source circuit entity decides to switch to circuit traffic, transmitting the previously supplied data, while the upper entity may continue producing *N.circuit.data.Rq()* operations.

The transport protocol has the responsibility to supply the circuit entity with transport data units at the requested rate r_C. If the transport entity generates packets at a higher rate, then the packets are likely to be dropped into the interface queues. Should the rate decrease, the network entity covers the missing packets with empty packets, thus resorting in a waste of bandwidth.

The adopted circuit set up algorithm can be fully exploited if the transport protocol makes use of an implicit connection set up mechanism. When explicit set up is used, the time that the transport entities consume to negotiate the connection parameters is covered by the underlying circuit protocol with the transmission of empty packets at the rate r_C to maintain the circuit open.

6 Concluding Remarks

A new set of protocols to establish network circuits with guaranteed bandwidth and maximum delay has been introduced. The protocols can be easily implemented on the top of a datagram Deflection Network, which provides the access scheme based on a priority mechanism. This makes Deflection Networks a solution capable to integrate multimedia traffic and an effective alternative to the ATM solution, even in wide area networks.

The described approaches are the evolution of the protocol firstly introduced in [4], that is based on the use of the booking mechanism. The new protocols introduce a more reliable and trusted measuring phase and the priority based access mechanism eliminates the pervasive use of token draining that wastes network capacity and makes the implementation difficult to conduct.

The study that has been presented in this paper is involved in a national project which aims to build an experimental network using the available low cost technology and off the shelf components and able to provide an aggregate throughput of tenths of Gb/s. The availability of the first prototype, that is being implemented, will serve as a testbed for the whole protocol stack that we introduced in Section 3.

References

[1] N. F. Maxemchuck, *The MANhattan street network*, IEEE GLOBECOM '85, December 1985, New Orleans, La.

[2] F. Borgonovo and E. Cadorin, *Locally-Optimal Deflection Routing in the Bidirectional Manhattan Network*, IEEE INFOCOM '90, June 1990, S. Francisco, CA.

[3] F. Borgonovo, L. Fratta and F. Tonelli *Circuit Service in Deflection Networks*, Proc. of IEEE INFOCOM '91, Bal Harbour, Miami, FL, USA, April 9-11, 1991.

[4] F. Borgonovo and L. Fratta, *Deflection Networks: Architectures for Metropolitan and Wide Area Networks*, Computer Networks and ISDN Systems, Vol. 24, 1992, pp. 171-183.

[5] P. Baran, *On Distributed Communications Networks*, IEEE Transactions on Communication Systems, Vol. CS.12, March 1964, pp. 1-9.

[6] R. Aiello, E. Pagani, G.P. Rossi, *Design of a Reliable Multicast Protocol*, Proc. of IEEE INFOCOM '93, San Francisco, March 1993, pp. 75-81.