WORKSHOP ON DATA COMMUNICATIONS

SEPTEMBER 15-19, 1975
CP-76-9

CP-76-9

WORKSHOP ON DATA COMMUNICATIONS

SEPTEMBER 15-19, 1975

International Institute for Applied Systems Analysis
2361 Laxenburg, Austria

PREFACE

With many existing computer networks and many more planned to come into operation, new problems are beginning to appear concerning their inter-connection. On September 15-19, 1975, a Workshop on Data Communications was held at Laxenburg, jointly sponsored by the International Federation for Information Processing (IFIP) and the International Institute for Applied Systems Analysis (IIASA). Participants discussed problems of inter-connecting computer networks, and made suggestions for new standards in this area. This report contains papers presented at the Workshop.

Table of Contents

A Survey of Problems in Distributed Data-Bases

Jean Le Bihan

1. INTRODUCTION

This decade is seeing the explosive growth of computer networks [1,10,11]. Although the external characteristics of these networks are somewhat different, most of them are built on the same architecture [9]. We can notice three layers:

a) a pure communication layer generally using a packet-switching technique [6];

b) an end-to-end protocol layer controlling the transfer of information between remote programs [13];

c) one or many applications layer according to the user's requirements, the purpose of which is to translate and process the information carried by the lower layers [5].

These lower layers have constituted, until now, the most important part of research and development activities for the networks people. Today, some proposed solutions are experimented with and standards are ready to be agreed upon. Also, endeavors are put upon the application level.

A computer network may be used for several utilizations, but the most strongly needed application seems to be data sharing and, more particularly, distributed data-bases.

In the CYCLADES project, some research has already been undertaken on this subject, and a new project sponsored by SESORI, "Service de Synthèse et d'Orientation de la Recherche en Informatique" - I.R.I.A., is being launched in order to federate the various research efforts in that field, start new studies, make experimentations, and define standards and products.

This paper is an outline of some ideas resulting from the discussion of a "distributed data-bases working group." It points out the different problems that have to be faced and could serve as a basis for discussion.

2. A DATA-BASE SCHEME

For describing clearly the various types of data-base distribution, it is necessary to have a simple scheme (Figure 1). In this scheme, we can discriminate three parts [2]:

a) The users (terminals or programs);

b) The data-base management system (DBMS) restricted here to two functions: a "utilization" function which interfaces the users and provides syntactic and semantic analysis, and a "data access" function which interfaces the data;

c) The data itself (on physical devices).

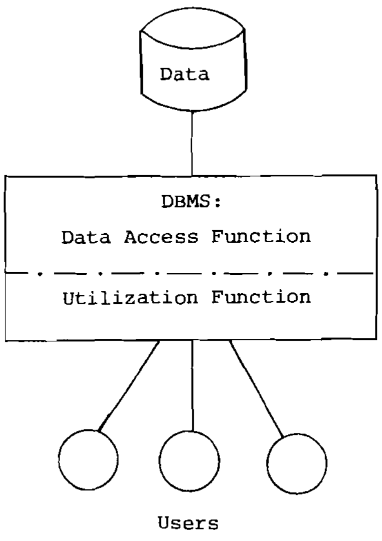


Figure 1.

This naive diagram immediately suggests two main classes of distributed data bases:

a) First class can be seen as a DBMS sharing among network users (Figure 2), the network being introduced between users and DBMS (data remains closely linked to a DBMS);

b) Second class can be seen as a data sharing among network DBMS (Figure 3), the network being introduced between DBMS and the data.

In fact, the distributed data-base structure is (or will be) a mixture of these two types.

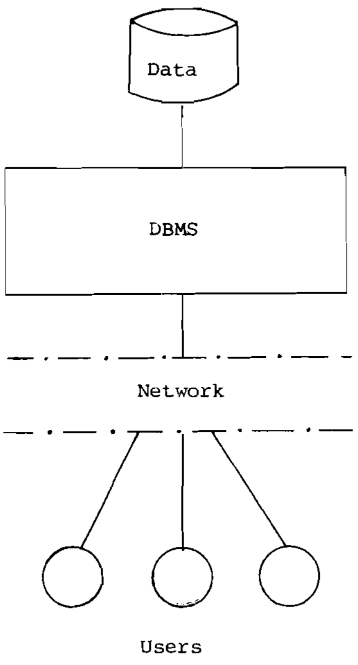


Figure 2.

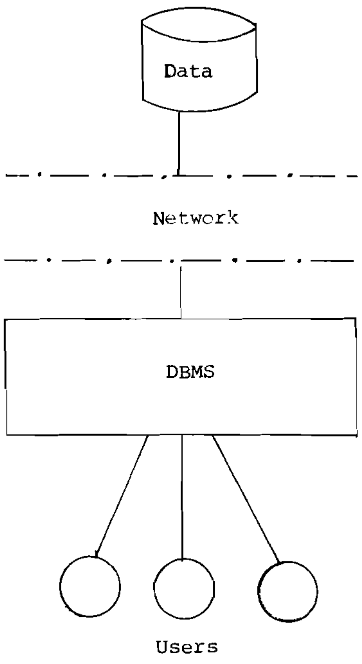


Figure 3.

3. THE DBMS SHARING

As the different couples (DBMS and data) are strictly independent or accept relations, we have two dissimilar levels of problems.

3.1 The Data-Bases are Independent

These DBMS can be heterogeneous. They can be used as regular network services (such as time sharing or remote batch system on a network). If a user wants to access many data-bases, he must localize them, question the interesting bases, and merge the answers. It is not very simple. In order to help him, we can try to automatize this manual procedure. We are then led to solve the following problems:

a) How is a data-base localized on a network? Is a centralized catalogue necessary? If so, on one or many nodes? Are broadcasting techniques usable? What is the cost?

b) What may be the standard language for the user's inquiries? Can the Codasyl or ISO proposals be extended for networking? Is the chosen language easily translatable into various DBMS languages? [8].

c) How does one process multibase inquiries? Splitting the user's inquiries requires information about the concerned data-bases; where can one find that information? (See localization problem.) Each subinquiry has to be expressed in a standard form (see the language problem) and the answers merged in order to give a synthesized response; in what manner should this be done? [7].

The simple scheme (Figure 3) must be complicated by adding these new functions of localization, translation, and inquiries preprocessing which take place as a new layer between users and DBMS (Figure 4). The existing DBMS can be used without internal modifications, and we notice the need for an "inquiry station" as a consequence of DBMS' independence.

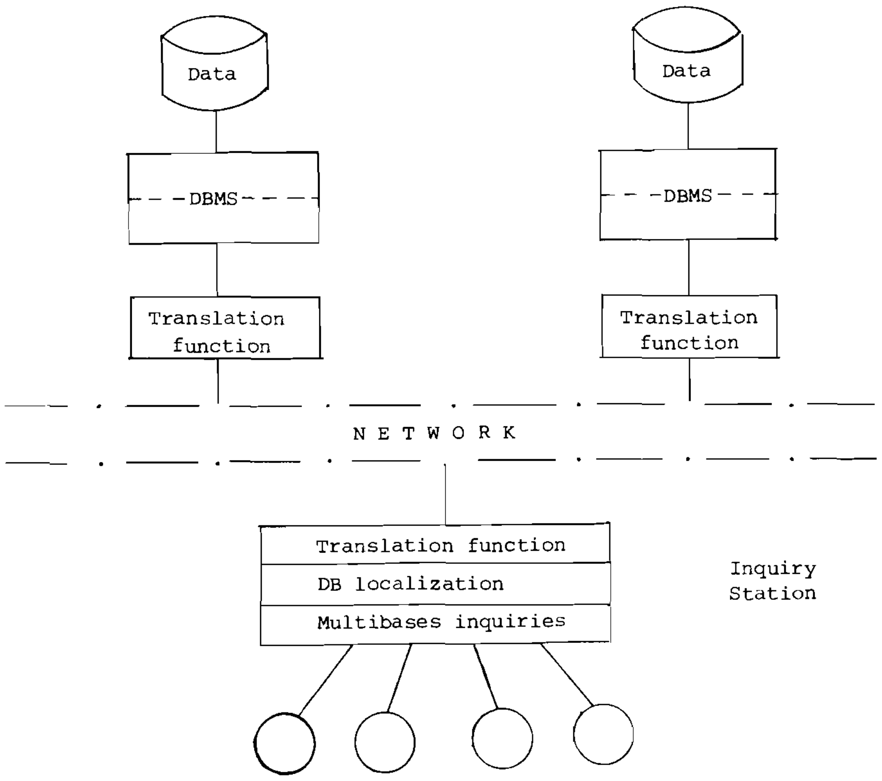


Figure 4. Data-base scheme (complex).

3.2. The Data-Bases are Interrelated

In the previous case, the user could work on a distributed data-base, but the structure was not materialized anywhere; it was only in his brain. In this new hypothesis, we consider that each data-base is made from the merging of local data with various external data; we call this a "virtual data-base" (VDB). This VDB is described by a scheme (or a sub-scheme) similar to that involved in classical data-bases [12].

In this case, the localization mechanisms must be located at the "data access" level, instead of at the user level as in the previous case. A DBMS-DBMS protocol must be defined: if DBMS are interfaced at the "utilization" level, we again find the external approach of the previous paragraph; if they are interfaced at "data access" level, a new class of problems appears--namely the data sharing among DBMS.

The inquiry station disappears; splitting of inquiries is automatically performed by DBMS.

4. DATA SHARING (Figure 3)

Until now, the two parts (DBMS and data) were seen as a whole. If we break the band and introduce the network, we have to make a surgical operation on DBMS. But where should we cut?

A first choice may be to cut at the physical access to the data. Then we have to define such as a network direct access method [4]. It is a simple solution when we consider the access of one DBMS to various data-bases. This solution can also be convenient for read only access to various data-bases by several DBMS. But it is not possible to implement control functions allowing simultaneous updates.

The second choice may be to cut at a logical access level within DBMS, perhaps at the interface between the utilization and the data access function (Figure 5). We introduce then the notion of "access method to structure data" [3], and again find the DBMS-DBMS protocol seen at data access level. In the same manner, we have introduced the "inquiry station"; we can talk of "data station."

5. CONCLUSION

Although this paper is only a rapid survey of the distributed data-bases, I hope it has clarified, to some extent, this new research field.

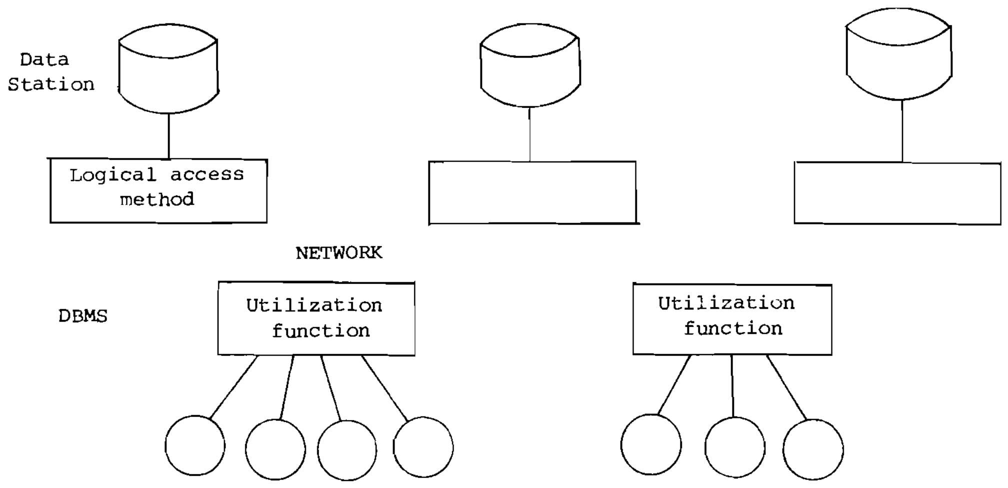


Figure 5.

References

[1] Barber, D.L.A., The European Computer Network Project, *ICCC*, Washington, D.C., 1972.

[2] Bulletin de liaison du club Banques de données, *IRIA* 13 (1974).

[3] Chupin, J.C., Control Concepts of a Logical Network Machine for Data Banks, in the *Proceedings of Information Processing 74*, Amsterdam, North Holland, 1974.

[4] Chupin, J.C., and S. Seguin, A Network Direct Access Method, in the *Proceedings of Information Processing 74*, Amsterdam, North Holland, 1974.

[5] Crocker, S.D., et al., Function Oriented Protocols for ARPA Computer Network, in AFIPS Proceedings, *SJCC* (1972).

[6] Davies, D.W., Principle of Packet Switching, *First European Workshop on Computer Networks*, Arles, France, 1973.

[7] Lagasse, J.P., et al., *A Processing Network with User Data Bases Interactive System*, Note Technique, Université Paul Sabatier, Toulouse, 1975.

[8] Le Bihan, J., *Specifications d'une communication entre des systèmes SOCRATE hétérogènes sur le réseau CYCLADES*, Bibliothèque CYCLADES DAT 500, 1973.

[9] Pouzin, L., Network Architectures and Components, *First European Workshop on Computer Networks*, Arles, France, 1973.

[10] Pouzin, L., *Presentation and Major Design Aspects of CYCLADES Computer Network*, Third Data Communications Symposium, IEEE, Tampa, Florida, 1973.

[11] Roberts, L.G., and B.D. Wessler, Computer Network Development to Achieve Resource Sharing, *SJCL* (1970), 543-549.

[12] Rochfeld, A., *Banques de données virtuelles*, Rapport de recherche, 73, SEMA, France.

[13] Zimmermann, H., and M. Elie, *Transport Protocol. Standard Host-Host Protocol for Heterogeneous Computer Networks*, IFIP WGI, INWG 6 1, 1974.

Host-Host Protocols and Hierarchy

André A.S. Danthine

1. INTRODUCTION

The communication between distributed processes may be considered an extension of the communication between local processes, based on an interprocess communication (IPC) facility. However, in the distributed case, the IPC facility is, itself, distributed and may be represented as two local processes which will be called transport station (TS), associated with a virtual link (Figure 1).

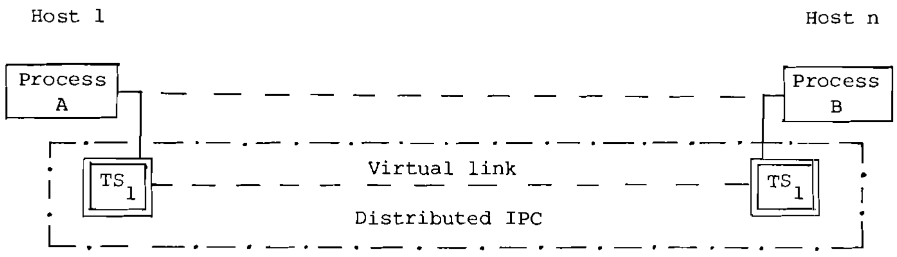


Figure 1.

The basic primitives are essentially the following:

a) OPEN (..,..,..) and CLOSE (..,..,..), used for creation and destruction of the virtual link;

b) SEND (..,..,..) and RECEIVE (..,..,..), used for message transmission;

c) DISABLE (..,..,..) and ENABLE (..,..,..), used for control of message flow between processes and the IPC;

d) UNSEND (..,..,..) and UNRECEIVE (..,..,..), used also for control;

e) STATUS (..,..,..), used by the process in order to know the state and context associated with the virtual link.

In this working paper, we look at a hierarchical decomposition of the distributed IPC. We shall assume that the physical support is a packet-switched network (PSN), and we shall take into account the fragmentation and the multiplexing which take place in the IPC of Figure 1.

2. FUNCTIONS OF THE PROTOCOL

Ideally, we want to have a perfect virtual link of level n between distant processes A and B allowing for communication by way of the exchange of messages of maximum text length L_m. By perfect virtual link, we mean that the content of each message is preserved, the order of messages is not modified, and we have no lost messages or duplicates.

The virtual link of level (n-1) between two TS_1 (Figure 2) also allows the exchange of messages of text length L_m. If this virtual link is also ideal, the function of the protocol of the TS_1 will be simple and will provide basically the required creation and destruction facilities.

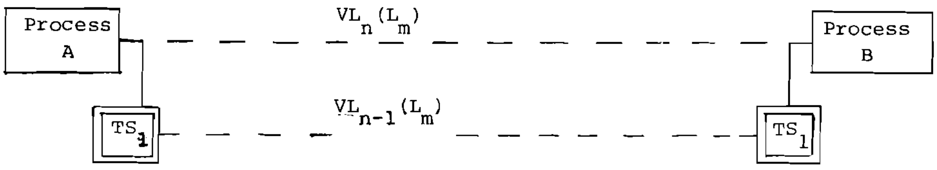


Figure 2.

If the virtual link is not ideal, the protocol TS_1-TS_1 must be designed in order to restore the expected characteristics of the VL_n.

In the Cyclades network [6], the following properties of VL_{n-1} are assumed:

a) L_m = 32,000 bytes;

b) The content of a message delivered by the VL_{n-1} to the receiving TS_1 may be corrupted;

c) Order of messages is not preserved;

d) Loss or duplicates are possible.

The mechanisms introduced at this level in Cyclades are based on a time-out and a numbering scheme modulo k of the messages (letters), which allow error control on arrival and flow control. A CRC software on letters allows for an error control on content.

As for the creation of the virtual link, it is based in Cyclades on a two-way handshake and is explicit, i.e., no communication is possible until the end of the virtual link establishment.

Methods have been proposed for detailed representation of such a protocol [4,5]; but they will not fit the space limitation of this paper.

3. FRAGMENTATION AND REASSEMBLY

In a PSN, the maximum text length L_m does not, in general, fit the maximum text length of packets. We must, therefore, introduce in the IPC facility an internal process for fragmentation and reassembly (FR) (Figure 3).

As for the communication between the two processes A and B, we need a virtual link, VL_{n-2}, between the two distant FR processes which will be implemented through two TS_2 and a virtual link VL_{n-3}.

The same problem applies at this level. What are the properties required on VL_{n-2}? Taking into account the properties of VL_{n-3}, what are the functions implemented at the TS_2 level?

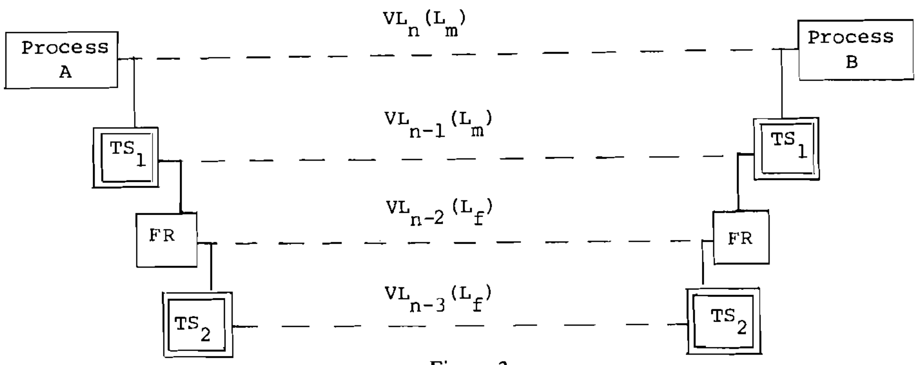


Figure 3.

Let us note that the FR may be considered a process attached to the VL_{n-1} and, therefore, there is no need for creating a new VL; thus the previous identification may be used.

In Cyclades, the only additional function is a numbering scheme attached to a given letter which allows the reassembly.

It is interesting to note that nothing (but efficiency) prevents one from introducing at this level and on the fragments, error control on arrival, flow control, and even error control on content.

4. MULTIPLEXING

In a computer network, we have in general several processes requiring the services of the distributed IPC; since a unique PSN is used, a multiplexing-demultiplexing process (MD) must take place (Figure 4).

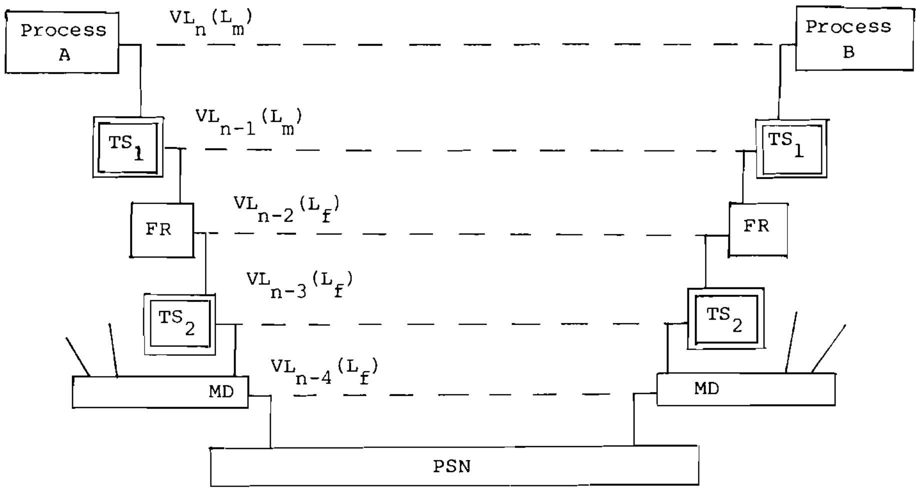


Figure 4.

As in Cyclades, it is assumed that the VL_{n-3} does not provide for the ordering of the fragments and may introduce losses or duplicates; the same is true for VL_n. The main function of the MD is to attach to the fragments the address of the destination MD which is, in fact, a part of the address of the connected processes.

In Cyclades, the VL_{n-4} is considered permanently established and is implemented through the PSN, Cigale, which provides only a datagram service.

