Addressing and Routing in Heterogeneous Data Networks

J.William Atwood
Jean-Marc Jézéquel
Anindya Das
M'Hamed Nour
DRAFT — DO NOT DISTRIBUTE

J.William Atwood, Department of Computer Science, Concordia University, 1455 de Maisonneuve Blvd, West, Montreal, Quebec Canada H3G 1M8
+1 (514) 848-3046 +1 (514) 848-2830 (fax) bill@cs.concordia.ca

Jean-Marc Jézéquel, IRISA, Campus Universitaire de Beaulieu, 35042 RENNES cedex France +33 99.84.71.00 +33 99.38.38.32 (fax) jezequel@irisa.fr

Anindya Das, M'Hamed Nour, Département d'informatique et de recherche opérationnelle, Université de Montréal, C.P. 6128, succursale A, Montréal, Québec Canada H3C 3J7
+1 (514) 343-5746 +1 (514) 343-5834 (fax) {das, nour}@iro.umontreal.ca

Abstract

Computer Networks are moving towards interconnection of local networks through the Asynchronous Transfer Mode (ATM) technology. The ATM offers speeds substantially above those available with current wide area networking technologies. However, effective internetworking using the ATM requires three conditions to be met: increasingly powerful routers must be used at the boundaries between ATM services and existing technologies; dynamic routing must be introduced within ATM itself; and unification of previously-distinct addressing domains (networks utilizing a single addressing family) is imperative.

The parametric addressing capability of the Xpress Transfer Protocol (XTP) permits the unification of addressing domains, provided that certain additional support is built into the associated routers. The CCITT Intelligent Networks (IN) structure provides a generic platform for this support, and for the support of dynamic routing within the ATM service. The increase in routing capabilities can be achieved using general purpose Distributed Memory Parallel Computers (DMPCs). DMPCs also offer the necessary processing power for the IN software.

Keywords: Computer Networks; Addressing; Routing; Xpress Transfer Protocol; Asynchronous Transfer Mode; Intelligent Networks.

1 Introduction

Traditional data networks have been characterized by point-to-point communication, single protocol families, and single addressing plans. Individual protocol families typically have had an associated addressing plan.

The development of protocol layering [1] has achieved a large measure of independence from the underlying media. However, the existence of the boundaries between different media has required internetworking—a routing function normally associated with the network layer. Proposed routing schemes to date have not, however, solved the problem of interconnecting hosts in different addressing domains. A solution to this problem would provide a valuable service to the networking community.

The desire to unify the movement of voice, video and data has led to the development of the Asynchronous Transfer Mode (ATM) [2], a design for a wide-applicability, high-speed communications service, which has been standardized by the CCITT [now the Technical Standardization Section of the International Telecommunications Union (TSS-ITU)]. It is seen by many as becoming ubiquitous, and therefore removing the need for internetworking and routing. Our position is that routing will continue to be needed, and that the higher speeds of future networks will require even more powerful routers than those available today.

In Section 2, we review the attributes of data networks, with emphasis on address families and routing. In Section 3, we introduce the Xpress Transfer Protocol, and show how its parametric addresses can be used to unify addressing domains. We also discuss how routing in XTP and routing in ATM can be uniformly treated as an Intelligent Network service. Section 5 discusses the construction of high-speed routers for XTP, and Section 6 lists our conclusions.

2 Data Network Attributes

2.1 Architectures and Services

A wide variety of physical media has been used in existing data networks, with widely varying Medium Access Control mechanisms and physical level protocols. One characterization that has been used is that of distinguishing between *local area* networks (LANs) and *wide area* networks (WANs). However, *campus area* networks and *metropolitan area* networks have now been added to this classification. Many technologies have been proposed, each targeted at a specific niche in the data-communications market, and enormous investments have been made in hardware and software based on these available technologies. In the past, significant differences existed between LANs and WANs, as LANs tended to be implemented on *shared* media, while WANs tended to be implemented on *switched* media.

At the boundaries between different media, equipment operating at the three lowest (physical, datalink, network) levels of the OSI hierarchy provides both routing capability and conversion of media formats. As technology has developed, the speeds available for WANs have increased, to the point where they now rival or exceed the speeds of most installed LANs. A particular example of this is the ATM, which is as adaptable to LAN environments as it is to WAN environments. This would seem to encourage a widespread conversion to the ATM technology, except for the large installed base of the old technologies and the high cost of the ATM hardware. Therefore, this new technology will get "close" to the desktop, and possibly all the way to some high-performance processors such as servers, but the "last 100 meters" will use other (mostly existing) media. This implies that routers will continue to be needed, and that the performance requirements will increase substantially as network speeds increase.

WAN transport services have provided primarily point-to-point connections, oriented either towards transferring relatively large amounts of data, or towards remaining connected for relatively long periods of time. Given the relative unreliability of the underlying media in early systems, these *connection-oriented* transport services have three phases: establish the connection, move the data, close the connection; this was based on the idea that it was necessary to ensure that the peer entity was really there before beginning data transmission.

LANs introduced the possibility of much more intensively cooperative programs, and the need for minimal overhead in these environments. The *connectionless* or *datagram* communications model was felt to have low overhead, because the communications subsystem needs to retain little or no information (state) about ongoing traffic. However, it also gained some of its efficiency from the assumption that unreliable (unacknowledged) transmission was acceptable. If reliable transmission was required, the reliability had to be provided at a higher level in the protocol stack.

More recently, it has been recognized that operation somewhere between the above extremes may be useful: *acknowledged datagrams* and *transactions* are examples of this trend.

2.2 Addressing

In datagram networks, the objects sent are completely independent of each other. Each object must contain the destination address as part of its overhead (typically in the header). For the Internet Protocol [3], source and destination addresses are each four bytes, and the network must be able to transmit objects up to 576 bytes (larger sizes are permitted, but not required). As noted above, datagram networks are stateless—there is no correlation between subsequent objects, so there is no possibility of using knowledge about the resource requirements of a stream of objects to control the flow or congestion in such a network. In addition, the size of the address field must be kept fairly small, to limit the overhead from these addresses, and to permit rapid routing of the objects.

It is interesting to note that the success of the datagram orientation of the Internet to date is based in part on the fact that its addresses are short (four octets). However, this address space is proving to be inadequate for the number of sites currently connected to the Internet. The Internet Protocol Address Extension (IPAE) project [4] is proposing to increase the size to eight bytes for each address. Addressing in OSI networks is variable, between one and 20 octets. These new address formats clearly present increasing overheads for datagram style operations.

In connection-oriented networks, while the complete address must be sent initially, it is possible to represent a connection with a short "connection identifier", and rely on the intermediate routers to retain state for a particular connection, so that decisions for the objects that follow can be based on simple table lookups. The requirement to mix voice, video and data has led the CCITT to define a basic unit for data carriage in the ATM, which has a 48-octet payload. This is combined with a 5-octet header into a 53-octet *cell*. Of necessity, the addressing information in a cell is small—only 28 bits are available for cell *labelling* within the network, and only 24 bits are available at the interface between the user and the network.

Over the years, a variety of addressing plans has been proposed and implemented in data networks. Each protocol stack appears to have its own. A given end system must support at least one protocol stack and its addressing plan, and may support several, but this support is usually of the form "establish a connection for using a particular protocol stack and within a specific addressing domain". Given the enormous investment in software for these specific situations, the wide range of addressing schemes will be in place for many years to come. It is clearly desirable that ways be found to utilize this investment: to stay within a particular address domain when possible, but to provide relatively easy access to services delivered in different addressing domains, when required.

2.3 Multicast Addressing

Multi-peer communication is important for some applications. For shared-media (local) networks, this can be achieved using the *broadcast* capability of the shared medium, and modes of operation where the membership of the group is not known are possible. For non-shared media, and for wider areas, much more selective mechanisms must be used [5], and it becomes necessary to specify group membership precisely, to prevent wasting bandwidth unnecessarily.

While shared-media physical layers have always had *multicast* or *group* addresses, it is only recently that network-level addressing plans have begun to have provision for multiple recipients of a transmitted message. The Internet addressing scheme dedicates one part of its address space for group addressing [6]. Within the OSI framework, a project is currently underway to define what *multi-peer* addressing means [7].