5. CK PROTOCOL

Figure 4 is still valid to describe the Cerf-Kahn (CK) protocol [2,3], but the characteristics of the VL differ in the following manner:

a) On VL_n and VL_{n-1}, L_m is not fixed, i.e., the message may be considered an infinite string of characters;

b) On VL_{n-1}, ideal transmission is assumed, i.e., no corruption, no disorder, no loss, and no duplicates. At the TS_1 level, the only function of the protocol is to create the virtual link. Here the mechanism is implicit (i.e., done at the beginning of the communication) and based on a three-way handshake).

c) As the transmission of fragments on VL_{n-2} is also ideal and the VL_{n-3} has the same properties as Cyclades, the protocol between the two TS_2 has the complete set of functions--error control on arrival and flow control through a numbering scheme modulo k' based on bytes number, and error control on content with a software CRC.

d) As for Cyclades, the VL_{n-4} is considered permanently established, and the PSN has to provide only a datagram service.

6. ARPA PROTOCOL

In the ARPA protocol [1,7], Figure 2 is valid with L_m = 8,000 bits, but the VL_{n-1} is also assumed to be ideal. The only function of the TS_1-TS_1 protocol is to create the VL through a two-way handshake procedure. However, here the multiplexing function takes place before the fragmentation and reassembly (Figure 5).

On the VL_{n-2}, the communication between two hosts is also assumed to be ideal and since the VL_{n-3} does not keep the order of the messages and may introduce losses or duplicates, the main functions of the TS_2 protocol are error control on arrival through a cyclic numbering scheme, and flow control by the request for buffer space and the ALL and RFNM control messages. Here the VL_{n-3} may be considered created just before the transmission time.

Here the fragmentation process adds only a packet number, and the VL_{n-4} is implemented through a datagram service. It is, however, important to note the boundary of the ARPANET in Figure 5.

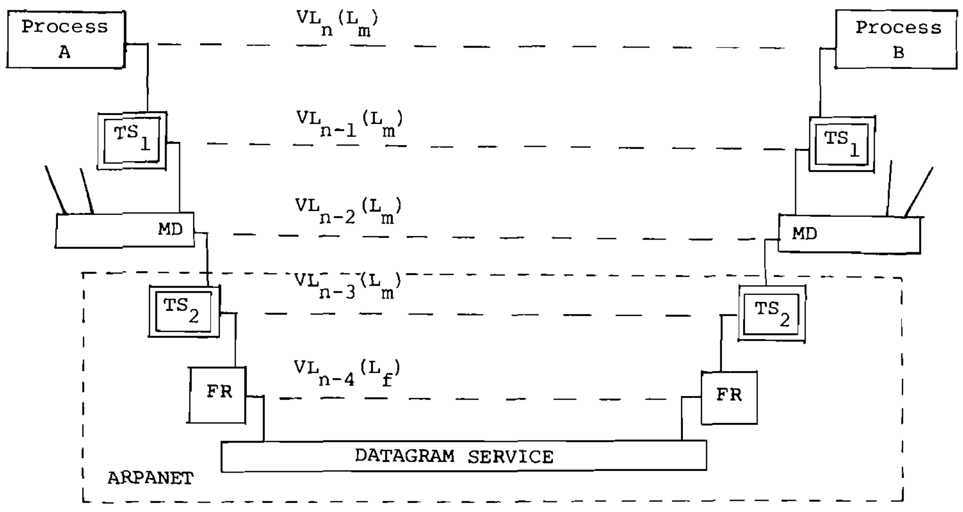


Figure 5.

7. CONCLUSION

This global presentation of three HOST-HOST protocols shows the basic hierarchy chosen by the designers in order to implement an ideal virtual link between distributed processes from the datagram service of a PSN. It will be possible to define other protocols or to add functions at some level of existing ones. However, the discussions regarding the advantages and drawbacks of each design do not, at the present time, appear to be supported by adequate performance studies.

References

[1] Bolt, Beranek, and Neuman, eds., Interface Message Processor, Specifications for the Interconnection of a Host and an IMP, Report No. 1822, (NIC 07958).

[2] Cerf, V.G., et al., Specification of Internet Transmission Control Program, *International Network Working Group*, INWG G072, December 1974.

[3] Cerf, V.G., and R. Kahn, A Protocol for Packet Network Intercommunication, *IEEE Transmission on Communication*, 1974, 637-648.

[4] Danthine, A., and J. Bremer, Définition, représentation et simulation de protocoles dans un contexte réseaux, *Journal International Mini-ordinateurs et Transmission de donn de données*, AIM, Liège, January 1975.

[5] Danthine, A., and J. Bremer, Communication Protocols in a Network Context, *ACM Interprocess Communication Workshop*, Santa Monica, March 1975.

[6] Garcia, C., et al., Spécifications de réalisation de la station de transport ST2 portable, *Réseau Cyclades*, SCH 536.1 (February 1975).

[7] McKenzie, A.A., Host-Host Protocol for the Arpa Network, *Network Working Group*, January 1972, (NIC 08246).

Performance Evaluation of Adaptive Control Policies for the Broadcast Channel

Banh Tri An and E. Gelenbe

1. INTRODUCTION

Among the schemes used for obtaining the transmission of packets of data between a set of terminals and a control computer, or among a set of computers, one of the simplest in appearance is the one first used in the ALOHA system [10]. Here a simple radio or satellite channel is used for transmitting packets from terminals, with the resulting disadvantage being that packets, whose transmission overlaps in time, are unintelligible and, therefore, have to be retransmitted until success is achieved. In the slotted version of this method, time is put in numerical orders in slots of unit time corresponding to the duration of transmission of a packet (each packet being of equal length). Terminals are then allowed to transmit packets only at the beginning of a slot. Many of the basic properties of these channels have been derived by Abramson [1] in a simple manner; he showed, in particular, that the effective throughput of the slotted broadcast channel (in successful transmission per unit time) is twice as high as that of a totally random transmission scheme. Further analysis and simulation of these matters and a classification of the methods for broadcast control can be found in [6,7].

The slotted (or unslotted) broadcast channel is unstable in the following sense: for a large enough ensemble of terminals, the steady-state throughput will become negligibly small, and almost all the terminals are in a blocked state, i.e., they are trying to retransmit a packet which they have previously transmitted unsuccessfully. This property has been widely observed. The first proof of this behavior, based on showing that a Markov chain model of the slotted channel is not ergodic, was given in [5]. In [6,7,9], various policies for controlling the channel have been suggested, so as to assure stability and maximize throughput. A more recent paper [4] is devoted to obtaining general conditions for channel stability and to deriving the optimal stationary retransmission policy for the slotted broadcast channel.

In this paper, we first recall some results from [3,4]; these are used to formulate some practical adaptive policies for the channel which are then evaluated in a series of simulation experiments.

In the last part of the paper, we examine some heuristic control procedures suggested by Metcalfe [9]. Since their mathematical analysis has failed so far, and because other quantitative results concerning their performance are unavailable, we conduct simulations to determine the associated stability problems and to compare their effect on channel behavior with that of the optimal policies.

2. ADAPTIVE POLICIES FOR CHANNEL OPTIMIZATION

The purpose of this section is to examine practical realizations of the theoretical optimal policies suggested in [2,4]. We first recall some definitions and results from [4,5]. These results would allow the optimization of channel behavior if the state of the channel, in this case the number of blocked terminals, were known exactly at all instants of time. Since this is not the case, the state must be either estimated indirectly, or some measurable parameter must be determined whose value is known when the channel is optimized via the optimal control policy. It is the latter approach which we shall take in the sequel.

A terminal accessing the channel can be in one of three states: it is *blocked* if its last packet transmission has been unsuccessful; it is in *channel state* if it is currently transmitting or retransmitting a packet; and it is *active* if it is in neither of these two states. Arrivals of previously nontransmitted packets occur from the active terminals. We shall consider the *open model* in which these arrivals constitute a Poisson process[1] of rate λ. In the *closed model*, the total number of terminals (active plus blocked plus those in channel state) is finite; the epoch between the instants of time at which a terminal becomes active and that at which it enters channel state is negative exponentially distributed of parameter λ.

Definition 1

Let $N(k)$ be the number of blocked terminals at the beginning of the k-th slot. The broadcast channel is *stable* if and only if $\lim_{k\to\infty} \Pr[N(k) < j] = 0$ for all $j < \infty$.

Clearly, there can be at most one successful packet transmission in a slot. As far as retransmissions of packets from blocked terminals are concerned, in this section we assume that each of the blocked terminals retransmits independently of all other terminals. Any given blocked terminal has a probability f of retransmitting during a slot; f is identical for all blocked terminals and is a function of the total number of blocked terminals $N(k)$ during the k-th slot.

[1]Poisson arrivals are justified for a large number of active terminals due to the Palm-Khintchine theorem concerning the superposition of independent renewal processes.

The following result is proven in [3] and in somewhat less general form in [4].

Theorem 1

Let d_n be the probability of a successful transmission in a slot at the beginning of which the number of blocked terminals is n. Assuming an open model, the channel is stable if,

$$\lambda < \limsup_{n\to\infty} d_n \quad ,$$

and unstable if,

$$\lambda > \limsup_{n\to\infty} d_n \quad .$$

For the open model, we have,

$$d_n = nf(1-f)^{n-1} e^{-\lambda} + \lambda e^{-\lambda}(1-f)^n \quad ,$$

while, for the closed model, the corresponding quantity, if M is the total number of terminals, will be,

$$d_n = nf(1-f)^{n-1}(1-g)^{M-n} + (M-n)g(1-g)^{M-n-1}(1-f)^n \quad ,$$

where $g = 1-e^{-\lambda}$ is the probability that an active terminal will transmit in a slot, assuming that slot duration is of unit time. A result analogous to Theorem 1 can be established for the closed model for $M\to\infty$. Here we restrict ourselves to problems of the open model; this constitutes a reasonable assumption for studying systems in which M is large.

Another important result established in [4] is:

Theorem 2

For the open model, the retransmission probability for each blocked terminal

$$f^*(n) = \frac{1-\lambda}{n-\lambda} \quad ,$$

maximizes channel throughput. It sets $d_n = e^{-1}$.

This result indicates that to maximize channel throughput, each blocked terminal should have perfect information concerning the arrival rate of packets from active terminals λ and the number of blocked terminals n. In practice, neither can be known since each individual terminal is unaware of the state of other terminals. Theorem 2 has a simple consequence for sufficiently large values of n; namely, if we call G the total channel traffic equal to the expected total number of packets transmitted in a slot, we will have,

$$G \simeq \lambda + f(n) \cdot n \quad , \tag{1}$$

and at the optimum $(f(n) = f^*(n))$ we have,

$$G^* \simeq \lambda + \frac{1-\lambda}{n-\lambda} \cdot n \quad , \tag{2}$$

which is,

$$G^* \simeq 1. \tag{3}$$

Thus, if we know that the total number of blocked terminals is large enough so that,

$$nf(n) \simeq \Sigma_i^n \binom{n}{i} i[f(n)]^i [1-f(n)]^{n-i} \quad ,$$

then it suffices to regulate f(n) in such a manner that G is maintained close to one; we will know then that f(n) is close to $f^*(n)$ without knowing the value of λ and of n. The adaptive policies we examine in this section are based on this simple observation.

One should note, however, that such a policy may not maximize channel throughput for small values of n.

A simple implementation of the optimal control policy is based on (3). Assuming a Poisson traffic in the channel (this is reasonable if the total number of blocked terminals is large) we have,

$$G = \frac{p_1}{p_0} = \frac{Ge^{-G}}{e^{-G}} \quad , \tag{4}$$

where p_i is the probability of i packet transmissions (from both active and blocked terminals) in a slot. In an interval of J consecutive slots, let U(J) be the number of slots in which successful transmission of a packet has taken place, and let V(J) be the number of slots during which there has been no transmission. Then an estimate of G is,

$$\hat{G}_i = \frac{U(J)}{V(J)} , \qquad (5)$$

where U(J) and V(J) are evaluated over slots i-1,...,i-J. If $\hat{G}_i > 1 + \varepsilon$, then each blocked terminal sets $\hat{n}_i = \hat{n}_{i-1} + x$, and if $\hat{G}_i < 1 - \varepsilon$, they set $\hat{n}_i = \hat{n}_{i-1} - x$, where $\hat{n}_i$ is the total number of blocked terminals used in computing the retransmission probability,

$$f_i = \frac{P}{\hat{n}_i} ,$$

at the i-th slot. The quantities ε and x can be determined by experimentation. In Table 1, we see the effect of ε on the total number of blocked terminals after 2000 slots starting with 200 blocked terminals; a "best" value of ε seems to exist. In Figure 1, we see the evolution of the number of blocked terminals with time for the same simulation experiments. We have also experimented with variable step (x) policies with less success.

Table 1. Number of blocked terminals after 2000 slots.

1-ε	1-ε	Simulation I	Simulation II	Simulation III	Average Value
1.0	1.0	78	95	42	72
1.1	0.9	42	112	140	98
1.2	0.8	16	70	16	34
1.3	0.7	86	72	60	73

3. DISTRIBUTED CONTROL POLICIES

These policies have important practical interests because the information used for the control of the channel are available at each terminal. They are based on ideas suggested by Metcalfe [8].

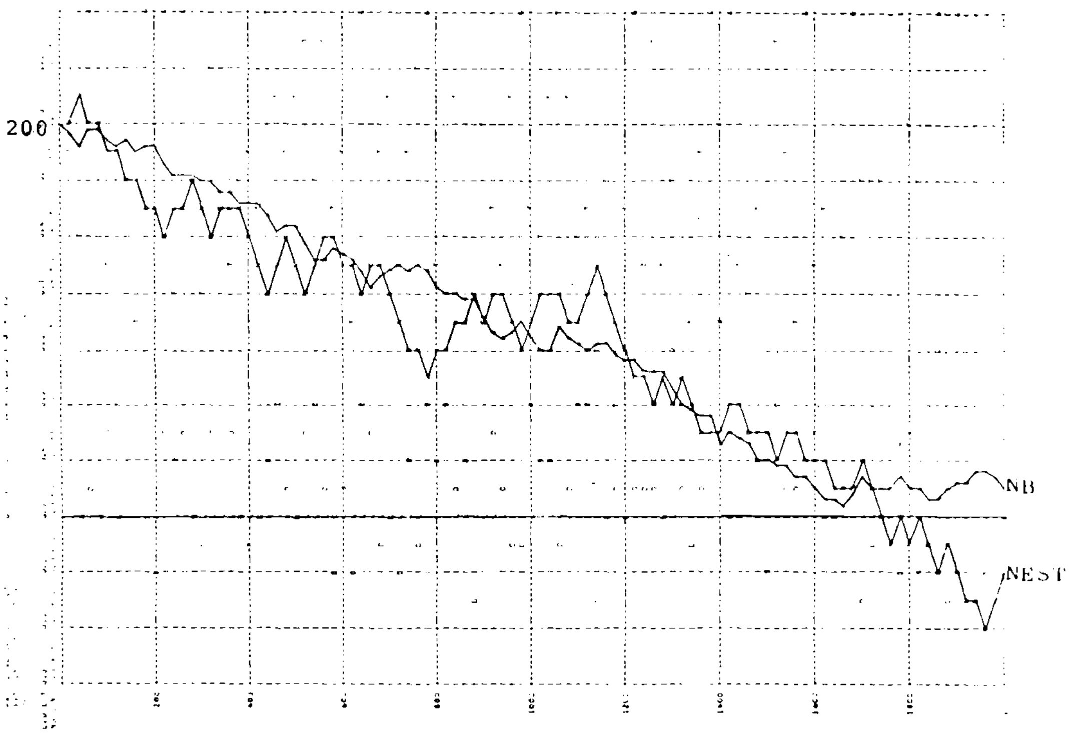
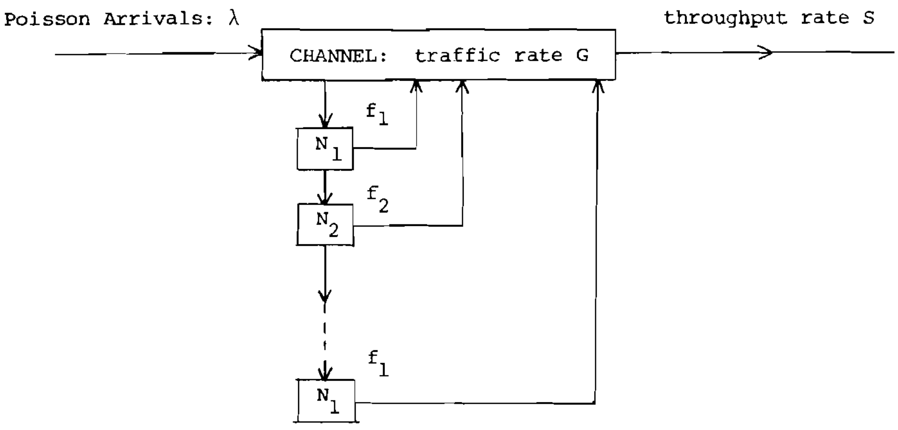


Figure 1.

Each terminal has a counter to update and register the number of times it has transmitted its current packet unsuccessfully. The functional scheme may be seen in the following diagram.



The terminals blocked for the i-th consecutive time belong to class i and retransmit their backlogged packet with probability f_i, such that:

1° $0 < f_i < 1$, $\forall\, i \quad 1 \leq i \leq l$

2° $f_{i+1} < f_i$

3° $\lim\limits_{i\to\infty} f_i = 0$.

The state of the channel must then be characterized by an l-vector:

$$N(k) = \begin{bmatrix} N_1(k) \\ N_2(k) \\ \vdots \\ N_l(k) \end{bmatrix}$$

, l finite or not is the number of the highest class present during the k-th slot,

denoting,

$$N'(k) = \sum_{i=1}^{l} N_i(k) \quad .$$

The definition of instability then becomes,

$$\lim_{k\to\infty} \Pr\,[N'(k) < \infty] = 0 \quad .$$

We shall try two simple expressions for f_i:

a) $f_i = \dfrac{p}{i^E}$; $0 < p < 1$, E constant > 0, and

b) $f_i = \dfrac{p}{C^i}$; $0 < p < 1$, C constant > 1.

We can verify that these two forms satisfy the three above conditions.

Simulations conducted with $f_i = \frac{p}{i^E}$ or $\frac{p}{C^i}$ confirm the unstable nature of the channel with input rate $\lambda < 1/e$.

With input rates smaller than 1/e, well-chosen values of p, E, and C lead to stability. Let us consider these two cases separately.

Expression (a): $f_i = \frac{p}{i^E}$

We can first notice that if $E >> 1$, the greater part of the retransmissions is due to blocked terminals in class one. Applying the qualitative condition $G = 1$, we obtain:

$$\lambda + N * p \simeq 1 \rightarrow p \simeq \frac{1-\lambda}{N_1} \quad .$$

We observe in the simulations a steady-state value for N_1 in each case; this helps us to choose p. As N_1 seldom increases over 10, we take $0.02 < p < 0.1$.

On this range of values for p, stability is achieved in our simulation for $1.5 \leq E \leq 2$. Beyond 2 or 3 we observe the same phenomenon as in the case f(N) such that $\lim_{N\to\infty} N*f(N) = 0$, i.e., insufficient retransmissions. The results are shown in Figures 2a, 2b and 2c.

Expression (b): $f_i = \frac{p}{C^i}$

We can make the same remarks as before and take $p = 0.065$ (as $\lambda \simeq 0.3$). As the exponential increases faster than a power, the relative importance of the first classes is emphasized.

The static and dynamic behavior of the channel is simulated. It is seen in Figures 3a, 3b and 3c that for $\lambda = 0.3$ and $p = 0.065$, channel stability is achieved with $1.6 \leq C \leq 2$.

We terminate by enumerating two drawbacks of these policies:

a) Theoretical study seems to be difficult, leading to untractable mathematics; and

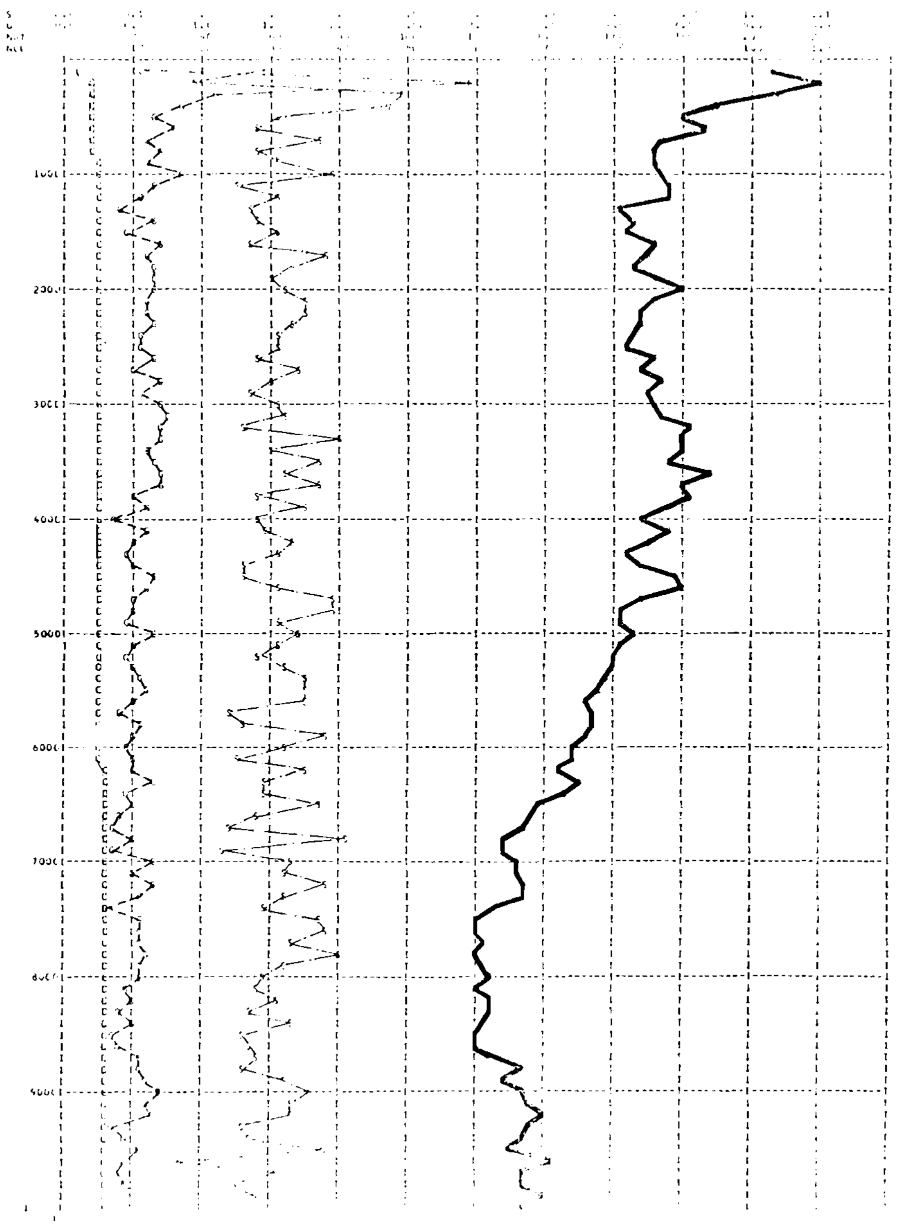


Figure 2a.