2.4 Routing

When two end systems that are not on the same (shared medium) LAN need to communicate, it is necessary to route the data through the underlying physical network. If the overall paradigm is one of connectionless transport, then the individual objects each carry an address, and this address is interpreted at each routing node. The address may either be the destination address, or in the form of a list of sites to be visited along the way. In the first case, the responsibility of the routing node is to forward the data to a node that is "nearer" to the destination. In the second case, the routing node forwards the data *to* the next named node (*strict* source routing), or *towards* the next named node (*loose* source routing).

If the overall paradigm is one of connection-oriented transport, then it becomes possible to contemplate replacing the real (globally unique) address with some sort of short identifier, which has only local significance, and then use this short identifier in all subsequent objects to be sent. Use of such a short identifier, with local significance, as an index into some sort of table can provide fast access to the information necessary to forward an object within a network. This is precisely the way in which *route* values are used in XTP (see Section 3.1).

This separates the routing problem into two parts—establishing the route, and using the route to move data. Many routing protocols exist [8], corresponding to each of the protocol stacks and/or addressing families that exist. The purpose of each of these routing protocols is to maintain the distributed database that contains the routing information, in the face of the continually changing topology of the network. Initial offerings of the ATM services will require that a route be established at the time that a connection is ordered (static allocation), and future offerings will permit dynamic allocation of routes. Le Boudec [2] suggests that this will be based on an extension of existing *signalling* protocols such as Q.931 [9]. *Switching* of ATM cells is based on the cell labels in an ATM cell—at each ATM switch, a routing table (constructed by the signalling protocol software) is consulted to determine the next edge to follow in the ATM network, and the new cell label to be used.

What is needed is a very generic platform on which to base these routing and label assignment strategies, such that a variety of routing plans/address families could be offered to the public.

2.5 Multicast Routing

On a shared medium, achieving multicast is very simple; all stations on the medium naturally receive their own copy of the multicast message. On switched media, multicast must be achieved either by replicating the message once for each participant in the group, or by replicating the message explicitly when the paths between the (one) sender and the (multiple) receivers diverge. The MOSPF routing protocol [5] defines the Internet procedure for ensuring that each member of the group receives a copy, but unnecessary duplicate messages are not sent along shared paths. It requires considerably more intelligence in the router than is necessary for routing unicast packets.

2.6 Intelligence

The need to introduce new capabilities in the telecommunications network, and to facilitate and accelerate service implementation and provisioning in a multi-vendor environment, has led to the introduction of a series of Recommendations within the CCITT for *Intelligent Networks* [10]. While these Recommendations were conceived in the context of telephone-network subscriber-oriented services, we will explore their potential use for routing later in this paper.

3 Bridging Address Domains Using XTP

The problem of bridging addressing domains can be addressed using some of the attributes of the Xpress Transfer Protocol. In this section, the pertinent properties of the protocol are presented, and then the solutions to the problems outlined in the previous section are shown.

3.1 The Xpress Transfer Protocol

The Xpress Transfer Protocol (XTP) [11, 12] is a so-called *transfer* protocol, specifically designed for high speed networks. It incorporates functionalities of both Network and Transport levels, as defined in the OSI reference model.

XTP provides a variety of *Types Of Service* (unacknowledged datagram, acknowledged datagram, connection, transaction, bulk data, and isochronous), plus Multicasting and Priority Delivery.

XTP has no addressing plan of its own. An XTP *Address Segment* consists of an *Address Descriptor*, which describes (among other things) what addressing plan is being used, followed by the actual source and destination addresses. This provides great flexibility, in that XTP can exist within any system that uses one of the addressing plans contemplated by the XTP Protocol Definition [11]. Some uses of this flexibility will be indicated later.

XTP has a variety of packet types, of which the FIRST packet, the DATA packet, and the CNTL packet are pertinent to this paper. The FIRST packet carries an Address Segment, and may also carry user data. The data packet carries user data, and the CNTL packet carries protocol control information.