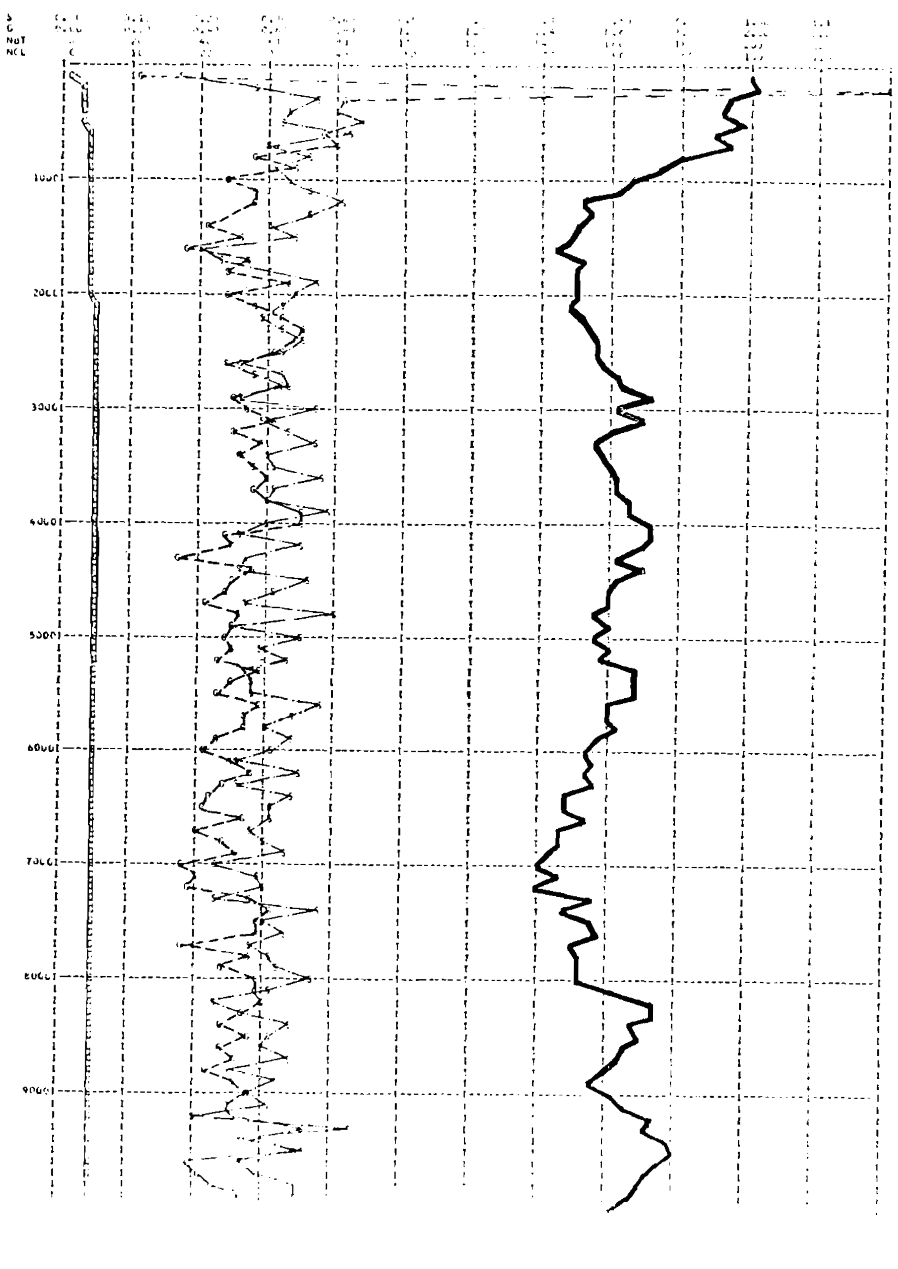


Figure 2b.

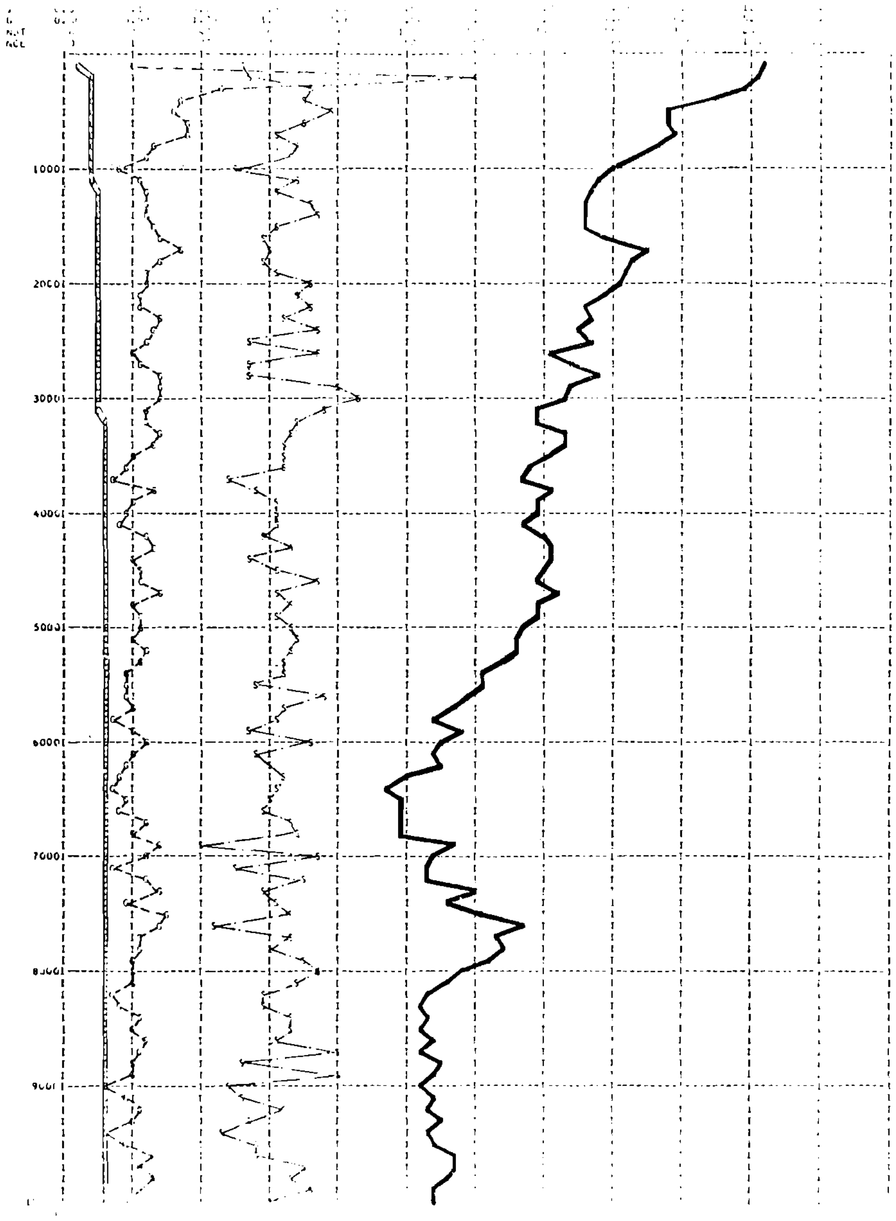


Figure 2c.

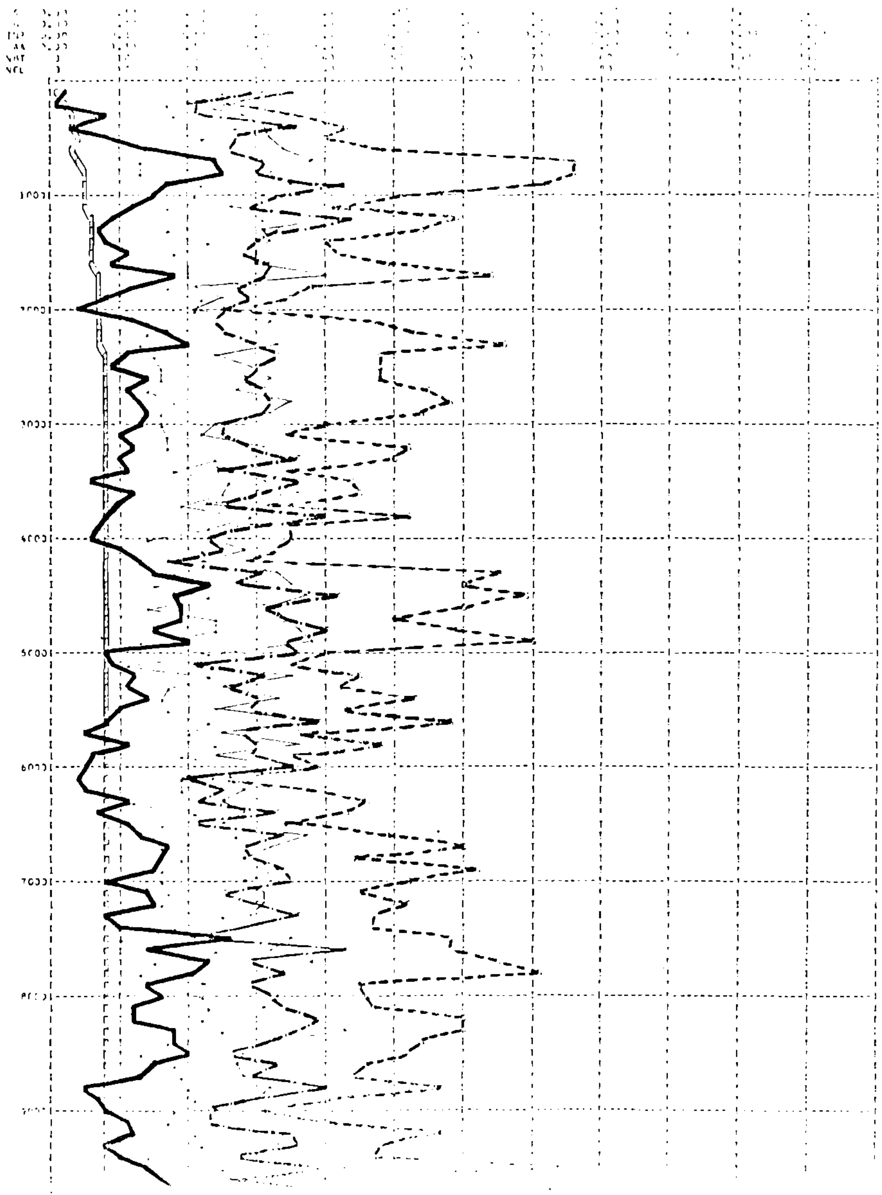


Figure 3a.

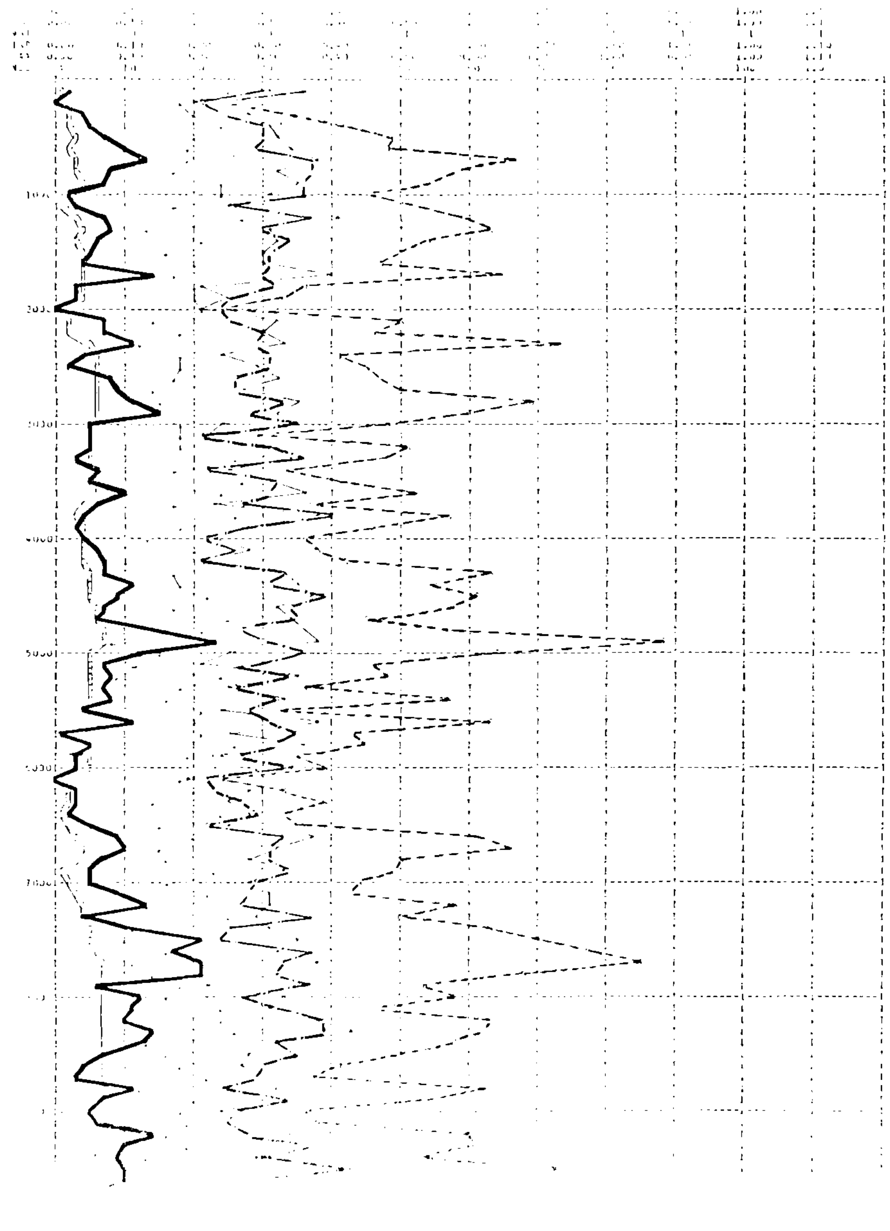


Figure 3b.

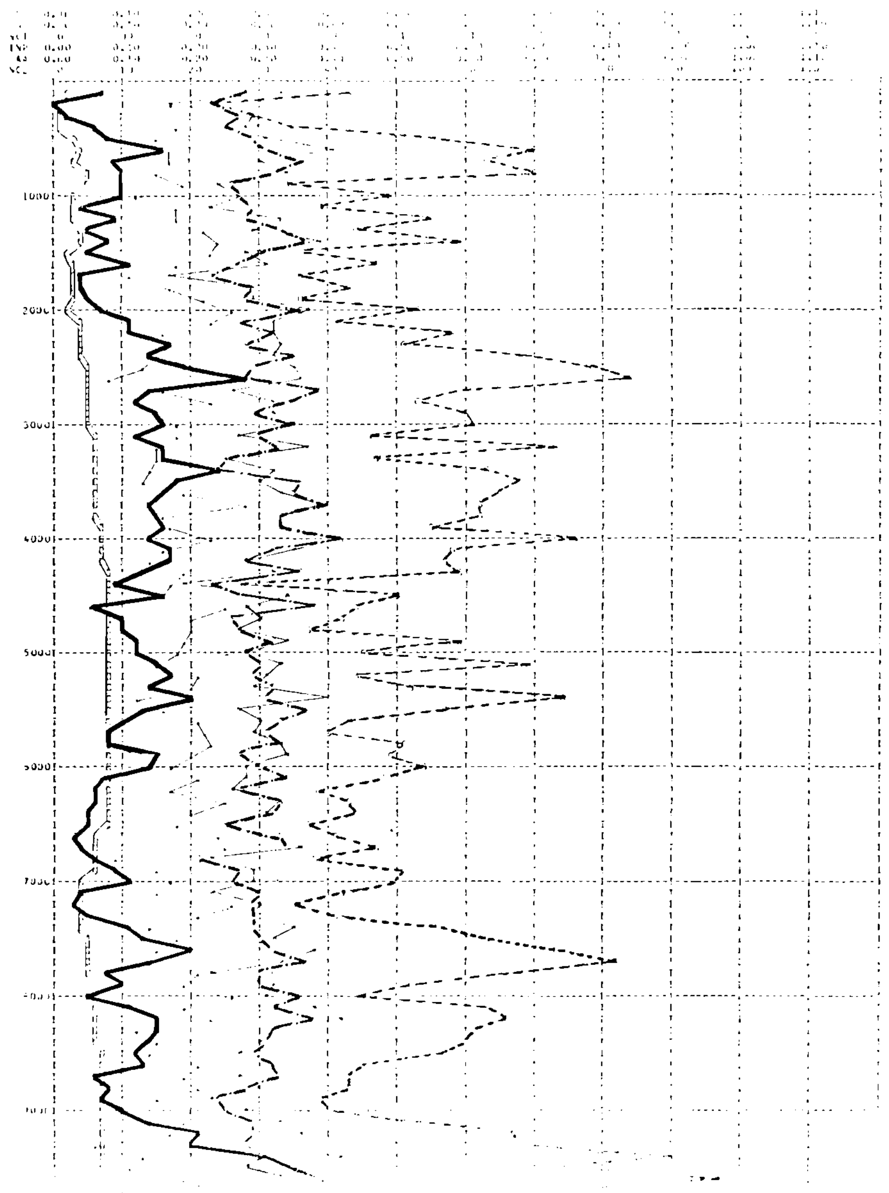


Figure 3c.

b) Although stability is achieved, the more a user has encountered unsuccess, the less is his chance to recover the active state. This probably is counter to what one might view as desirable behavior.

References

[1] Abramson, N., Packet Switching with Satellites, *AFIPS Conference Proceedings*, National Computer Conference, New York, June 4-8, 1973; 42 (1973), 695-702.

[2] Banh Tri An, and H. Demarbaix, *Politiques de contrôle pour un canal ALOHA*, Research Report 75-3, Université de Liège, Informatique. Presented at the International Seminar on Models and Measures for Computer Systems, February 25-29, 1975, Bologna, Italy.

[3] Fayolle, G., *Thèse de 3ème Cycle*, Université de Paris VI, (in preparation).

[4] Fayolle, G., et al., Stabilité et Contrôle d'un canal de transmission de données partagé, Research Report 75-2 Université de Liège, Informatique, 1975.

[5] Fayolle, G., et al., The Stability Problem of Broadcast Packet Switching Computer Networks, *Acta Informatica*, 4, 1 (1974), 49-53.

[6] Kleinrock, L., and S.S. Lam, Packet-Switching in a Slotted Satellite Channel, *AFIPS Conference Proceedings*, National Computer Conference, New York, June 4-8, 1973, 42 (1973), 703-710.

[7] Lam, S., Doctorate Thesis, Computer Science Dept., University of California at Los Angeles, 1974.

[8] Metcalfe, R.M., Steady-State Analysis of a Slotted and Controlled ALOHA System with Blocking, in *Proceedings of the Sixth Hawaii International Conference on System Sciences*, University of Hawaii, Honolulu, January 1973.

[9] Metcalfe, R.M., Doctorate Thesis, Massachusetts Institute of Technology, 1973.

[10] Roberts, L.G., *ALOHA Packet System With and Without Slots and Capture*, ASS Note 8 (NIC 11290), ARPA Network Information Center, Stanford Research Institute, Menlo Park, California, 1972.

Communication System and Access Method in RPCNET

P. Franchi and A. Fusi

1. REEL PROJECT OVERVIEW

The REEL Project was formally established in June 1974 as a cooperative effort among the IBM Scientific Center of Pisa, the Computing Center of the University of Padova and CNUCE. Other partners such as the University of Torino, CSATA (Bari) and CNEN (Bologna) joined the project later.

The objective of this cooperation is to study a networking solution for the Italian scientific community. More specifically, the purpose of the project is to provide computing centers in the education and research area with a sensible way of sharing their computational resources, such as application programs, data sets, compilers, and programing subsystems.

This objective should be attained without causing unnecessary interference with the normal activity of the centers and at the same time minimizing additional hardware and software requirements. For this reason, the basic features of RPCNET (REEL Project Computer NETwork) are: distributed control topology, dynamically variable configuration, and nonhomogeneous nodes.

Another important objective is to increase the partners' know-how and to promote a variety of computer network studies and activities.

Since the above-mentioned objectives--realization and study--are both relevant to the REEL project, the following four main activities were individuated:

a) Design and implementation of a communication system, including both the basic packet-switching subsystem (common network) and the interface to the network applications;

b) Definition and implementation of a macrolanguage at the assembler level (RNAM) which allows network users to access the communication system.

The communication system and the RNAM (REEL Network Access Method) language constitute the "architectured" part of the network. They account for distributed control and variable configuration at the design level, and nonhomogeneous nodes at the implementation level. Additional activities in the project are:

c) Definition and implementation of two user protocols--the interactive session protocol and the spool-to-spool protocol;

d) Definition and implementation of two higher-level network access methods CMS/NAM for VM/CMS users and VS/NAM for OS/VS users.

The purpose of these activities is to provide an actual basis to test the network, to evaluate its first impact on the users' community, and to promote the development of further applications.

In particular, the higher-level NAMs should allow implementation of process-to-process type of applications such as accessing remote files, distributed data base, remote transfer of control.

2. COMMUNICATION SYSTEM

The communication system is the functional unit built up by the logical and physical components of each node which allows the internode communications and provides the interface to the applications (Figure 1). The main function of the communications system is to move the user's basic information units (BIUs) from one location to another location.

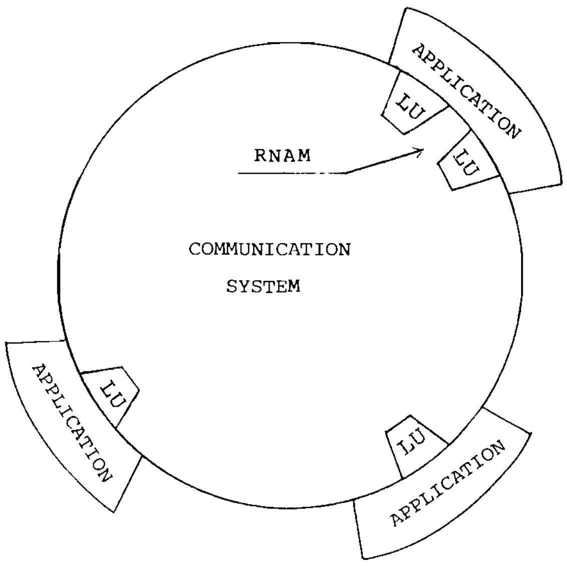


Figure 1. Applications, RNAM, and communication system.

BIUs are originated by and directed to applications. The ports, or sockets, through which the applications can access the communication system are called logical units (LU). LUs share a network-wide name space.

Each LU is allowed to exchange with other LUs "short" request/response messages any time and to establish a session with another LU. Once in session, two LUs can trade "long" messages without any specific data flow constraint.

The communication system is a distributed control machine with a variable number of control points.

An automatic reconfiguration mechanism is built into the system and is triggered by node and link activation, deactivation, and failure.

The control points of the communication system are addressable from the applications and provide them with a certain set of services. Among these services there is the possibility to inquire about the physical and logical configuration of the network and to establish sessions between pairs of LUs.

3. THE ACCESS METHOD

The access method (RNAM) provides macroinstructions that enable an application to establish a connection and exchange information with another application in the network area.

The application program must set up control blocks that contain information used to direct these operations. The general sequence of macroinstructions to establish connection and to transfer data is described in the following.

An application program must use the NACB macroinstruction to set up a network application control block (NACB), which indicates the network name of the application and may contain a password for authorization purposes.

The EXLST macroinstruction creates an exit list which contains the addresses of exit routines that are to be given control under specified conditions.

The RPL macroinstruction creates a request parameter list to be addressed by the request of an application port.

The OPENLU macroinstruction must be issued to contact the communication system and to activate the application port as an LU.

The application program is now ready to utilize the mailing and inquiring services of the network environment (MAIL and INQUIRE macroinstructions) and, moreover, is ready to become connected to another LU.

There are two methods of establishing connection. In the first, the application program takes the active role and asks directly for connection (BIND macroinstruction). In the second method, the application program is ready to accept a request for connection, putting itself in an invite state (INVITE macroinstruction).

The data transfer operations (SEND and RECEIVE macro-instructions) can begin as soon as connection has been established. The data transfer begins at that side of the application which played the active role in the connection phase.

The sender-receiver role can be changed at any time during the session on the initiative of the sender.

The receiver has the possibility to send data to the sender without breaking up the normal sender-receiver data flow (BREAK macro).

When an application program finishes communicating with its counterpart, it can break the connection by issuing the UNBIND macroinstruction.

The application program can initiate a new connection using the BIND or INVITE macroinstruction.

When an application program wants to disconnect the communication system, it must issue the CLOSELU macroinstruction.

Interactive Terminal Access in RPCNET

L. Lazzeri and L. Lenzini

1. TERMINAL ACCESS OVERVIEW

RPCNET network is a packet switching, resource sharing, distributed computer network. One of the facilities offered by RPCNET is the capability, on behalf of an interactive terminal, of accessing to a remote time-sharing system.

From the hardware point of view, the situation in RPCNET is shown in Figure 1.

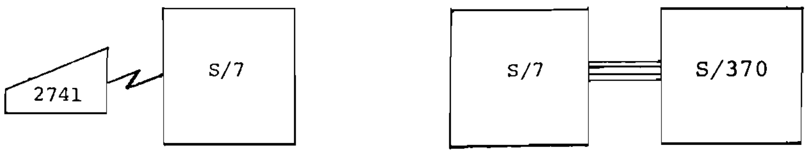


Figure 1.

The interactive terminal is usually an IBM 2741 with the break feature. The terminal is connected, locally or via TP line, to a System/7. On the other side of Figure 1, a S/370 is running a time-sharing system, and a S/7 is connected to it via a channel attachment.

By choosing the channel attachment, it is possible to emulate, quite easily, on the S/7, an IBM 70 TCU. Since this TCU is supported by the majority of the IBM operating systems, this solution does not require any software modification or addition at the S/370 side. As far as the terminal side is concerned, only an I/O support is required.

Figure 2, in terms of RPCNET architecture, gives a more general view of the problem. Within the application, it is possible to distinguish two layers:

a) The I/O support (2702 emulator or 2741 support);

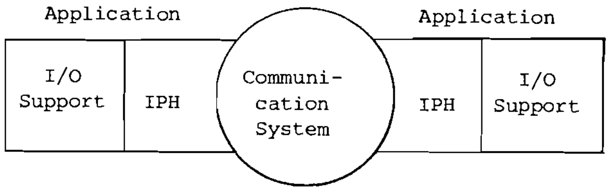


Figure 2.

b) The interactive protocol handler (IPH), the software component that interfaces the communication system and that implements the interactive protocol.

The communication system is concerned with the sending of messages from application to application. Messages are sent to and received from the CS through a specifically designed macrolanguage (RNAM).

2. INTERACTIVE TERMINAL HANDLING

At present, there are many interactive terminals available, all having different hardware characteristics; each of them having a particular set of control characters by means of which the terminal can be handled. Control characters must be used according to the physical protocol of the terminal.

It is clear that if the traffic of control characters through the communication system (CS) is very heavy, the response time will be greatly increased. Therefore, in RPCNET, the 2741 I/O support has been designed in such a way that the control characters are handled within the support itself, thus avoiding these delays. In other words, a general interactive terminal has been defined which does not require any control character and whose operation set is: PREPARE, if the break feature is supplied, READ and WRITE.

This choice implies that, also at the 2702 emulator side, the physical protocol must be managed locally. In order to achieve this, an emulator of an interactive terminal supported by the operating system (in our case just a 2741) has been added to the original 2702 emulator.

Consequently, the two I/O supports are viewed from the CS as a general interactive terminal (GIT) and as a general terminal controller (GTC).

This strategy presents other important advantages. First, every real terminal that can be mapped into our GIT can be connected to the S/370 even if it is not supported by the operating system.

Second, the definition of an end-to-end protocol for every type of terminal is not necessary; on the contrary, the definition of a single terminal protocol is made possible.

3. THE SOFTWARE ORGANIZATION

Let us now consider the problem of organizing the software implementing the protocol. It is possible to distinguish, within a protocol, the control functions and the management functions. This gives rise, from the software point of view, to two different layers: the message flow control (MFC) and the management data protocol (MDP), respectively. In the MFC, every incoming message is checked for validation of the rules established by the protocol. Nevertheless, the same checks apply to every outgoing message. If a violation occurs, the message is rejected and the sender is notified of this action.

The MDP handles data received from the I/O support and sets up messages in accordance with the rules established by the protocol. Messages are then submitted to the MFC for approval. After the MFC checks, data are extracted from the incoming messages, and the appropriate I/O command sequences are generated and submitted to either GTC or GIT for execution. The resulting situation is shown in Figure 3.

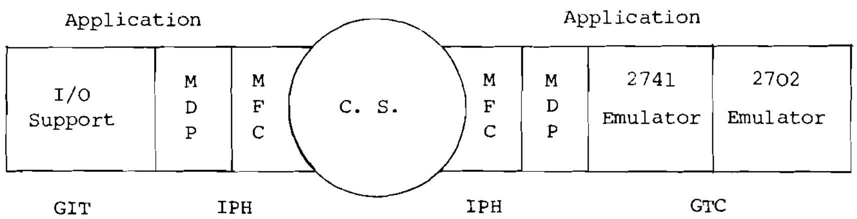


Figure 3.

Our software has been designed so that the interface between GTC and MDP is the same as that existing between MDP and GTC. Consequently, the GIT and the GTC could also be connected directly, so that the whole S/7 would behave exactly as a real 2702.

In this situation, the MDP at the 2702 side receives commands from the GTC but, instead of executing them on the terminal, it sends them in message form to the MDP at the terminal side. A complementary situation happens at the other side.

4. GENERALITIES ABOUT THE PROTOCOL

Let us now describe the protocol. The messages are constituted of two logical parts--data (not necessarily present), and protocol information. Two different types of messages are used--requests and responses, depending on the setting of the bits in the protocol information.

Data are carried through the communication system only by means of requests. Each request is associated to a sequence number which identifies the request itself.

Also, each response carries the sequence number of the request being responded to. Two different types of responses are used--positive and negative responses; the latter can carry sense data. Chains of messages are also allowed; they are handled as one logical message. Protocol information indicates whether the message is first, middle, or last in a chain. There are three possible options with regard to requesting responses: no response, exception response (a response is required only if an error occurs), definite response (a response is required in any case).

The protocol operates in a typical half-duplex, flip-flop way. The sender always controls the logical channel; control is transferred to the other side by setting the change direction indicator on the protocol information.

5. MODES OF OPERATION AND RELATED PROTOCOLS

There are several ways in which a terminal can interact with a remote time-sharing system. Three modes of operation can be distinguished: synchronous, quasi-synchronous, and asynchronous. Each of them requires a specific protocol. First, let us outline the synchronous mode.

Suppose that the GTC issues a WRITE command to the MDP; if the MDP is the sender, it will send, through the CS, a request with the following characteristics: unique in chain, definite response, no change direction; the WRITE operation will be left open to the GTC. When the message reaches the MDP at the terminal side, a WRITE operation is really started at the terminal and, at operation completed, a response is sent with the actual ending status. Upon the reception of this response, the ending status is given to the GTC and the operation is closed, thus allowing a further one to take place.

As far as the READ operation is concerned, a request with the following characteristics will be sent: unique in chain, response on exception and change direction; no data are carried to the other side. Because of the change direction indicator, the MDP at the terminal side takes control of the channel. A READ operation is really started at the terminal and data are sent by means of the request specifying unique in chain, exception response, and change direction. At the GTC side, data are given to the S/370, and the operation is closed, allowing further operations to take place.

As far as the PREPARE operation is concerned, no message is generated; however, at the terminal side, unless it is a READ or a WRITE operation, a PREPARE operation is always being executed. In these circumstances, if the attention key is hit, an asynchronous message is sent to the 2702 side, and all the activity at the terminal side is frozen until a response to the asynchronous message is received. While this is happening, other incoming messages are discarded. When the asynchronous message reaches the 2702 side, the sending operation is closed with an appropriate ending status simulating the hitting of the attention key.

In the quasi-synchronous mode, messages from the GTC to the GIT are chained together. Let us use N to indicate the maximum number of messages in a chain. When a WRITE command is received, if chaining is under way and N-1 messages have not yet been sent, a message is sent with the following characteristics: exception response, middle in chain, no change direction. The WRITE operation is then closed providing a normal ending status and thus allowing further operations to take place. When N-1 messages are already sent, the Nth message is sent with: definite response, last in chain, no change direction; the write operation is left open until the response to the chain is received.

As far as the READ operation is concerned, a message with exception response, last in chain, and change direction is sent, and the operation is left pending until data are received. For the PREPARE operation, the same considerations as made for the synchronous mode also apply to this mode.

The quasi-synchronous mode is the mode implemented in RPCNET and, at the moment, it is being tested. More detailed information about terminal protocol will be given as soon as the software has been successfully tested.

The experience obtained by testing the quasi-synchronous protocol will form a basis for the completion of the asynchronous protocol design which is now under way.

The Services Provided by Datagrams, Virtual Circuits, and Host-Host Protocol

D.W. Davies

1. INTRODUCTION

Packet-switched services are developing in a number of different ways with a consequent problem of compatibility. The most difficult compatibility questions revolve about where in the network the different levels of communication service should be provided. This paper describes three levels of service, datagram, virtual circuit, and host-host protocol and attempts to show how they are related to the communication requirements of computers and terminals.

2. TWO ASPECTS OF COMMUNICATION

Communication may be thought of in two ways--either the transportation of messages or the provision of a circuit joining the two communicating entities.

With the aid of a circuit (which is bidirectional), message transport can be organised so it might seem that a circuit is more comprehensive. On the other hand, the communication process is basically an exchange of messages (this is particularly clear for data communication between computers). The message passing from A to B may be long or short, but the data processing scheme is usually such that A does not expect a reply until its message is complete, and B cannot give a meaningful reply until it has received the whole of A's message. An exception to this simple rule is that B may need a mechanism to interrupt this flow from A if there has been some misunderstanding or error. This is, however, an exceptional situation, and usually the outcome of it is uncertain. So, the conversation must be reset afterwards to some agreed state.

It soon becomes clear that interactive or "conversational" computer processes (whether from computer to computer or between a computer and a man at a terminal) can best be described in terms of message transport. This may explain the popularity of packet-switching with some computer-system designers. At the other extreme, there are clearly some situations which are best handled by a circuit at the present state of technology.

3. THE DATAGRAM SERVICE

In its simplest form, packet-switching provides a mechanism to transport a packet from any "port" on the network to any other. The form of this packet transport service, which is being considered for international agreement at CCITT, has come to be known as the "datagram" service, and we shall therefore use this term for the packet transport facility.

A computer may wish to present a "multiple" appearance to the outside world so that within its main address (which is an address known to the data communication network) it may have a subaddress defining a port. This idea of multiple connections will also appear in connection with the virtual circuit and host-to-host protocol services later. In the datagram service, it is merely an addressing convention.

4. THE DATAGRAM CONVERSATION

The simplest form of conversation using the datagram service is for A to send B a datagram and then wait for B to reply with a datagram. The alternation rule is not strict in that A may wish to resend after a time-out if B has not replied, and the alternation may be broken by the start of a new conversation.

Practical examples of datagram conversations are electronic fund transfer, credit card validation, cash dispensers, and so forth. The precise protocol which determines who sends a message next or whether the conversation is finished will be dependent on the application. For example, a cash dispenser might send an identification and request; the central computer might reply with the authorisation; and the cash dispenser reply with confirmation that the transaction was completed. The protocol of this three-message conversation would have to provide rules for each type of abnormality or failure that might occur.

Protocols for these datagram exchanges would generally be very simple and would not have to be the subject of network-wide standards. Classes of users, e.g., banks, might find it convenient to set their own standards.

Since datagrams will provide for up to 255 octets of user information, it seems that a very large number of network applications can be organised into this form. Indeed, predictions about the quantity of "fund transfer" operations in a new data-network make it seem likely that the majority of traffic will be in datagram conversations.

Let us suppose we have a more complex situation such as a central computer polling a number of telemetry stations. Given the datagram service, this application can be forced into the mould of a datagram conversation. There might be ways to economise in packets by using the higher-level services to be described below, but the simplicity to be obtained by devising specialised protocol for a particular application would also make the datagram conversation preferable.

5. MULTIPLEXED DATAGRAM CONVERSATIONS

A single process in a computer, using a single port address, would be capable of multiplexing a number of channels by polling or carrying out simultaneous contending conversations with several other ports or terminals. The software required for such multiple conversations over one port would be complex, and an alternative would be to use one of the higher-level services, to be described below, with its associated protocol.

6. INTERFACE REQUIREMENTS FOR DATAGRAM CONVERSATION

We stated above that the protocol governing a datagram conversation would be determined by the application. The format of the datagram is a network-wide standard and, therefore, an important part of the definition of the packet interface between the network and its subscriber.

The link between the subscriber and the network must be able to transport packets reliably and be able to exert local flow control so that, for example, the network can hold off incoming traffic to avoid local congestion. There are many well-known protocols that would be suitable, for example HDLC is a likely candidate for general adoption. But this local-link protocol is a purely local matter, and it could vary from network to network and from subscriber to subscriber.

Thinking of the datagram conversation mechanism as having the components shown in Figure 1, it is clear that error control is already provided on a link-by-link basis. Flow control is broken in the communication network, but is restored, end to end, if the protocol chosen for the datagram conversation is sensible. There is generally a specialised form of end-to-end control. Within the datagram, the messages have a format which is specific to the application, and there are usually many conditions for the application program to recognise a satisfactory conversation. Thus, in a sense, there are forms of end-to-end error and flow control built into the application itself.

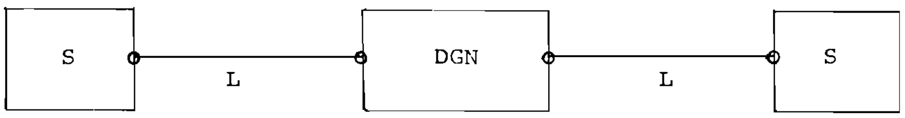


Figure 1.

Datagram conversations are not confined to single datagram messages, but problems of assembly and error control become more complex with longer messages, and it would be better to adopt one of the more elaborate services or protocols.

7. THE VIRTUAL CIRCUIT

For the transport of messages of indefinite length, a protocol is needed between two subscribers (or two ports) for error and flow control and possibly for assembling the message at the destination. During the operation of this protocol, the two subscribers (or ports) are closely associated and might be said to be logically linked together. One type of protocol of this kind provides a service which is known as a "virtual circuit". This can be understood best if the service is provided using the datagram service as its underlying mechanism.

Like a real circuit, a virtual circuit provides for a stream of bits to be sent from one port to the other without error and without losing their original sequence. In general, a circuit is bidirectional, and our use of the words "virtual circuit" will assume this.

The bit streams will be carried as packets, and one of the simplest ways to organise this is to leave the datagram network unchanged and to organise the virtual call protocol within the subscriber's computers. This situation is shown in Figure 2. The basic task of the virtual call protocol is to keep the packets in the right sequence. For this purpose, it must add sequence numbers at the source and check them at the destination. Since errors might leave the sequence indeterminate, the virtual call protocol must provide error control by retransmission to ensure that a packet is reliably received before it is delivered in the sequenced stream. The virtual circuit protocol should also provide flow control so that if the destination slows down or ceases to accept packets, the source will not waste its efforts sending ones that cannot be received.

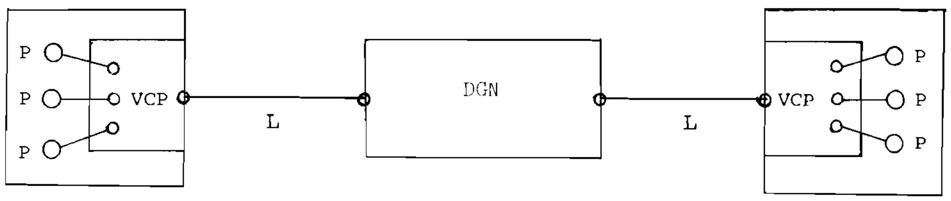


Figure 2.

In the scheme described so far, the packets retain their identity even though it is the string of bits (or octets) within the packets which passes over the "virtual circuit". This feature is thought to be generally useful since what passes over the circuit are messages having a special format which is meaningful to the application, and it should be possible to use the boundaries of network packets (datagrams) to help define this format. For example, the first few bits of each packet might encode its function--such as data/control and first/interior/last packet of a message.

This is not quite essential, however, and, according to another scheme, the packets sent into the virtual circuit may be split up and rejoined en route, retaining only the same bit (or octet) sequence but losing the original division between packets. However, we think that this restructuring of packets in the virtual circuit is not helpful to most applications. It is perhaps too literal an interpretation of the idea of a virtual circuit and brings the system unnecessarily close to properties of a real circuit which are not necessarily the best for the computer communication situation.