In XTP, a *context* represents the information required to maintain one end of a conversation or *association*, and a *route* represents the path between the two hosts. An association is established when one context (the *initiator*) sends a FIRST packet to another context (unicast case) or contexts (multicast case). These contexts are called *listeners*. The FIRST packet carries an Address Segment, in addition to a *key*, representing the context, and a *route*, representing the path between the source and destination systems. In effect, the *key* is a short form for the transport level part of the address in the Address Segment and the *route* is a short form for the network level part of the address in the Address Segment. The FIRST packet is sent with the full address, and with a locally-significant *key* and *route*. The full address is used to choose a route through the system, and the *route* value is retained at each router to permit directing subsequent packets along the same path as the one used by the FIRST packet.

3.2 Generalized Addressing using XTP

Given the users' desire to use general addressing, a number of ways of forwarding packets within addressing domains and between addressing domains can be suggested. Some of these make use of XTP's Address Segment to achieve very general addressing; others are transitional measures that can be used with existing protocols to achieve a measure of connectivity between addressing domains.

1. If an end system is a participant in a particular addressing domain, which implies that it can reach its router directly, then the XTP FIRST packet can be sent to that router, containing an arbitrary address from that domain.

2. If an end system is running a version of XTP that is able to deal with a range of addressing domains, but is isolated from the router for the desired domain, then it can wrap the XTP packet in a network packet corresponding to the addressing domain that it lives in, and send it to a "smart" router (i.e., a router that can forward into the desired addressing domain).

3. If an end system (XTP or otherwise) is unaware of the possibility of multiple address families, then it is possible to define a set of "special" addresses within an addressing domain, corresponding to a routing service. At the point corresponding to that address, a value-added service can do the translation. Two possibilities are apparent:

   - Use the given address as a point of negotiation, expecting to get back a second address, to be used subsequently. Packets sent to this new address would be automatically forwarded to the desired destination address. This possibility can be used with any existing or future protocol.
   - If the protocol being used is XTP, it would be possible to send a two-address descriptor. The first address would be the address of the service, and the second address would be the actual desired destination. This possibility has the advantage that the *key* and *route* chosen can continue to be used for subsequent transmissions.

   Two-address descriptors are not currently defined in XTP. However, correct operation of this proposal depends only on the fact that the *routing service* interprets the data after the Address Segment as a second address, formatted as an XTP Address Segment.

The implementation of generalized addressing will require more "intelligence" in routers than is normally found in today's products. This "intelligence" may be seen as a value-added service that might be offered using the Intelligent Network [10] platform. XTP has the mechanisms necessary for multicasting of packets. Its ability to provide multicasting services will clearly depend on the existence of multicast addresses within the addressing plan in use, and on the provision of sufficient knowledge within XTP routers to meet the demands of multicast routing.

3.3 Routing in XTP

Since it encompasses the Network level, XTP must deal with switching of packets. In Figure 1, we present the procedure that is used to establish a communication and exchange of data between two XTP entities (called EXTP in the following) through an XTP switch.

Let EXTP1 be an XTP entity willing to communicate with a distant peer EXTP2. EXTP1 searches its context and route tables for free identifiers (say $k1$ and $r1$). Then it sends a FIRST PDU (with both $k1$ and $r1$ in its header) towards the XTP switch.