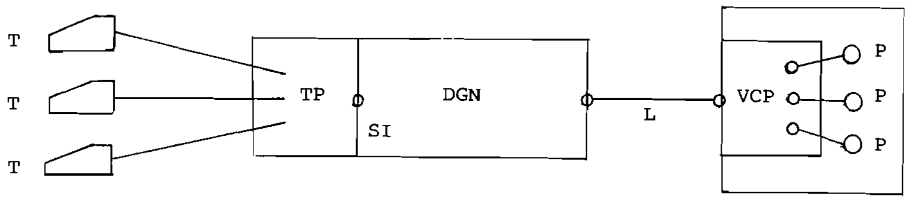
In most respects, the virtual circuit is unlike a real one. For example, the amount of data stored in the network is variable, and the data rates at source and destination are not instantaneously locked together.

8. THE VIRTUAL CIRCUIT AND THE CHARACTER TERMINAL INTERFACE

A character terminal such as a visual display unit receives and transmits a stream of octets, and, during a conversation, it would normally exchange these with only one port in a distant computer. (A possible exception is that the setting up of the call might involve a different port.) Since the sequence of the bytes is important, the character terminal requires a virtual circuit service and not a datagram service.

In many packet-switched networks, character terminals are handled by a special service called the "terminal processor" or "packet assembler/disassembler".

Therefore, the terminal processor administers one end of a virtual call protocol, but, because the data rate for these character terminals is not high, it might be a simplified version of the virtual call protocol defined for the network as a whole. Figures 3 and 4 illustrate this situation.



DGN	datagram network	P	port or process
L	local link	SI	software interface
S	subscriber	TP	terminal processor
VCP	virtual call protocol	T	character terminal

Figure 3.

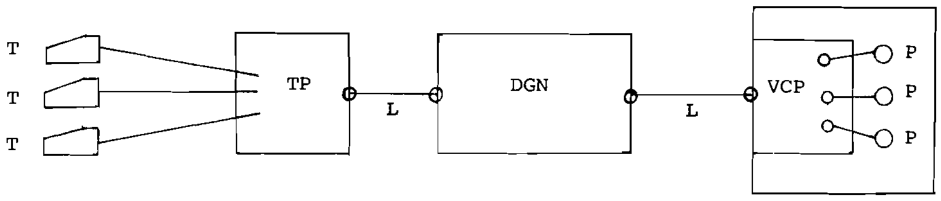


Figure 4.

9. MULTIPLEXED VIRTUAL CALLS

The existence of a single channel virtual call should not be ignored, but, in the majority of cases, computers on the network will be capable of multiple access. The virtual call protocol which they implement, though it uses a single local link to the datagram network, will provide virtual calls to many ports at a time. One reentrant program can administer several ports by using a set of status indicators, counters, and buffers for each port. This situation is already illustrated in Figure 2. In a similar way, the terminal processor normally handles a number of terminals as shown in Figures 3 and 4.

10. VIRTUAL CALL FACILITIES WITHIN THE NETWORK

We have described a virtual call protocol as if it were implemented entirely outside the data communication network because this is the simplest model. In the case of the terminal processor, it is a reasonable objective for the data communication network to provide a character interface as well as a packet interface, and it may be combined in the network either as a separate processor or combined in one machine with a software interface as shown in Figures 3 and 4, respectively.

A reasonable objective is to make the terminal processor compatible with the virtual call protocol so that any one of the terminals (see Figures 3 and 4) has a bidirectional virtual circuit to one of the processes (or ports) in the subscriber's computer.

A second scheme for incorporating virtual calls in the network is to require all the network's services to take this form. For a packet interface this is shown in Figure 5. Both the network and the subscriber are involved in virtual call protocol since L is, effectively, a link at the datagram level.

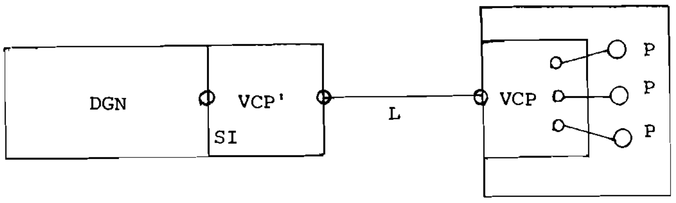


Figure 5.

This apparently untidy arrangement is advocated because it imposes flow control on each virtual circuit, whereas the flow control in Figures 1 and 2 is carried out by the subscribers. The problem, seen particularly by telecommunication authorities, is that a wrongly functioning protocol in the subscriber's equipment will fill the datagram network with packets which cannot be delivered.

However, there are proposals to protect the network against these types of failures, and similar failure mechanisms can be proposed for the virtual circuit facility of Figure 5.

11. HOST-TO-HOST PROTOCOL

The virtual call provides two useful attributes of a successful conversation. It links, conceptually, the two ends of the call throughout the period for which the call is maintained, and it keeps the datagrams in correct sequence. In doing this, the virtual call protocol must also provide end-to-end error and flow control.

The virtual circuit facility does not undertake the reassembly of messages. Although it allows long messages to pass through, it does not recognise them as messages.

As we stated previously, computer communication is organised in messages and generally very short ones, but a facility for transporting long messages will, on occasion, be required. An end-to-end protocol that organises long messages goes by a number of different names such as "host-to-host protocol" in the ARPA network, and "transport station" in the CYCLADES network.

In the transport of long messages, the flow control between the two ends has to be tied in with the computer's operating system. Finite amounts of store are available for message reassembly, and there may have to be a pause in transmission while more storage is found. This is relatively easy by administering the host-to-host protocol in the subscriber's computer. Very long messages will not be held in simple input/output buffers but will be transferred in pieces to and from other stores, making use of the variable data rate and dynamic multiplexing which characterises packet-switched networks.

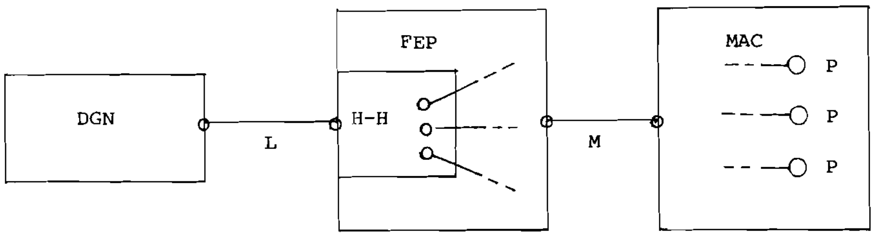
Transport of long messages introduces the problem that a virtual circuit may be occupied with one message for a long time. Short urgent messages may be delayed. One solution is to provide these short urgent messages with an alternative virtual circuit operating as a signalling channel. To save the overhead of setting up two circuits, we can provide for urgent messages to break into the stream of datagrams that carry the main flow. This is the function of the "telegram" of the CYCLADES transport station.

The host-to-host protocol is normally multiplexed so that it can handle a number of virtual circuits at a time. Figure 2 could, with H-H replacing VCP, illustrate the host-to-host protocol in action between two multiaccess computers.

12. LOCATION OF THE HOST-TO-HOST PROTOCOL

The host-to-host protocol is a complex thing having of the order of 10,000 statements in its program. It has sometimes been suggested that it should be contained within the network or carried out in a front-end computer which is linked to the main multiaccess machine. Such a situation is shown in Figure 6. It is immediately obvious that the link marked M in this figure is very complex. It would probably be based on a local link protocol like L, but, on top of this, it must have its own protocol capable of communicating between the collection of ports served by the host-to-host program and the corresponding processes in the multiaccess computer. Furthermore, all the flow control, error control, and signalling considerations for each port must be extended to the multiaccess computer. The problems of how much reassembly of messages is carried out and which levels of storage are employed in the computer are all matters of agreement between the front-end processor and the multiaccess computer.

The arrangement of Figure 6 is unnecessarily complex, but it might be justified in one situation. This arises where the complex link M has already been provided in the software of the multiaccess computer for some other purpose. Sometimes, in order to avoid writing special software for the multiaccess computer, the front-end processor is made to look like one of its standard peripherals such as a disc store, and a complex mapping is made between the facilities of the host-to-host protocol and those of the preexisting link M with the protocols that implies. This arrangement may be inconvenient or restrictive since M was not designed for the purpose. Its only virtue is that it leaves the main computer's software unchanged.



VCP'	modified VCP	FEP	front-end processor
H-H	host-host protocol	MAC	multiaccess computer
		M	link with additional protocol

Figure 6.

We see this as a transitory situation until there is agreement on the host-to-host protocol, and the main manufacturers have come to terms with it sufficiently so that it is generally provided within their software. The investment involved is small compared with present-day telecommunication software packages.

13. RELATIONSHIP OF HOST-TO-HOST AND VIRTUAL CALL PROTOCOLS

One merit of the virtual call protocol is the attention given to it by the telecommunication authorities who are discussing it as a partner to the datagram service. However, it stops short of the full range of long message transport facilities which the host-to-host protocol provides.

In the same way that the datagram service is simpler than the virtual call when applied to a packet-switched network, there are some respects in which the host-to-host protocol is a simpler one to implement than the virtual circuit. This arises because the messages handled by the host-to-host protocol, though they may be large, can nevertheless be limited in size. The size limit implies that messages which are still longer must be split up by the application program. However, the problem of maintaining long messages in sequence and dealing with their flow control adds very little to the normal work of the application program. For example, a magnetic tape is customarily divided into blocks, and these should travel as messages since they have their own flow and error control requirements. It would be pointless to regard them as a single long message for data communication purposes.

14. CONCLUSION

A very rough summary of the situation is as follows. Experience has shown the utility of the terminal processor and the host-to-host protocols. The datagram service can also serve a very large number of network applications.

The virtual circuit occupies an intermediate and uncertain position. As it is normally presented, it is more powerful than the terminal processor requires though the terminal processor could be a subset of it. It is less specific to the long message transport requirement than the host-to-host protocol. On the other hand, the host-to-host protocol cannot, in the general case, be incorporated in the network itself but should be in the subscriber's computer.

Perhaps an ideal situation, though it may be unattainable, would be to have the virtual circuit protocol incorporated as a lower level of the host-to-host protocol. This may or may not be efficient. It would, however, make a properly layered structure in which the terminal protocol was just one "setting" or restriction of the more general virtual call protocol, and this, in its turn, would be part of the host-host protocol.

We repeat that it is by no means clear that this kind of structure is efficient; it is suggested simply because its structure seems right.

The Uses of the ARPA Network via the University College London Node

Peter T. Kirstein, and Sylvia B. Kenney

1. INTRODUCTION

The main theme of the present workshop is the "Interconnection of Computer Networks". It is our thesis that key questions in the interconnection of such networks are those that relate to the nature of the use that will be made of the networks--how will they be used, by whom, and to what extent and purpose? The responses to these questions should influence vitally the technical aspects of the design of the individual networks, and also of the methods used to interconnect them. To provide some guide to answering these questions, we describe in this paper some of the uses made of an existing computer network, ARPANET, which has a node at the University College London (UCL). In fact, a number of networks are interconnected via the UCL node, but the expected usage has influenced strongly what facilities we have provided at our network gateways. This paper, therefore, discusses what use has been made of the UCL node.

To explain the usage, we discuss in section two the UCL configuration, and in section three the conditions of usage. These clearly have a great effect on what use is made. To corroborate users' own reports of their activities, we have introduced monitoring arrangements, which we mention briefly in section four. The actual usage made is discussed in section five. The requirements for successful usage and our conclusions are presented in sections six and seven.

2. THE UCL NODE

At UCL, there is the standard ARPANET terminal handling communication computer [13], which can handle both character terminals and packet-mode computers. The terminals and the computers can be either local or remote. The UCL configuration is sketched in Figure 1. Here PDP-9A is used as a front-end to several other service computers, in particular the Rutherford Laboratory (RL) IBM 360/195, and the Computer Aided Design Centre (CADC) Atlas computers. Users of both centres can get out from their sites to ARPANET through PDP-9A, so that all traffic to and from the network and the service sites passes through this machine. We note on the PDP-9A system log the identifier of every user passing in either direction and how long he is attached. We are also able to note which host he is accessing, but do not yet do so.

The speeds of the lines between the service computers and the PDP-9A were dictated by the facilities convenient for those machines. In addition to the service computers, a number of leased and switched telephone lines are attached to the terminal ports. The lines attached as of 1 July 1975 are indicated in Figure 1.

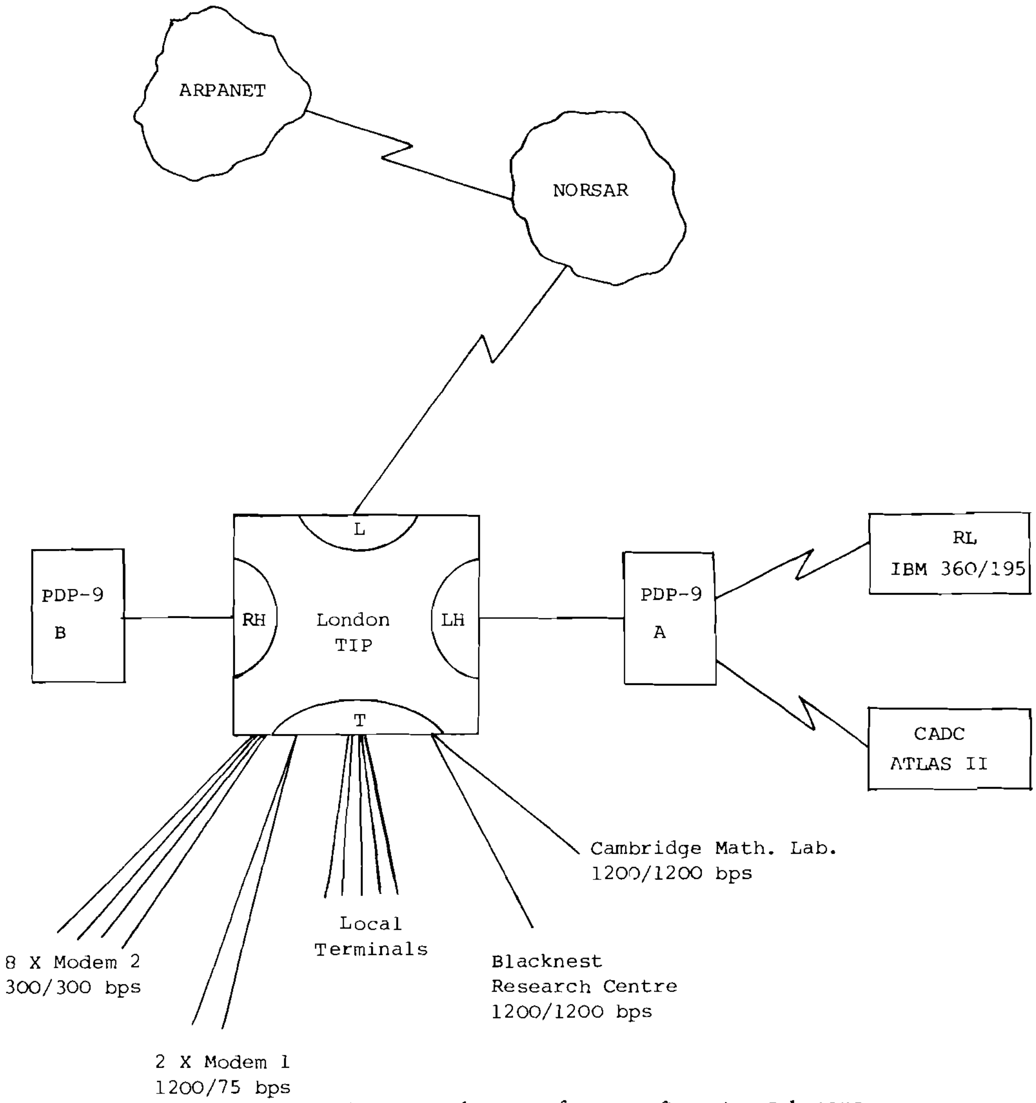


Figure 1. Schematic of UCL configuration, July 1975.

The speeds of the leased terminal lines were indicated by the availability of terminal facilities and the fact that input of higher rates than 300 bps cannot usually be sustained by the TIP in series with one slow line (7.2K bps via satellite) on the path to the US. The 300 bps ports can support all the usual speeds and ranges of terminals operating at up to that capacity in duplex mode. The number of such ports reflects the large demand for those facilities. The 1200/75 bps ports are provided for the attachment of VDU terminals; this is the highest speed available in the UK for duplex terminals on the Public Switched Telephone Network.

3. CONDITIONS ON THE USE OF HOSTS AND THEIR EFFECTS

Largely because of PTT policies, there can be no direct payment internationally for the use of computer time, and all use must be noncommercial. For this reason, all the use of the network via the UCL node must be cooperative in the sense that an organization in one country must meet any direct costs incurred by users in the other. As a result, almost all the use is confined to situations where research groups in each country are working in the same field, so that the foreign activity can be justified fairly easily as part of the same project. To a small extent, there are guest accounts at some sites; however, the facilities available with such guest accounts are so limited that the amount of serious work that can be done in this way may be discounted. Nevertheless, even this limited access has a considerable educational value. Most sites have a large number of documentation files which may be examined on-line using a guest account. These provide a valuable insight into the interests and research activities of that particular site. The more helpful sites will also provide the names of people to contact for further information.

There is a formal control mechanism through the governing committee of the project. Any UK group wishing to make use of facilities through the UCL node must apply for permission to use the network. Any US group must obtain the permission of ARPA. Thus, the number of groups authorized to use the link is strictly controlled. During the initial period of the connection, any noncommercial cooperative research project from a UK university or government organization could be considered for approval. Moreover, no direct charge was made to any UK university research groups, either to use host facilities in the US, or to pay a proportion of the common communications facilities costs. Those organizations with leased lines to UCL did have to pay (except RL which is in a different category).

4. MONITORING ARRANGEMENTS

We have had to examine the problem of measuring the traffic going through the UCL TIP. In general, it is the function of the communications subnetwork to keep records for billing purposes of the traffic between sites and specific user groups. However, the initial plans for ARPANET to keep such records have been shelved. The accounting information generated in the host sites is of variable quality. It is often of limited value for assessing specifically foreign usage because of the cooperative nature of many of the projects. People on both sides of the Atlantic use the same accounts, so that the use cannot be separated out. For this reason, the only place to generate any statistics on network usage to and from the UK is at UCL.

The duration of use of host computers through PDP-9A of Figure 1 is noted on the latter machine as mentioned in section two. PDP-9B is used for substantial periods each day to contact continually all the dialled-in ports, and to request incoming users to identify themselves, and to declare the host they want to access. This information is also kept in machine-readable form.

From this automatic collection of the duration of network usage by each group, it is possible to build up a matrix of usage. This information helps calibrate the subjective and objective information given by the different groups themselves about the extent of their network usage.

5. NETWORK USAGE VIA THE UCL NODE

It is interesting to determine what usage is made in practice of facilities such as those available over the ARPA network.

5.1 Data-base Usage

Probably the largest single usage is that of information retrieval from data-bases. Two UK projects in particular come into this category--those of the British Library and those of the Blacknest Research Centre.

As part of the British Library Research and Development Short-term Experimental Project, several user centres are accessing the MEDLARS data-base via the National Library of Medicine"s MEDLINE system on their IBM 370/155. Smaller, more specialised data-bases, such as TOXLINE, are also accessed. Each center collects various data on its usage of the system via ARPANET, and these, as part of the experiment, are correlated with usage of other data-bases. We at UCL are also measuring various characteristics of this usage to allow extrapolation to other similar systems.

The primary interest of the Blacknet Research Establishment in using ARPANET is the exchange of seismic data. Users at the Blacknest Research Establishment access the data-bases generated by the various seismic arrays in Norway and the US, operated in conjunction with the US Seismic Data Analysis Center, on a computer at the University of Southern California. US users access a seismic data-base provided by Blacknest Research Center on the RL IBM 360/195. This data-base consists of the last 30 days seismic data (specifically, P wave onset times, amplitude, and period), for four seismic reporting stations in Scotland, England, India, and Swaziland. Two additional stations in Canada and Australia will be added soon. In this way, the seismic data may be accessed much faster than was possible previously. The hardcopy listings of seismic events are still prepared, but from the on-line data.

5.2 Intergroup Communication

In fact, this is by far the most useful and widely used facility--partly in its own right and partly as part of other projects. There are excellent message and teleconferencing facilities available on several of the host systems [2,7], and no cooperative activities could really flourish without these. Care must be taken in their use; one of the conditions of international usage insisted on by the PTTs is that messages can be sent only in connection with computations in progress. However, almost all the groups using the network rely heavily on the message facilities for their communication with the changing systems and their cooperating partners.

It should be emphasised that these facilities are quite different from simple telex. Besides allowing several participants to access files at the same time, they allow document preparation and review. In addition to allowing coordination of activities, they allow comments and queries about the use of programs to be answered quickly and naturally, and it allows the input and output of programs to be passed easily between people at sites widely dispersed.

Several of the projects have been based on the teleconferencing facilities. Examples are the seven teleconferences listed below (with the UK participants in parentheses):

a) The interaction of telecommunications and travel (UCL) [9];

b) The design of a questionnaire for certain field tests (UCL);

c) The effect of sulphur pollutants on people (St. Bartholomew's and St. Thomas's Hospitals) [8];

d) Political simulation (Southampton U);

e) The design and implementation of internetwork protocols (UCL, National Physical Laboratory) [10];

f) The design requirements for, and review of, message processing facilities (UCL) [6];

g) The design of satellite communication experiment (UCL).

Each of these conferences extended over several months and involved participants at more than four geographically separated sites. The fact that it was possible to hold such conferences over long periods, while in possession of all the relevant reference material, was very important. It was also vital that the participants did not have to be at terminals simultaneously, though if they were, almost interactive communication could be achieved between the participants. In this way, a whole group of people could take part in a conference without the usual disruption of their normal working schedule caused by having to attend a meeting at a particular place and time.