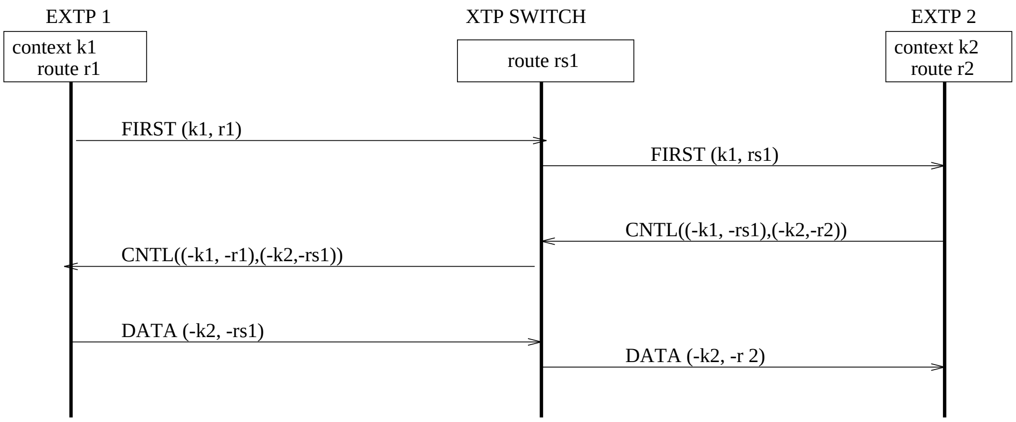

Figure 1: Routing through an XTP switch

When the switch receives this PDU, it searches in its route table for a free route identifier to use for this connection (say $rs1$). Using the Address Segment of the FIRST PDU the switch then selects an outgoing link for it, updates its route table, replaces $r1$ with $rs1$ in the FIRST PDU, and forwards it towards the next switch or the final EXTP.

When EXTP2 eventually receives the FIRST PDU, it also searches its context and route tables for free identifiers (say $k2$ and $r2$), and associates them with the incoming identifiers $k1$ and $rs1$. At this stage, EXTP2 can reply to EXTP1 with a PDU using the negative key and route identifiers ($-k1$ and $-rs1$) to allow direct access in the context table of its peer (with $-k1$) and route table(s) of the intermediate switch(es) (with $-rs1$).

On the other hand, the positive identifiers (used in the EXTP1 $\rightarrow$ EXTP2 direction) imply a sequential search in these data structures. So, to make the communication symmetric, EXTP2 can perform a key exchange. To do that, a CNTL PDU is sent back to EXTP1 through the switch. This PDU carries the values ($-r2$ and $-k2$).

- the switch then uses $-r2$ instead of $rs1$ in packets that it forwards to EXTP2, and substitutes $-rs1$ for $-r2$ in the CNTL PDU
- EXTP1 then uses $-rs1$ and $-k2$ to talk with its peer entity.

Then during the rest of the communication the switch can access its route table directly to substitute the right route identifiers in an incoming PDU and to forward it on the right outgoing link.

3.4 Routing in ATM networks

While the ATM switching strategy is currently very clearly defined, the routing strategy is not. Establishment of routes will be at subscription time, or dynamically, perhaps using an extended Q.931 [2]. Referring to Figure 2, the routers above each ATM switch have the

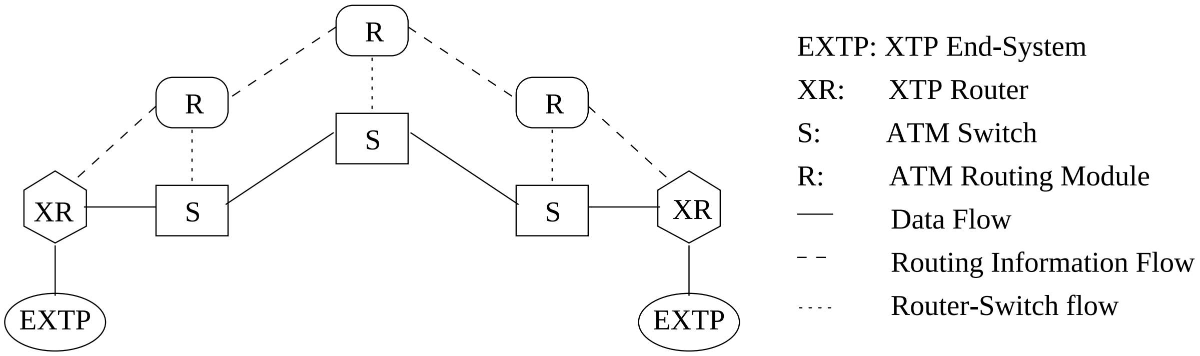

Figure 2: ATM Routing

responsibility to choose the correct path through an ATM network, and to set the switch tables within each ATM switch so that subsequent data cells will be correctly forwarded. Clearly, choosing and setting up the correct routes must precede switching of data cells. Routes might be completely established before data flows (connection-first model), or routes might be established "just in time", in front of the data flow (low-latency model).

ATM cell labels are divided into Virtual Path Identifiers (VPI) and Virtual Channel Identifiers (VCI). Although the VPI/VCI switching rules are actually more general, allocation of the VPI to correspond to the *route* values of XTP, and of the VCI to correspond to the *key* values of XTP represents a simple allocation rule. Note also that the boundary between the VPI field and the VCI is somewhat arbitrary—given the generality of the permitted substitutions at each switching node, any given ATM router can adopt whatever division into "contexts" and "routes" is useful to it.