These examples illustrate some of the requirements for message traffic. In each case, the teleconferences were about matters which involved extensive computation, and the participants exercised each other's computer programs and discussed their data-bases.

It was vital to be able to switch easily between powerful text information retrieval facilities, running of computer programs, and plain text input.

5.3 Programming Packages

Many of the applications involved groups developing computer programs jointly or allowing mutual access to already existing programs. In one instance, similar systems were developed on either side of the Atlantic and then compared in terms of efficiency and power [4]. In other cases, valuable data were supplied by a group working in one country, to be run over the network with a program developed by a group at a remote foreign site, to the mutual benefit of both sides. This has overcome the problem that existed previously, of having to take a program from the machine on which it was developed, and adapting it to run on a completely different system.

Examples of these activities are development and use of packages in the following areas:

a) Location of seismic disturbances from seismographic data (Blacknest Research Establishment);

b) Algebraic manipulation (Cambridge Univ.) [1];

c) Computer aided building design (Royal College of Art) [5];

d) Shared file systems (National Physical Laboratory, Queen Mary College) [3];

e) Advanced graphics systems (Computer-Aided Design Centre, Cambridge);

f) High-energy physics experiments (Oxford Univ., Rutherford Laboratory).

5.4 Use of Unique Resources

Many of the examples given above already fall into this category. Here, a "unique resource" may be a type of computer; it also may be a unique set of software. For example, there are more than twenty PDP-10s on the ARPA network, but only one, at Bolt, Beranek and Newman (BBN), has a particularly advanced BCPL compiler. A few additional ones are given below, again indicating the UK groups involved:

a) Use of Illiac IV (Culham Laboratories [12], Reading Univ., Salford Univ. [14]);

b) Artificial Intelligence packages (Institute of Neurology, Sussex Univ., Essex Univ., Edinburgh Univ. [11]);

c) Large data-bases (London Univ.).

6. REQUIREMENTS FOR SUCCESSFUL USAGE VIA UCL NODE

From the large variety of usage made of the UCL node in its first eighteen months of operation, we realise some of the requirements for successful results:

a) It is essential that a good documentation and advisory service supports the existence of the node. Advice must be given about minor operational problems such as how to access systems, how to use the various network facilities, and what documentation exists. It is particularly important to keep users supplied with up-to-date information on changes to systems and facilities. Accounts must be organised and introductory courses given. A mechanism for requesting and disseminating documentation must be set up. Specific documentation not available elsewhere must be prepared.

b) All parts of the system must work reasonably reliably at the times of interest to users. Thus, for example, one system in California is available only from 5:00 a.m. local time; this unavailability in the mornings (UK local time) seriously affects the usefulness of the system to us. This type of regular unavailability is much less serious than an irregular one. For on-line usage, it is very important that a communication system and any service hosts have a high availability, and that terminals be readily available to the participants.

c) Although in some cases cooperative activities have been possible without personal acquaintance, in most cases the network cooperation is supplemented by personal contact, at least occasionally. In the majority of instances, the cooperating parties knew each other's work before network communication started, and usually knew each other personally. Very often a member of a UK group has come from a host site in the US or has spent some time there. There is still no effective method of finding out on ARPANET who is interested in specific subject areas and of enabling the parties to become aware of their mutual interest. In addition to network communication, the use of conventional communication means, such as exchange of large documents, drawings, and telephone conversations, are still desirable for cooperation.

7. CONCLUSIONS

A significant body of cooperative work has been possible in the first eighteen months of operation of the UCL node of ARPANET. This usage has been in widely different fields, most of which was not foreseen at the start of the project. The principal uses have been for information retrieval, communication between research groups, and shared development and use of common programming packages. There has been some use of unique resources, but this has been of lesser importance; this is partly because of the way cooperative use was encouraged and partly because of the range of computer facilities available locally to the participants.

In our experience, the subjective responses of the groups on the extent and success of their usage has not been too reliable. Automatic monitoring has provided good objective comparisons with the subjective responses.

There is still a huge scope for determining the effects of cost factors on network usage. In our project, the charge to individual users has been divorced deliberately from the costs of the whole project, the effect of charging and access control on network usage is a complex and important question.

Acknowledgments:

Whom to acknowledge in a paper of this kind is difficult. Clearly, this work would have been impossible without the support of the project sponsors: the British Department of Industry, the British Library, the British Post Office, the British Ministry of Defence, the British Science Research Council, and the US Advanced Research Projects Agency.

The work which has gone into the build-up of the UCL facilities has involved many people. A.R. Duncan and H.R. Gamble constructed the necessary interface hardware and have kept the system operational; P.L. Higginson developed most of the software used in PDP-9A as the UCL front-end to the RL machine; D.L. Bates, R. Gould, P. Mott, and A.V. Stokes were responsible

for the monitoring software; SR Wilbur has organised much of the documentation and information dissemination mechanism. Finally, the many user groups from outside UCL have contributed their experiences.

References

[1]* Barton, D., and J. Fitch, Applications of Algebraic Manipulation Programs in Physics, in *Reports on Progress in Physics*, 35, 3 (1972).

[2] Black, E., *The DMS Message Composer*, MIT Dynamic Modelling Center, 1974.

[3]* Coulouris, G., and M. Cole, *On Distributed Computing and Reactive Systems*, Computer Science Lab. Report, Queen Mary College, Nov. 1974.

[4] Fitch, J., A Solution of Problem # 3 Using CAMAL, *SIGSAM Bulletin*, 32 (Nov. 1974).

[5] Johnson, C., *An Abstract Experimental Text of IMAGE*, Dept. of Design Research Working Paper, Royal College of Art, July 1975.

[6] Kirstein, P.T., and S.R. Wilbur, The Impact of Integrated Message Processing Facilities on Administrative Procedures and Inter-Personal Interactions, *Proc. European Computing Conf. on Communications Networks*, Sept. 1975, 395-414.

[7] Lipinski, H.M., and R.H. Miller, FORUM: A Computer Assisted Communications Medium, in *Proc. 2nd Int. Conf. on Computer Comm.*, 1974, 143-147.

[8] The Human Responses to Sulfur Pollutants, Institute for the Future, Oct. 1974.

[9] *Proc. of 1st International Computer-based Conference on Travel/Communication Relationships*, Bell Canada, July 1974.

[10] Lloyd, D., and P.T. Kirstein, Alternative Approaches to the Interconnection of Computer Networks, *Proc. European Computing Conf. on Communications Networks*, Sept. 1975, 499-518.

[11]* Milner, R., and R. Weyrauch, Proving Compiler Correctness in a Mechanised Logic, *Machine Intelligence*, 7, Edinburgh University Press, Edinburgh, 1972.

*These references, while relating to the network projects of the individuals concerned, do not describe their network activities specifically.

[12]* Petravic, M., et al., Automatic Optimization of Symbolic Algol Programs, *Journal of Computational Physics*, 10, 3 (1972).

[13] Rettberg, R.D., *Specifications for the Interconnection of Terminals and the Terminal IMP*, Report No. 2277, Bolt, Beranek and Newman Inc., 1972.

[14]* Walkden, F., et al., Shock Capturing Numerical Methods for Calculating Supersonic Flows, *A.I.A.A. Journal*, 12, 5 (1974).

*see footnote on previous page.

Connecting Data-Networks: Standardization of Routing

G. Megman

1. EURONET

EURONET, network for retrieval and exchange of scientific and technological information, is being created by authority of the EEC Council of Ministers. The totality of users interested in this information will, via the network, obtain rapid access to the totality of information resources. Simultaneously, the network will offer vast opportunities for dialogue among users and for cooperation among information resources (e.g., distributed data bases).

In principle, the geographical area to be covered is not restricted to EEC countries. The project must, at any stage, take advantage of existing infrastructure and install complementary infrastructure where necessary. In the initial stages, this philosophy implies interconnection with other networks in the area, if feasible. Consequently, questions of internetwork compatibility must be investigated.

In basic design, EURONET could be conceived conventionally as a network of node and interface processors, linked by full duplex leased lines; the node processors constituting a packet-switching network; the interface processors representing the sources and sinks of this network. In principle, all node processors could be identical, but interface processors may develop in a variety, each kind handling certain types of terminals or hosts or servicing sluices to other networks.

The outstanding feature of packet-switching networks is the efficiency in the use of communication lines that can be achieved. In other words, such a network may be designed with little over-capacity on the lines. Addition of another traffic generator will increase the total traffic and also modify the traffic pattern, enhancing traffic on some lines more than on others. Saturation of some of the lines and, hence, partial or total breakdown of the network may occur.

The network operator must safeguard the existing level of performance of the network. Therefore, he should estimate in advance the impact of an additional traffic generator on network performance. In practice, this can be done only for well-known categories of traffic generators. By monitoring the traffic in the network, the operator can gradually improve his knowledge of the properties of the participant traffic generators, and, if a similar generator is to be connected, he can reasonably well predict its impact on the network.

If an unknown generator is to be connected, a different technique must be used. Initially, it can be introduced to only a few of the other generators. Its impact on the network will then stay small, and the operator can monitor this impact and then extrapolate to the case when the new generator would be fully introduced. When in doubt, he may choose to introduce the generator subsequently to different subsets or larger subsets of traffic generators.

These techniques would also seem adequate for the interconnection of networks. Each network represents to the other network an additional traffic generator. Then a priori knowledge about the power of these generators may be considerable because each operator can inform the other about the traffic-generating properties of his network (see example below). So, each operator can start redesigning his network in order to accommodate the additional load expected. Probably several alternative designs must be prepared to evaluate, e.g., the advantages and disadvantages of multiple connections between the networks. The operators then must reach agreement on a preferred single design. All in all, the design phase could be rather prolonged. Therefore, it is sensible to start simultaneously the procedure of gradual connection, implementing initially only a small capacity link and allowing only small subsets of the traffic generators in each network to be introduced mutually. This approach offers the advantage that unexpected problems are discovered early and can be corrected in the design.

2. EXAMPLE OF A CALCULATION ON INTERNETWORK TRAFFIC

Suppose network A offers three services, and its average client traffic generator using these services has output/input:

$$(c_1,d_1) \quad (c_2,d_2) \quad (c_3,d_3) \quad \text{bps.}$$

Suppose network B offers two services, and its average client traffic generator using these services has output/input:

$$(c_4,d_4) \quad (c_5,d_5) \quad \text{bps.}$$

Suppose network B has n_1,n_2,n_3 clients for the services of A, and network A has n_4,n_5 clients for the services of B. Then the expected traffic between the networks is:

from B to A: $n_1c_1 + n_2c_2 + n_3c_3 + n_4d_4 + n_5d_5$;

from A to B: $n_1d_1+n_2d_2+n_3d_3+n_4c_4+n_5c_5$.

If the activity of a generator in B is in average a factor f higher than in A, the expected traffic between the networks is:

from B to A: $f(n_1c_1 + n_2c_2 + n_3c_3) + (n_4d_4 + n_5d_5)/f$;

from A to B: $f(n_1d_1 + n_2d_2 + n_3d_3) + (n_4c_4 + n_5c_5)/f$.

Of course, the actual calculations may be more refined if the knowledge about each network is more detailed or if restrictions apply (e.g., limited capacity of a service), and also if one of the networks is already connected to a third network.

3. ROUTING

It becomes desirable to formulate principles that can govern the traffic in and between data networks, particularly in situations of congestion.

Such an inventory of routing options, formulated in sufficient detail and adequately published, would stimulate coordinated research and lead to a better understanding of the properties of networks. Selected routing options may then gradually emerge as recognized standards. In this way, a frame of reference is created that will facilitate the construction and interconnection of networks. Schemes for routing should be simple and at the same time effective.

Our aim is to reduce traffic to congested areas in the network, without requiring the individual node computer to "know" more than just the local traffic situation. To this end, a proper level of housekeeping is required in the nodes. For example, a node keeps track of its own level of congestion, defined as the relative occupancy of its buffer area for packets. If the level of congestion rises, the actions available to the node are: decreasing input and increasing output.

a) Output can be increased by directing internal traffic to output lines that may otherwise not be fully utilized.

b) Input can be decreased in two sensible ways: by asking neighbor nodes to reduce their output; by asking traffic generators to reduce their output (i.e., their input to the network).

Such requests cause extra traffic. So if not exercised with great care, this medicine may make the illness worse. The requests should preferably be directed only to one neighbor or to one traffic generator, selected according to well-defined criteria.

If we now elaborate these three actions in more detail, we shall assume that hierarchical addressing applies, which enables each node to map the large group of packet addresses into a smaller group of address clusters, which again is mapped into the group of output lines (neighbor nodes) by means of a routing table (RT). The RT indicates how suitable the output lines are for each address cluster. The sum of these "priorities" will be constant, i.e., not dependent on the address cluster.

The RT's are presumably prepared by the network operator, but in an intelligent way, so that historical traffic patterns in the network are routed optimally. The concept of an RT may be simple or more sophisticated, e.g., for each address cluster just one output line may be indicated, or priorities may be given for all output lines.

3.1 Utilization of Output Lines

Full utilization of the output lines can now be achieved by adding the "age" of the packet to the priorities as obtained from the RT. The additional advantage is that no packet is ever "forgotten", e.g., by waiting in a closed output queue. The age of the packets can be defined by a counter at buffer entry. The design of buffer and queue managers is obviously important, but the subject is well known and can be dealt with separately.

3.2 Requesting Distant Traffic Generators to Reduce Output

The origin of packets in the buffer is distributed over the address clusters. Excessive deviations from a defined normal distribution can be identified. Subsequently, the packets from the excessive clusters can be scrutinized to discover an eventual excessive traffic generator.

Probably an excessive traffic generator is troublesome for several nodes. So, to aviod an avalanche of requests that could become a new source of congestion, the packets carrying these requests should be recognizable. Then a node can discard such a request if it had lately passed on or issued the same request. In this connection, the timing should, of course, be related to packet transit time. Further investigations could aim at a better understanding of the cost/benefit balance for this type of request.

3.3 Requesting Neighbors to Reduce Output

Suppose input to a node over a particular line is excessive. The node then asks the relevant neighbor to reduce speed on that line and also specifies a recovery time, which would be a characteristic time possibly adjusted for congestion.

The neighbor would immediately decrement the speed of the relevant output line and would increment this speed again after the specified delay, unless a similar request is received during the delay. Speed reduction is affected by idling after each transmitted packet. Idle time may equal the product of packet length (in seconds) and the desired speed reduction, expressed as a fraction.

The need to reduce input depends on the level of congestion (x) and its rate of increase (r). For example the condition may be:

$$r < (x_0 - x)/\tau \quad .$$

Also, the frequency of testing such a condition should depend on x and r. For example, test again after an interval $(x_0 - x)/er$.

Once it is established that input to the node must be reduced, the crucial question is, "which input line should be reduced?" To this end, the node may consult, e.g., a topology table, which relates input to output lines. It then becomes possible to calculate "desired" input speeds, based on actual output speeds. Subsequently, the differences between actual and desired input speeds are determined, and the line with greatest excess should be reduced.

4. RESEARCH TOPICS

Packet-switching networks may be expected to support congestion waves. Even with "stationary" traffic input from the generators, these congestion waves may excite spontaneously, in particular when adaptive routing is utilized.

The amplitude of these waves would, of course, be the major concern, but also frequency, wavelength, and propagation speed are of interest. Is the damping positive or negative, and what are the effects of a collision between two waves? Does the network have resonance frequencies, and where are the resonant points located? How do the parameters of the network influence these properties, and, in particular, what routing algorithm can achieve favorable properties?

Basic tools to forge answers to these questions may possibly be borrowed from other disciplines. Transport theories exist in economy and in the social sciences. Close similarities seem to be exhibited by transport phenomena in solid state physics in the presence of thermal phonons.

It appears that this area of research is wide and largely unexplored. Our understanding of node and network behavior could be greatly enhanced by the development of analog and digital simulators, designed to clarify specific issues.

Tests on Data Communication Equipment Using Minicomputers

F. Oismüller

1. INTRODUCTION

This presentation is concerned with work on the field of data communications as done by the Department for Applied Electronics of the Bundesversuchs- und -Forschungsanstalt, Vienna.

This work consists of:

a) studies related to security questions in data communications;

b) interface problems and compatibility questions;

c) writing software for data communications and for test setups for data transmission equipment.

According to this division, the first part of this paper deals with aspects of data structures, noise structures, and redundancy check facilities and their use in transmission security measurements. In this connection, some compatibility problems appear, which are dealt with in section three; section four describes some program modules developed in the Department for Applied Electronics.

2. DATA STRUCTURES, NOISE STRUCTURES

When performing comparative measurements on security of data communication equipment, generally acknowledged standards are needed. Well known are the CCITT test patterns as defined in V52 and V56, which, respectively, are decided for measurements of distortion and error rates at Baud rates under 4800 bit/s and over 4800 bit/s up to 96000 bit/sec. The principles of generation are the same in both cases:

a) pseudostochastic structure;

b) patterns must contain long sequences of logic zeros and ones;

c) pattern must be much longer than the period of typical noise bursts;

d) simple method of generation.

The generation of CCITT test patterns by software raises some complications that are discussed in section four of this paper.

The introduction of redundancy check facilities in each case is increasing the expense of transmission time or bandwidth. The theory of these methods is well known, but detailed experimental investigations of the dependence of error rates on noise structures, signal structures, and redundancy are still missing.

As experience shows, white noise (although not in each case) is a useful means in performing error-rate measurements on data communication systems. In most of the cases, disturbances occur as bursts, i.e., large noise amplitudes can be observed during short time periods.

In the Department for Applied Electronics, a transient recorder is under construction, which will be able to analyze transients with a conversion period of 20 ns with a solution of 8 bits. The use of this instrument in telephone or other networks will bring deeper understanding of the structures of noise. In measurements, the application of the "stochastic burst generator" turned out to be very useful.

3. COMPATIBILITY QUESTIONS

The test patterns, as mentioned previously, are raising the first compatibility questions. Both 511 bits (V52) and 1 048 575 bits (V57) cannot be divided into bytes, although today byte organization in computers is dominating.

It is not useful to store long test patterns (e.g., 1 048 575) in a core memory of a minicomputer. Instead, it is proposed that one search for combined hardware-software solutions. In this case, the ease of interfacing is of major importance.

We suggest, when introducing standards for redundancy check facilities, that one consider the generally acknowledged principles of computers internal organization. Redundancy check options, as available from computer manufacturers, sometimes work on a software and sometimes on a hardware base. Very often the manufacturers use these options to provide intentional incompatibility to the products of other companies. Examples for redundancy check facilities are VRC, LRC, and CRC, the latter using test polynoms (CRC-16, CRC-12).

Another kind of problem is involved with synchronization. Data transmission interfaces of computers usually contain sync character codes as hardware functions. It is desirable that standardized test data include sync characters in a sufficient number.

4. TEST PROGRAMS

The program modules, as described in this section, are able to execute the following tasks:

a) generation of test patterns;

b) storing of test patterns in the core memory (if necessary);

c) transfer of test pattern to the data transmission equipment;

d) receiving of the transmitted, distorted, and disturbed signal;

e) comparison of received and transmitted signal;

f) counting error rates;

g) printing and plotting of results;

h) statistical evaluation of results;

i) control of parameters of measurement.

Data exchange between the processor and the data communication interface is accomplished via data break facility to or from any location of the core memory.

The instructions used in test software include initializing the data transmission equipment, receiving go and transmitting go, skipping to start subroutines; among other instructions are those for diagnosis and/or debugging.

When starting the program, the test engineer inputs duration of test (as a number of seconds or a number of test patterns), the parameters of the transmission line (e.g., attenuation, delay, noise level,etc.), Baud rate, and finally the starting instructions.

Upon receipt of two consecutive synchronizing characters, memory references start, and the comparing routine detects transmission faults and sets the bit-and-block error counters. Both erroneous bits and 8 bit, 64 bit, and 511 bit blocks are counted.

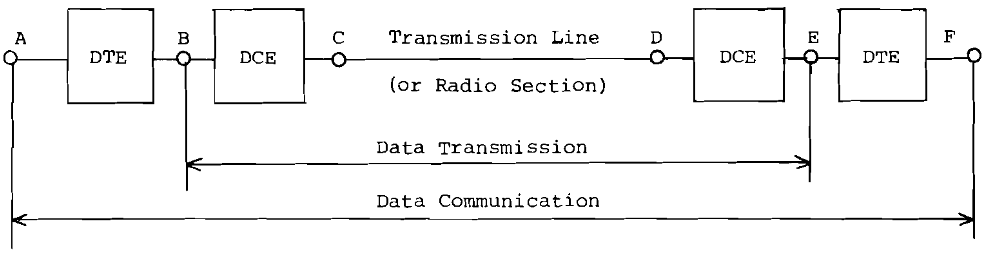
The program modules concerning computations, printing, and plotting are written in the programing language FOCAL. The test programs themselves are written in assembler, and they are called by a specially defined function. The elements of text, the format instructions, and the commands for the different evaluation procedures are very simple FOCAL routines and can be loaded by the paper-tape reader. The plotting routine itself computes the plot using regression methods. FOCAL resides in 8 k of core memory, and the test routines require 4 k; thus, the total requirement of core memory is 12 k.

Standards and Measurements on Data Transmission Equipments

A. Sethy

1. DATA COMMUNICATION AND DATA TRANSMISSION

IIASA is planning an Inter-European data network. In the course of the installation of an international data network, many telecommunication (i.e., transmission) problems are relevant. In the transmission techniques, the international standardization is highly developed; therefore, these aspects are also of great importance for data communication. For a clear understanding of all questions, the relation of the terms *data communication* and *data transmission* must be well clarified. The usual terms and definitions in different languages are not the same and, therefore, are not clear. The following figure shows schematically a data communication system.