3.5 Routing as a Value-added Service

The ATM routing strategy will, in the future, have to reflect the possibility of a user asking to be connected to "anyone", given an address from "any" of the supported addressing plans. This requires a mechanism for the representation of general addresses. XTP Address Descriptors provide exactly this mechanism: an Address Descriptor consists of a header that distinguishes which of the different possible address families is being used, followed by representations of the source and destination addresses, in accordance with the encoding rules of that address family.

The question that remains is how to structure the functionality of determining the routes to be used. In the network parts that are exterior to an IN service, routing must be done in the classical way, and the proposals given in section 3.2 are applicable. The value-added functions would have to be structured as user processes "listening" at a specific address. Within the part of the network that provides an IN [10] service, however, routing could be formulated as an offered service. Multicast routing clearly needs a significant level of support, which would be relatively easy to provide on this platform.

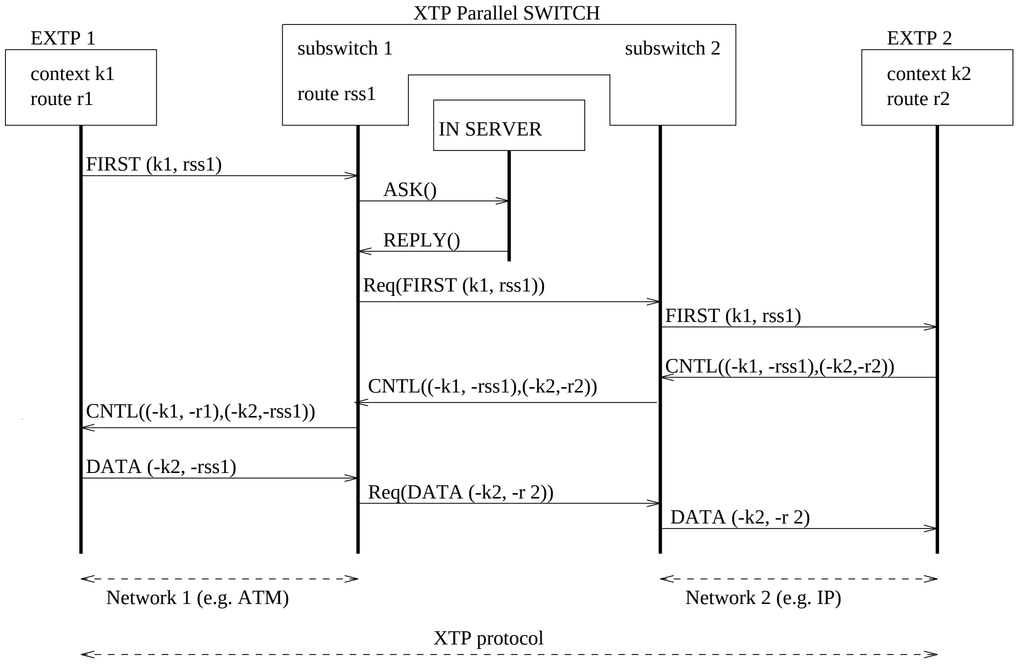

Figure 3: Routing through an XTP parallel switch (using the IN)

4 Implementing High Performance XTP Switches

4.1 The Need for a Parallel XTP Switch

In the context of heterogeneous high speed internetworking, XTP switches would be heavily used to implement the connectivity (see Figure 3 where an XTP PDU arrives on an ATM link and leaves on an IP one).

Thus to implement these switches, one must consider hardware and software architectures powerful enough to deal with this kind of high speed network traffic, both in terms of switching power and network management (billing for instance). The most common solution relies on special purpose multiprocessor computers with high I/O capabilities, which is a well-mastered technology. However when the various processors communicate through a shared memory, the total throughput of the switch is limited by the bottleneck of the bus access. These classical architectures would then have difficulties to deliver the desired throughput. Their capacity is not easily customizable, so various versions of the software are needed when a range of performances is to be provided. Furthermore, these special purpose multiprocessor computers are built in few instances (from several tens to several hundreds), have an *ad hoc* operating system (most of the time), and a poor development environment. So they are expensive to build, deliver and maintain.

So we think that general purpose Distributed Memory Parallel Computers (DMPC) with standard operating systems are an interesting alternative to implement these switches. Since there is no common bus in a DMPC, the associated bottleneck problem vanishes. By construction, DMPC exhibit easily customizable performances, and their operating systems and programming methods and environments are maturing, so in the long run, using this kind of parallel computer would be a very cost effective solution.

4.2 A Parallel XTP Switch in the IN Context

The parallelization of an XTP switch as defined in the previous section has been described in reference [13]. Here we just present an overview of the design of our parallel XTP switch.

The sequential XTP switch has been broken down into a set of independent subswitches (one for each available processor), each one dealing with a subset of the incoming and outgoing links (see Figure 3). Internal communication between subswitches occurs when the outgoing link for a communication is not located on the same processor as the one that had the incoming link (as in the Figure 3 example). In fact, the table providing the correspondence between EXTP routes and outgoing links is distributed among the available processors. When a processor owns the information locally, it computes the location of the relevant outgoing link (using the IN server when needed), and fully processes the incoming PDU. Then if the required link is not attached to this processor, it simply requests the processor responsible for this outgoing link to send the outgoing PDU on that link (this is the role of the internal messages called Req on the Figure 3).

On the other hand, if the processor receiving an XTP PDU does not own the switching information, it is however able to compute the number of the processor owning this information, and then it forwards the incoming XTP PDU to this processor, as if it were coming there directly from the outside.

An interesting property of our parallelization method is that it is scalable: the actual number of processors allocated to the parallel switch need not be known until runtime, thus the very same program is used regardless of the number of processors.

5 Conclusion

The networks of the world have developed as isolated entities, corresponding to particular addressing domains. The development of parametric addresses in XTP provides a mechanism for introducing a single transport protocol that will be able to unify these formerly disparate domains.

While the increased capability of the routers that will be necessary at the transitions between different media and at the boundaries between addressing domains, could be formulated in terms of specific programming of these routers, it is likely that use of the general platform provided by the Intelligent Networks service will result in a more comprehensive service offering.

We have experimented with the design of highly parallel XTP routers, and future studies will include a simulation of the operation of just such a router in the ATM context.

Acknowledgements

Financial support for J.W. Atwood, A. Das and M. Nour was provided in part by the Natural Sciences and Engineering Research Council of Canada, and in part by the FCAR Fund of Québec. Financial support for J.-M. Jézéquel was provided by the CNRS. Financial support for meetings was provided to A. Das and J.-M. Jézéquel by the Coopération Franco-Québequoise program.

References

[1] ISO 7498: Information Processing Systems - Open Systems Interconnection - Basic Reference Model. International Organization for Standardization.

[2] Jean-Yves Le Boudec. The Asynchronous Transfer Mode: a tutorial. *Computer Networks and ISDN Systems*, 24:279–309, 1992.

[3] RFC 791: Internet protocol. DARPA, 1981.

[4] Internet protocol address extension. Internet Activities Board.

[5] MOSPF. Working paper obtained from gated.cornell.edu.

[6] RFC 1112. Internet Activities Board.

[7] OSI Group NSAP Addressing. Proposed Work Item for JTC1/SC6/WG2.

[8] Radia Perlman. *Interconnections.* Addison-Wesley, 1992.

[9] *CCITT Recommendation I.451 (Q.931), ISDN User Network Interface Layer 3 Specification for Basic Call Control, CCITT Blue Book*, 1988.

[10] José M. Duran and John Visser. International standards for intelligent networks. *IEEE Communications Magazine*, pages 34–42, 1992.

[11] XTP Forum, 1900 State Street, Santa Barbara CA 93101. *XTP Protocol Definition, Version 3.6*, jan 1992.

[12] W. Timothy Strayer, Bert J. Dempsey, and Alfred C. Weaver. *XTP: The Xpress Transfer Protocol.* Addison-Wesley, 1992.

[13] J.-M. Jézéquel. Parallélisation d'un routeur XTP. In *Actes du colloque CFIP'93 sur l'ingénièrie des protocoles, Montreal.* Hermès, September 1993.