DTE = Data Terminal Equipment

DCE = Data Circuit-Terminating Equipment (MODEM)

Figure 1. Scheme of a data communication system.

The section in Figure 1 between the interface points B and E including the transmission line and the DCEs (MODEMs = MOdulator and DEModulator equipment) is called "data transmission". The task of the MODEMs is the preparation of the data for the transmission; therefore, this function is a pure telecommunication procedure. The DTEs are mostly parts of calculators; they perform the adjusting, the protection, etc., of data. Therefore, these functions are mainly data processing, and the sum of both these functions is also mainly data processing.

The Department for Data Transmission of the Institute for Electronics (ETVA) of the Bundesversuchs- und Forschungsanstalt in Vienna (BVFA) is occupied especially with measuring problems of data transmission and with testing of MODEMs. In this area, it collaborates very closely with competent international organizations, as well as with the International Telecommunication Union (ITU).

2. INTERNATIONAL STANDARD FOR DATA TRANSMISSION

According to Article 1 of the agreements between the Organization of the United Nations and the ITU, it was accepted that ITU is the responsible organization for promoting the development of technical facilities for worldwide telecommunication services, including data transmission, and also for laying down of standards in this field. The suborganization CCITT (International Telegraph and Telephone Consultative Committee) of the ITU is responsible for data transmission. In the standardization work, ITU/CCITT are obliged to collaborate with other relevant organizations, such as the International Standard Organization (ISO) and the International Electrotechnical Commission (IEC).

The division of competences according to the agreements between these organizations is as follows:

a) CCITT is responsible for the standards for the transmission *channels* including all aspects of the networks;

b) The standardization of *MODEMs* is the province of the CCITT;

c) The standardization of *interface* between DCE (MODEM) and DTE is a matter for agreement between CCITT, ISO, and IEC;

d) The devices for *error protection* are, until now, a matter for the users;

e) The Telegraph *"Alphabet No. 5"* is a product of an agreement between CCITT and ISO (CCITT/V.3, ISO/646-1942)

f) General *coding* questions are the province of CCITT, if the public network is used;

g) The limits for transmission *performance* fall within the competence of CCITT;

h) The limits for *performance of the DTE* should be fixed by agreement between the ISO and CCITT;

i) Alone, CCITT can lay down procedures for the setting-up, holding, and clearing *calls for data communication* when the general public network is used.

Since it can be assumed that the public network will be used for the planned IIASA Inter-European data network, the CCITT recommendations are of great importance to this work.

3. CCITT RECOMMENDATIONS FOR DATA TRANSMISSION MEASUREMENTS

According to the above-mentioned rules, CCITT publishes recommendations every four years for data transmission. The last edition has been the so-called *Green Book* from 1973, volume VIII [2]. This contains two series for this subject: The "Series V", general for data transmission over telephone or telex networks; and the "Series X" concerning data networks.

The recommendation "V" describes the principles, signal structures, transfer rates, modulation equipments (MODEMs), error protection, and measurement procedures for data transmission. Until now, about 30 recommendations have been accepted.

The Department Data Transmission of the ETVA/BVFA is working with the measurement problems, and the author of this paper is chairman of a working group at CCITT for this special problem.

The V-recommendations concerning measuring techniques in data transmission are as follows:

V.50, standard limits for transmission quality of data transmission;

V.51, maintenance of international circuits for data transmission;

V.52, characteristics of distortion and error-rate measuring apparatus;

V.53, limits for the maintenance;

V.55, impulsive noise measuring instrument for data transmission,
V.56, comparative tests of MODEMs;

V.57, comprehensive data test set for high data signaling rates.

4. COMPARATIVE TESTS ON MODEMs: WORK OF THE ETVA

The recommendation V.56, "comparative tests of MODEMs for use over telephone-type circuits," is a product of the experiences in the MODEM testing work of the ETVA. A great number of products of the most important companies of the world have been tested in ETVA at the transmission speed of between 600 and 4800 bit/sec. The measuring criteria are bit-and-block error rate, telegraph distortion, etc., by different simulated transmission parameters and noises, as described in V.56. Many theoretical and practical problems have been investigated during this work (see [1,3,6]). At present, we are developing new testing equipment for MODEMs for the transmission rates between 48 and 96 kbit/sec according to recommendation V.57.

For the practical comparison and selection of MODEMs, internal limits exist for measuring results, which are known by the manufacturers. If a large organization needs a great number of MODEMs for its planned data network (i.e., ORE, which is the international railways organization), it orders tests on the present types on the market, and the most suitable types will be chosen based on our measuring work. It could be possible that international standard and measurement aspects and experiences will be useful for the future work of IIASA.

References

[1] Eckl, K., Prüf- und Messmethoden in der Datenübertragungstechnik, *Eletrotechnik und Maschinenbau*, 91 (1974), 32-35.

[2] *Green Book*, ITU, Geneva, 1973.

[3] Marko, H., and D. Heidner, Ein Messplatz zur Prüfung von Datenübertragungssystemen, *Nachrichtentechn. Z.*, 22 (1969), 78-84.

[4] Sethy, A., Fragen zur Messung der Qualität von Datenübertragungssystemen, *Nachrichtentechn. Z.*, 22 (1969), 85-87.

[5] Sethy, A., Messergebnisse bei der Prüfung von Datenübertragungsgeraten mittlerer Geschwindigkeit, *Nachrichtentechn. Z.*, 23 (1970), 542-547.

[6] Sethy, A., Blockfehlerhäufigkeitsmessung bei Modems mit Hilfe eines Störspannungsgenerators, *Nachrichtentechn. Z.*, 27 (1974), 383-386.

An Operational Method for Achieving Dynamic Sharing of Files in a Distributed Interactive Computing Utility

N.H. Shelness, and J.K. Yarwood

1. INTRODUCTION

In early 1975, the computing power of the Edinburgh University Interactive Computing Utility was doubled by the installation of a second large ICL System 4/75 computer. The decision to install a second free-standing machine rather than to upgrade the one we already had was forced by certain structural features of the ICL System 4 range. The System 4/75 is a single-processor machine. This makes enhancement of the system by the addition of a second processor impossible. There is also no larger machine in the 4 range on which to base an upgraded system.

The same structural features make it impossible to create a single fully integrated system, despite the fact that certain peripherals (drum and disk drives) are accessible from both machines, and other peripherals (tape drives and slow devices) may be switched from one machine to the other. The impossibility of creating a truly integrated system forced us, in the first instance, to run two separate free-standing versions of the Edinburgh Multi-Access System (EMAS), with each serving a separate group of users and being accessed from a separate group of interactive terminals. This was not a satisfactory state of affairs. There are some natural breaks in our user population, but these breaks are not such as to distinguish two totally disjoining groups of users. If it were impossible to achieve full hardware integration of the two systems, an approach would be needed that would give the external appearance of integration.

Two areas dominate the external appearance in a system such as EMAS. The first is the means by which the system is accessed; the second is the ability to dynamically share and control access to files. Changes were needed to achieve the appearance of integration in these two areas. The first was achieved by the creation of a freestanding terminal network serving both systems, the second by a scheme for the sharing files between the two systems. The latter is the subject of this paper.

2. FILE OPERATIONS

It is possible to distinguish two types of operations on file access operations and organizational operations. Access operations are those that take place on the contents of a file (read, write, and search operations). Organizational operations fall into two groups--those that operate over a file system as a whole, such as space allocation and deallocation (file creation and deletion operations), and those that control access to a file or group of files (file open and close operations). The division between access and organizational operations is not precise and varies considerably from system to system. In a data-base system, almost all access operations are performed via organizational operations. In a simple tape-based system, organizational operations may not even exist, or they may be limited to simple label processing. It is also the case that there is a certain degree of recursion in the relationship of the two types of operations; as organizational operations are, out of necessity, constructed out of access operations.

If a file system is to be shared by more than one system, then not only must access operations be possible from each system, but also organizational operations.

3. FILE OPERATIONS IN EMAS

The Edinburgh Multi-Access System supports a four-level storage hierarchy consisting of core, drum, disk, and tape store. Files that have not been accessed for four weeks are held on tape. All other files are held on permanently mounted disks.

All user computations, and system tasks that require files (the slow peripheral spool process, and the archive (tape) store manager for example) are performed in disjoint virtual machines, of which the system can support 63 simultaneously. Each virtual machine has a 16 mega-byte linear virtual address space organized into 256 segments of 16 4096-byte pages each. All access to files is performed through the virtual memory via a mechanism that associates segments of disk storage with segments of the virtual address space. The system is then responsible for moving pages from the disk store to core store as they are addressed through the virtual memory mechanism. In order to maintain the page traffic between disk and core within that which the disk channel can sustain, drums are used by the system to hold the most commonly accessed pages of the moment. The organization of both the core and drum stores is managed for each virtual machine by the EMAS resident supervisor. The organization of the disk store is managed from within each virtual machine by a supervisory system process (the director). A shared disk segment of director code is mapped into each virtual memory whenever a virtual machine is created along with an unshared segment that holds the tables associating disk segments with virtual addresses, the director stack, linkage area, and off-stack region. In addition, 14 segments of the virtual-address space are available for accessing file directory segments which are mapped into the virtual memory as required by each director process. These 16 director segments are at the top of the virtual memory, and cannot be accessed by the user process which has only a 240-segment virtual address space available to it.

All file organizational operations are performed on behalf of a user process by its director. The available operations are listed in appendix A. The file store is organized into disjoint file systems each associated with a single disk pack. The directory segments for each file system are at a fixed disk address which is known to the director processes. Directories are organized as a two-level hierarchy indexed by user identifier and file name. The position of a user directory within a file system is held in the root directory of each file system. A separate directory segment holds the free-space tables for each file system.

Synchronization of access to each user directory and the free-space tables is controlled via a semaphore maintained for each directory within a file system by the resident supervisor. This allows directories to be accessed in parallel with contention occurring only when two directors require access to the same directory simultaneously.

4. SHARED FILE OPERATIONS

Two sets of proposals were put forward for creating an apparently integrated EMAS file store. The first called for the creation of a free-standing file store controlled by a dedicated file processor. The second for the creation of a nominated director on each EMAS system that would act as an agent for file organizational operations originating from the other system. The second proposal was finally accepted and implemented, largely because it created little disturbance to the existing systems, and because it required a minimum of implementation effort. Despite the fact that we have only implemented the second proposal, we shall discuss each for the light they shed on the general problem.

5. FREE-STANDING FILE STORE

The idea of separating the permanent file store from the direct control of a central system has been advanced in Edinburgh for a number of years. It is our belief, strengthened by the use of similar concepts in the design of the CDC STAR 100 computing system, that a computing system should be functionally distributed along the dimension of its memory hierarchy, with central processor executing programs, a paging processor managing main memory, a drum processor managing the drums, etc. Within this scheme, the file system might well be managed by two separate processors--a file-index processor and a file-access processor.

In the context of EMAS, we chose to start with a single system performing both sets of operations. Access operations would be initiated, as at present, by the virtual memory management component of the resident supervisor. Organizational operations would be forwarded by director processes, rather than performed by them as is done at present. This change would be invisible from the user process, as the user file system interface would remain unchanged.

Two different interfaces would be constructed between the free-standing file store and a central EMAS system--a page transfer interface and a control interface. The page transfer interface would be a byte transfer DMA interface under the direct control of the file processor. Information would be transferred with minimal buffering between the core store of the central system and the file medium. This would avoid the need for a large temporary buffer area in the file processor.

The control interface would be a duplex channel with access and organizational requests passing in one direction and replies in the other. The unit of transfer would be a 32-byte block in EMAS system's message format, identical to that currently employed as the standard EMAS interprocess communications mechanism.

Alternative interfaces would be available for connection to other processors, such as a slow peripheral spool processor, if one were added.

The mechanization of file access and organizational operations in the file processor have not been worked out in detail, though certain philosophical decisions have been taken. Access scheduling would depend on the characteristics of the physical device attached. Directory operations would be single threaded, with file descriptors for open files held in the main store of the file processor, along with the free-space tables. All directories would be held on the same physical device as the files they described. An event numbering scheme would be employed to achieve consistency dynamically in the case of a central system failure, avoiding a hiatus each time a central system failed.

A short description of the sequence of events involved in both file organizational and access operations originating on EMAS and being serviced by a free-standing file processor appears in Appendix B.

6. AGENT PROCESSES

The second scheme for achieving file-system integration is deceptively simple, and requires almost no implementation effort. This was, in no small part, due to the elegence of the existing EMAS design. Rather than servicing the requests of its associated user process to connect and disconnect files, the agent director services requests to open and close files forwarded by directors running on the other system, as a result of requests made on them to connect or disconnect a file. These requests are passed through the EMAS system's message mechanism which has been extended to allow communication between processes over both systems. The agent does not need to service file creation and deletion operations, as these can only be performed by a user on his own files, and a user is still required to run on the system to which his files belong.

Access operations are performed directly by each system on all disks. This introduces the chance of one system interfering with the disk operations of the other. Initial analysis indicates that, due to the effectiveness of the drum-buffering scheme in minimizing the disk transfer rate, and the relative infrequency of cross-system accesses, this should not be a problem. If it were to become a problem, or if the two machines were incapable of cross-accessing each others disks, as would be the case if they were geographically distributed, then a file access agent would be required on each system to handle file-access requests originating from another.

A short description of the sequence of events involved in both file organizational and access operations originating on one EMAS system and being serviced by agent processes on the other appears in Appendix C.

7. CONCLUSION

We have described a working scheme for achieving dynamic file sharing between two disjointed, loosely coupled, operational-computing systems. We have described the structure of agent processes performing organizational file operations originating on another machine, and we have proposed a scheme for operating file-access agents to service file-access requests originating on another machine, in cases where direct access to the file media is impossible. Together, these agents provide a means of sharing files between two remote machines in a single computing utility, should the bandwidth of the intermachine links be sufficiently wide. We are, at present, quantifying this bandwidth requirement under operational conditions.

References

Arden, B.W., et al., Program and Addressing Structure in a Time-Sharing Environment, *JACM* 13, 5 (1966).

Corbato, F.J., and V.A. Vyssotsky, Introduction and Overview of the MULTICS System, *AP IPS Conference Proceedings*, 27 (1965) 185-196.

Denning, P.J., The Working Set Model for Program Behaviour, Comm. *ACM*, 11, 5 (1968), 323-333.

Millard, G.E., and H. Whitfield, The Standard EMAS Subsystem, *Ccmputer Journal*, 18, 3 (1975).

Rees, D.J., The EMAS Director, *Computer Journal*, 18, 2 (1975) 122-130.

Shelness, N., et al., The Edinburgh Multi-Access System Scheduling and Allocation Procedures in the Resident Supervisor, *Proceedings of the International Symposium on Operation System, Theory and Practice*, IRIA, Paris, April 1974.

Stephens, P.D., The IMP Language and Compiler, *Computer Journal*, 17, 3 (1974) 216-223.

Whitfield, H., and A.S. Wight, The Edinburgh Multi-Access System, *Computer Journal*, 16, 4 (1973) 331-346.

Wight, A.S., The EMAS Archiving Program, *Computer Journal*, 18, 1 (1975) 131-134.

APPENDIX A

User Program File Requests (by SVC)

CREATE

DESTROY

RENAME

Specify a "new generation" of a file

CHANGE SIZE (file may be connected)

Set archiving/back-up status

Set access permissions

CONNECT

DISCONNECT

Change access mode

Get file information

Get file names

Offer file to another user

Accept file from another user

APPENDIX B

A Step by Step Description of the Handling of a File Organizational Operation and a File Access Operation Originating on EMAS and Being Serviced by a Free-Standing File Processor

FILE ORGANIZATION OPERATION

1. A user issues a file connection request by sending a system's message addressed to the user connection process. The system routes this message to the user's director.

2. The director readdresses the message to the system file connection process, and identifies itself as the sender. The system routes the message to the file processor, control interface handler.

3. The handler transfers the message to the file processor.

4. The message is routed within the file processor to the file index process.

5. The file index process requests the transfer of the nominated user directory from the file medium.

6. The user file directory has been transferred. The file index process verifies the right of the requesting user to access the requested file in the requested mode. If access is not allowed, an error reply is constructed and sent. If access is allowed, a file descriptor is constructed in the file access table, and an index to the descriptor is returned in the reply which is constructed and sent.

7. The reply is routed in the file processor to the control interface handler, which transfers it to the appropriate central system.

8. The handler in the central system receives the transfer and sends the message to the addressed director.

9. The director receives the reply. If an error is indicated, this information is passed back to the user process with the appropriate flag set. If the request has been satisfied, the director enters the returned file, descriptor index in the table associating file segments with segments of virtual memory. If the file is more than one segment in length, a subindex is inserted for each segment after the first. A successful reply is then sent to the user process.

FILE ACCESS OPERATION

1. A page fault interrupt causes a system's message to be generated and sent to the virtual store manager. If the virtual store manager wishes to service the page fault and the page is in the file store, it acquires a page of main store and constructs a system's message from information held in the virtual memory association table. The system will route this message to the file processor, control interface handler which transfers it to the file processor, where it is routed to the file access process.

2. Using information held in the indexed file descriptor, a file page transfer is scheduled.

3. When the transfer is initiated, information is transferred, via a tight loop, through the file processor, page transfer interface from the file medium to the store of the requesting processor. If the transfer is successful, an appropriate reply message is constructed and sent. If an error has occurred, then a retry is scheduled. Should recovery be impossible, the file is marked as inaccessible, and an error reply is constructed and sent.

4. The reply makes its way back to the virtual store manager in the same way as the file connection reply above. If the transfer were successful, the appropriate tables are updated and the process rescheduled. If the transfer failed, a first-level contingency is generated and sent to the process on whose behalf the transfer was attempted.

APPENDIX C

A Step by Step Description of the Handling of File Organizational Operations and File Access Operations Originating on one EMAS System and Being Serviced by Agent Processes on the Other

1. A user issues a file connection request by sending a system's message addressed to the user connection process. The system routes the message to the user's director.

2. If the request is for a file held on that system, the director processes it. If the directory is not held on that system, a message is constructed and forwarded to a nominated process in the system on which the directory is held.

3. The intermachine message handler transfers the message to the addressed system where it is routed to the nominated process.

4. The nominated director services the request. If access is not permitted to the file, an error reply is generated. If the access is allowed, a reply is generated and a 256 byte, large system, message block is acquired. The address of the disk segments holding the file is entered on the big block. The big block is then also addressed and sent via the big block interface.

5. Both the reply and the big block message are routed to the director which forwarded the request.

6. In both the case of the file connection request that was serviced immediately and the one that was serviced remotely, the local director is responsible for setting up the table, associating disk segments with virtual address segments. In the first case, the identity of the segments is copied from the directory. In the second case, the address is copied from the big block.

7. An appropriate reply is sent to the user process indicating success or failure.

Simulation of Resource Allocation Strategies for Interconnected Computer Networks[1]

P. Haas

1. INTRODUCTION

With the increasing number of installed or planned computer networks, the question arises as to how they should be connected with each other. This problem has to be investigated not only from the technical but also from the operational viewpoint. This paper deals only with the latter aspect and suggests, as solution, the usage of a newly developed simulation model, which, on the one side, is a sufficiently precise model of the operational situation in interconnected networks, and, on the other side, can still be used as a practical tool by the average user of such a network.

2. OPERATIONAL MODELING OF INTERCONNECTED NETWORKS

The development of the model is based on the following assumptions:

There exist two separate computer networks, called N_1 and N_2. Both N_1 and N_2 comprise sets of resources called $\{R_1\}^2$ and $\{R_2\}^1$, respectively. Topology and operational characteristics of both networks are known. Each net has sets of users called $\{U_1\}$ and $\{U_2\}$; each user sends jobs into his network. The job streams can be described by means of the statistical distributions of interarrival times and execution times (also called job length).

The individual members of the sets $\{U_1\}$ and $\{U_2\}$ assign a certain utility coefficient T to their networks, which is, of course, a subjective measure and hardly expressible as a figure. Generally, one can only say that the utility coefficient T is a function of direct cost (e.g., data transmission cost), and system quality.

[1]This paper describes a part of a comprehensive work, which was submitted under the title "Investigations Concerning the User's Behaviour to Achieve Optimal Load Scheduling in Computer Networks" as inaugural dissertation at the University of Linz (Austria).

System quality can be expressed in quantitative terms called system characteristics (e.g., turn-around time) and, in nonquantitative terms, called system properties (e.g., the advantage of a net being able to cope with peak load).

Though it is very difficult to arrive at a precise specification of the utility coefficient, it definitely should be used as a measure in all decisions concerning the operations of a network.

Let us assume now that for some reason it is contemplated to interconnect N_1 and N_2. Now, the users have to operate in a changed environment and will, most likely, have to assign a new utility coefficient T, different from the old T, to the new combined computer network N_1 plus N_2. Each user will then want to know how he can achieve the greatest possible T'. This means that he needs a new operational procedure and, most likely, a new resource allocation strategy. Since the above-mentioned system characteristics depend largely on the workload scheduling, a model was developed which allows calculations of system characteristics as functions of the chosen resource allocation method.

The users of this model must make certain assumptions about the resource allocation strategy and are then able to calculate the respective system characteristics (e.g., turn-around time); these values, in turn, help them in determining their utility coefficient. By means of an appropriate set of such model runs, it is possible to arrive at a certain optimal resource allocation strategy, where the word *optimal* must be understood in context with a defined operational environment and in coexistence with competing other users, who also want to operate with their maximum possible utility coefficient.

3. DESCRIPTION OF THE STRATEGY MODEL

In order to combine both nets N_1 and N_2 in one model, each of the two networks are transformed into identical models, as shown in Figure 1; the two models are then interconnected at a certain point and thus form the strategy model. It is beyond the scope of this paper to explain why this structure of the model has been chosen; all results obtained in the simulation runs, however, have confirmed the validity of this model setup.

The individual components of this model have the following functions:

a) JS_1 (resp. JS_2):

Job streams as input to N_1 (res. N_2). Represented either by empirically measured (determined) job streams or by statistical values such as average arrival rates and average service rates. For most investigations, one chooses a POISSON-distribution for the arrival rates and an exponential distribution for the service rates.

b) $WL_{1,1}$ (res. $WL_{1,2}$):

Queues for the arriving jobs. Queuing discipline: FIFO. Physical representation: stacks of card decks,... Infinite length.

c) SP_1 (res. SP_2):

"Strategy points." The strategy point constitutes a major feature of the model. At the strategy point, a decision has to be made whether a job remains in its original network or is sent to the other one. This decision depends on conditions such as: job length, length of waiting line in $WL_{i,j}$, processor-utilization.

The type of the conditions and their combination can be chosen from a great variety of possibilities and reflect the user's resource allocation strategy. A simple example for such a strategy is the following one: if processor P_2 is much more powerful than P_1, members of set $\{U_1\}$ could pursue the policy of sending all jobs with a job length above a certain threshold to $WL_{2,2}$ and thus to P_2 for faster execution. In spite of additional transmission costs (which can be specified in the model), they would be better off since their turnaround time is reduced; on the other hand, members of set $\{U_2\}$ are somehow penalized by the additional work load of "their" processor P_2. This, in turn, could lead to increasing usage costs of P_2 for members of set $\{U_1\}$. One need not proceed further in order to realize that the situation evolves into a competitive one, underlying the rules of game theory. Thus, the simple example already demonstrates that problems of this kind, with their numerous variations, ask for thorough treatment by means of models.

d) $WL_{2,1}$ (res. $WL_{2,2}$):

Queues of jobs waiting for execution in P_1 (resp. P_2). Queuing discipline: FIFO. Finite length.

e) P_1 (res. P_2):

Processors, representing resources which process jobs. Each processor is assigned a performance factor which reflects the capacity of the respective computer networks N_1 and N_2.

The output of the model consists of values for the following system characteristics: throughput, processor-utilization, transmission rates SP_i — $WL_{j,2}$, statistical distribution of queue length in $WL_{i,j}$, average waiting time in $WL_{i,j}$, turnaround time.

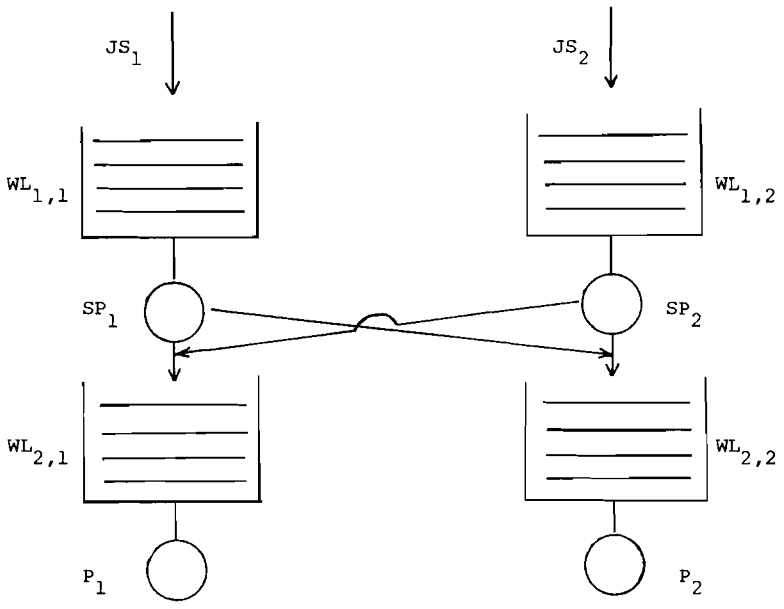


Figure 1. Strategy model.

4. IMPLEMENTATION AND USAGE OF THE STRATEGY MODEL

The strategy model was designed as a simulation model and implemented as a program written in FORTRAN IV. It requires a computer with at least 24 k working storage. The input consists of all simulation and model parameters and is made via cards. A printer must be available for the output of the simulation results.

The strategy model has already been used and has proven its usefulness in a number of investigations. Among others, the influence of a job profile containing a large number of short jobs and very few large jobs (typical situation in a college data center) on the system characteristics of coupled computer networks was studied. Essential conclusions concerning the best way of operation of such networks can be drawn from these investigations.

The experience of this strategy model has so far provided the conclusion that it is very important to study the operational conditions and especially the problem of optimal resource scheduling in order to make the best usage of computer networks. Good design of computer networks is only one prerequisite for satisfactory operation. It must be followed by intensive studies of the operational characteristics. Simulation models, such as the one described in this paper, seem to be a good tool to perform such analyses since they can be used not only by system designers but also by the users themselves. Such models lend a certain assurance that the users can employ their computer network (interconnected or not) in a somehow optimal way.

Synchronous Telecommunication Protocol

Edson C. Hendricks

1. INTRODUCTION

This working paper sets forth some basic principles which may be usefully applied to the design of telecommunication interfaces. An experimental IBM System/7 message-switching system called NET/7 uses a protocol whose design was derived from these concepts with encouraging results. NET/7 and the approach to protocol design described in this paper are completely experimental in nature, and are not IBM products. Synchronous Data Link Control (SDLC) [1], has become recognized as IBM's standard line discipline. The ideas expressed in this paper were developed completely independently of SDLC, are not necessarily in conflict or competition with the concepts of SDLC, and do not represent any kind of IBM standard or policy.

2. EXPOSITION

Two logical processes can be said to be "synchronous" with respect to each other only if their relative timing is predictable in the sense of being definite. Pairs of logical processes which do not exhibit this property would, correspondingly, be termed *asynchronous*. When two distinct logical processes which operate asynchronously exchange information, they must access a common communication channel. Also, they must cooperatively utilize their shared communication channel according to a single set of mutually understood rules which depend upon the exact nature of the communication channel in use. The word *protocol* is commonly used in reference to that set of mutually understood rules.

A communication channel is any set of facilities which can effect the transfer of information from one logical process to another. A communication channel can consist of an asynchronous pair of logical processes commonly utilizing another communication channel according to a particular protocol. Another "higher level" protocol may be employed by a different pair of communicating logical processes which utilize this sort of communication channel. The inductive nature of this definition will permit an arbitrary number of "levels" of protocol to be associated with the concept of a communication channel.

Numerous attempts have been made over the past decade or so to define protocols with sufficient completeness and generality so that further protocol design would not be necessary. However attractive or dubious (whichever) that objective may seem, it could become realistic only after experience has yielded sufficient knowledge to allow the establishment of a much more rigorous discipline in which, for example, functional claims could be precisely expressed and perhaps proven or disproven. For the present, the "black art" of protocol design will have to remain with us. The purpose of this paper is to illuminate some of the factors basic to protocol design, and to suggest methods of dealing with some identifiable difficulties.

It seems safe to say that the development of a general purpose telecommunication system is a very complex task, and that our ability to manage complexity is not yet boundless. We must recognize that each special provision which may be added to the protocol will potentially impair the clarity and consistency of its definition, compound the complexity of the associated logic, and lead directly to specification difficulties, decreased reliability, and increased cost. For this reason, a protocol ought to be limited to the simplest, least interdependent, and least restrictive set of rules which is free of redundancies and ambiguities within its level of functional jurisdiction. To meet this objective, the designer of a protocol must deal exclusively with precisely those functions which are not addressed by a lower level of protocol, and which cannot be coherently addressed by some higher level. In other words, protocols ought to be structured such that the function of any particular level is clearly defined with a manageable degree of complexity.

A particular protocol may or may not include an inherent distinction between the two logical processes which are in communication with one another. When no such distinction exists, the protocol is said to be "symmetrical"; otherwise, the protocol would be "asymetrical". A process which adheres to a symmetrical protocol could normally communicate with an identical copy of itself, using an appropriate communication channel. To cite an example in System/360 terminology, the protocols employed for communication between a central processing unit (CPU) and an I/O channel, or those between an I/O channel and a control unit, are clearly asymmetrical at some level. When there is no salient functional difference between the processes involved, the use of asymmetry in their communication protocol would add complexity, diminish flexibility, and provide little or no benefit in return. A protocol ought not to include any asymmetrical provisions, except where inescapeably imposed by functional requirements that cannot be coherently addressed at a higher level of protocol.

A telecommunication system must communicate and manipulate only two different kinds of information at any particular level of protocol. "Control" information is generated and interpreted by the cooperating processes, and is used to coordinate the mutual operation of their shared communication channel. "Data" are presented and accepted by those processes which make use of the telecommunication system, and that system seeks to communicate this kind of information from process to process with a very high degree of fidelity. Whenever a protocol requires or permits the communication of information which is intended to exhibit both properties at once, it necessarily restricts the design of the interface between the telecommunication facility and the processes which utilize that facility, while increasing its own complexity. Since such restrictions can impair the efficiency and flexibility of the telecommunication system, any provision for "hybrid" (both control and data) information in a protocol ought to be very thoroughly justified, or avoided entirely.

A great deal of effort has been devoted to the specification of a set of line control procedures called Binary Synchronous Communication, BSC [2,3]. While some difficulties with BSC still remain, it has achieved wide acceptance, and it provides many valuable concepts and techniques which may be readily utilized.

3. TRANSMISSION PROTOCOL

The lowest level of protocol must provide for framing of transmissions. BSC defines a certain synchronizing sequence which allows the receiving terminus to establish the alignment of a transmission, and an initial sequence to signal the beginning of the transmission data. A contextual ending sequence is also defined to allow the receiving terminus to detect the end of a transmission. These BSC framing conventions are perfectly acceptable because they are completely adequate for their purpose. (In a full-duplex environment, synchronization need not be abandoned. In this case, a continuous idle sequence would be maintained between transmissions, and the presence of a transmission would be detected by the receipt of a nonidle initial sequence).

BSC error checking is performed by means of a check sum which is accumulated by both transmitter and receiver, inserted at the end of the transmission by the transmitter, and compared to the receiver's check sum upon receipt. This error checking is performed only for that information transmitted in text mode, which is set and reset according to certain contextual control sequences. Ordinarily, data are transferred in text mode and control sequences are not, meaning that line errors occurring in control sequences may not be detectable. This is puzzling since the provision for nontext mode serves no apparent purpose, and errors will occur with equal probability in either type of information. Even if errors in control sequences were consistently detectable, BSC fails to provide completely adequate techniques for dealing with their occurrence. To the extent that the hardware provides the facility, protocols should be structured such that every transmission of control and data information is checked for errors. Higher level control techniques can be designed so that control sequences received with an indication of a possible error may be discarded, with no catastrophic effects, in a manner which will not require awkward recovery procedures.

BSC provides for transparent mode in which data of any bit configuration may be transferred with no possibility of its misinterpretation as control information by the receiver. As is the case with text mode, initiation and termination of transparent mode are effected through the use of contextual control sequences. Nontransparent mode may provide some very slight improvement in line efficiency, but its appropriateness appears to be limited to special purpose applications. In order to provide for simple and reliable general purpose data transmission, all transmissions of both control and data information should be in transparent mode. Conventions for distinguishing between data and additional control information are to be established at a higher level of protocol.

For our purposes, transmission protocol includes only matters of framing, error checking, and prevention of control sequence ambiguity within this context. The choice of this set of control functions as a separate protocol is somewhat arbitrary. The decision to separate these functions from others is most closely related to the nature of existing line-controlling hardware, which is often designed to automatically perform much or all of the functions. As a result, the functions tend to be completely and explicitly predefined in a manner that would be extremely difficult to modify. The limitation of the transmission protocol to the stated functions is a manifestation of the stated objective of minimizing complexity without sacrificing flexibility.

4. EXCHANGE PROTOCOL

The transmission protocol provides only for the coherent transfer of a sequence of information from one process to another, such that the presence of an error is detectable. Two logical processes utilizing a transmission line and associated modulation equipment according to this protocol constitute a certain communication channel. A different pair of logical processes may utilize this communication channel in a conversational fashion to transfer many sequences of information, repeating as necessary to overcome the effects of detected transmission errors. We postulate such a pair of processes, one of which accesses a single sequential source of data "blocks," and transfers these blocks in the described fashion to the other process, which sequentially deposits the blocks upon receipt in a single sink. The exchange protocol will comprise exactly that set of rules through which these processes may function cooperatively in this manner.

The exchange protocol will address the sequencing of transmissions. In a half-duplex environment, an elementary rule provides that one process may begin transmitting after a short delay following the termination of transmission by the other process; and that a process which is not transmitting must be prepared to receive a transmission from the other process. In a full-duplex environment, a process may begin a transmission at any time and must be prepared to receive transmissions from the other process at all times. The exchange protocol established the procedures to be followed by a process when error conditions are encountered. Some care must be exercised in the formulation of these procedures if unnecessary complexity and functional restrictions are to be avoided.

Basically, BSC provides that a process, that receives a block of data from another process utilizing the shared communication channel must respond, and may respond positively ("ACK") or negatively ("NAK"). The transmission of a negative response is to follow the receipt of a block which is indicated to be in error. The receipt of a negative response is to result in the retransmission of the last data block transmitted. The receipt of a positive response is to result in the transmission of the block which sequentially follows the last block transmitted. This approach fails to take into account some moderately degenerate circumstances which are not uncommonly encountered, especially when ordinary telephone lines are in use. There is the possibility that, for one reason or another, the two processes may accidentally fail to agree to the identity of the "last block transmitted." If the "last block transmitted" turns out to be a negative acknowledgement, the receipt of a negative acknowledgement will lead to an awkward situation. Some additional related control conventions have been included in BSC, perhaps with the intention of remedying difficulties of this kind. Positive acknowledgements are to be of odd or even parity to avoid losing blocks, although the precise technique for treatment of the receipt of an acknowledgement exhibiting incorrect parity is not made entirely clear. The ENQ character is apparently to be used as a negative acknowledgement to a control sequence, although error checking of control sequences is not provided, and the ENQ character is used differently in other contexts. Difficulties of this kind arise not from the nature of the problem, but rather from a subtle deficiency in the solution technique employed.

The essential decision required of the process with a sequence of data blocks to transmit concerns the choice of a block to be transmitted at some particular time. The process that receives the data blocks possesses all the information required for that choice. The technique provided by BSC requires the receiver to positively or negatively acknowledge previous receipt of a block. By implication, this amounts to a request for retransmission or no retransmission, and, only by further implication, to a request for transmission of some particular block. The potential for ambiguity here could be dispelled if a mutually understood technique for identification of individual blocks were established. A number of existing telecommunication protocols employ block serial numbers for similar purposes. The exchange protocol provides that the receiver will explicitly request transmission of each block by its serial number, and that the transmitter will respond to the receipt of such requests by transmitting the requested block.

The transmitting process functions by associating an initial serial number with the first block it obtains from its source, and increments that serial number by one for each successive block so obtained. The receiving process requests transmission of particular blocks by explicitly including in a request control sequence the block serial number, which it generates in a like manner. Upon receipt of a request, the transmitting process transmits the block associated with the requested serial number, if possible. If for any reason a block is not successfully received in response to a request, the same request is repeated, indefinitely as otherwise appropriate. Requests are checked for transmission errors with near certainty, and are simply discarded when they are indicated to be in error or are otherwise unacceptable. These provisions entirely remove the need for negative acknowledgements, and so nothing analogous to a NAK response is included in the exchange protocol.

The process which receives data blocks is required to ensure that the blocks are delivered to its sink in precisely the same order in which they were presented to the transmitting process by the source. The exchange level "L" is an attribute of the exchange protocol which is defined as the maximum number of data blocks that can be concurrently stored and accessed by a process outside of the sources and sinks. In other words, the exchange level is the minimum number of data block buffers available to each communicating process, and must be a positive integer greater than zero. A receiving process may not request a block whose serial number is greater than the sum of L-1 and the lowest serial number which has been requested and not successfully received. Conversely, a transmitting process may interpret the receipt of a request for a particular serial number as an implicit acknowledgement that the receiving process has successfully received all blocks of serial number less than the requested serial number minus L-1. The serial number will have a maximum value which will "wrap" to the initial serial number value when incremented. To avoid ambiguity, this maximum value must be at least L plus the initial serial number value. A larger range of serial numbers may be established as desired.

In a half-duplex environment, an exchange level greater than 1 cannot be utilized to advantage. In the special case of an exchange level of 1, the block serial number need not be explicitly attached to its data block by the transmitting process, since the receiving process will possess sufficient information to positively associate each block received with its serial number. If the exchange level is greater than 1, each data block transmitted must be labeled with its serial number to provide positive identification to the receiving process. Precise rules for the sequencing of requests and data block transmission are system design issues and must be established as part of the protocol implementation design. The exchange protocol as described may be easily generalized to allow the communicating processes to receive and transmit blocks at once, in an interleaved fashion. In this case, each transmission would comprise an explicit request for a block by serial number, followed by a requested block of data when appropriate. The establishment of conventions for interpretation of this kind of composite transmission poses no serious problem. An exchange level must be defined for each direction of data transmission, and the two need not be identical. The sum of the two exchange levels specifies the maximum number of data block buffers concurrently required by each process. No sequence of detected line errors and malfunctions will result in any ambiguities, regardless of the magnitude of the problems. The conceptual simplicity of the described scheme for request and data block exchange provides the potential for improved reliability at reduced overhead cost.

Additional functions could be provided within the basic exchange protocol structure. An explicit request for no block transmission in place of a request for a block by serial number might be appropriate in circumstances where the receiver's data block sink is temporarily or permanently not available, neatly obviating the need for the BSC "Wait Before Transmit" functions. (Note that this could not be so easily accomplished if explicitly serialized acknowledgements, rather than requests, were to be employed). There might be a need for the exchange of certain information related to the status of the communicating processes. A formated block of such information as is required could be maintained by each process and transmitted to the other upon receipt of a unique and explicit request. This sort of function would best be addressed at a higher level protocol, and its inclusion in the exchange protocol would necessitate the addition to the process model of a separate status block source and sink.

In a situation where neither process has any data blocks to transmit to the other, the exchange protocol would allow them to simply alternate transmission of requests to each other. It may be desirable to halt this process when it imposes an added system load of some kind. In order to properly deal with the matter, we define *exchange mode*. In a loose sense, exchange mode is active only when the processes are exchanging transmissions, as described above, and may be deactivated through a further exchange protocol provision. When a process has no data block to transmit, it could transmit a block request along with an explicit request to terminate exchange mode. Upon receipt of such a request to terminate exchange mode, a process with no data block to transmit could return a request to terminate exchange mode, and that process would consider exchange mode terminated at that time. Upon receipt of a request to terminate exchange mode immediately following the transmission of a similar request, a process would consider exchange mode terminated at that time. Exchange mode would be reinitiated by the receipt of any sequence conforming to the transmission protocol. Similar provisions can be imagined for graceful disconnection of a transmission line in several different ways, and exact procedural details would be established as part of the protocol implementation design.

5. RELATED PROTOCOLS

A comprehensive telecommunication system or network, perhaps will need to provide many facilities which are not implied by the design of the exchange protocol. Most existing protocols include rules for identification of processes or groups of processes, addressing of data by means of such identifiers, data description, and so on. These matters ought to be addressed only as necessary, and any number of higher-level protocols could be concerned with these functions. For instance, the described data blocks might include "headers" which may be generated, interpreted, and acted upon only by processes which utilize the communication channel provided by the exchange protocol. Explicit provision for that kind of information management at the exchange protocol level would lead only to increased complexity and a loss of generality.

Very little has been said about the nature of communication between pairs of processes which have been described as adhering to different levels of protocol. As an example, a process adhering to the exchange protocol was postulated to access data block sources and sinks. Alert readers have probably already recognized that such sources and sinks might comprise additional processes, communication channels, and protocols which cannot be viewed as higher or lower level with respect to those protocols described. It is not really possible to say very much about that subject without establishing some details related to the overall nature of the environment shared by those processes. These details will vary drastically from system to system, and have been left unconstrained so that factors which have not been addressed here may be taken into account as necessary. This leaves open the possibility that these processes, which have been regarded as separate for the purposes of protocol design, can be otherwise viewed as separate parts of a single process.

Acknowledgments

The Cambridge Scientific Center as a group has contributed immeasurably to the design approach described. In particular, discussions with Charlie Salisbury, Allen Springer, and Peter Vile led directly to the development of this paper.

References

[1] Donnan, R.A., and J.R. Kersey, Synchronous Data Link Control: A Perspective. *IBM Systems Journal*, 13, 2 (1974), 140-162.

[2] Eisenbies, J.L., Conventions for Digital Data Communications Link Design, *IBM Systems Journal*, 6, 4 (1967), 267-302.

[3] General Information - Binary Synchronous Communications, IBM Systems Reference Library, GA27-3004.

[4] James, M., *Systems Analysis for Data Transmission*, Prentice-Hall, Englewood, New Jersey, 1972.

APPENDIX A

Throughput and Block Size

Questions dealing with synchronous telecommunication line throughput and block size are frequently encountered [2,4]. Dynamic adjustment of data block size adapted to observed error rate is sometimes suggested as a means for maximizing line efficiency. The analytical derivation of a method for calculating expected throughput and optimum block size is outlined below.

DEFINITIONS

S - Block Size: The quantity of data to be sent during a single synchronous transmission (bytes).

V - Data Rate: The constant speed at which data are transmitted during synchronous transmission (bytes/second).

C - Block Overhead: The constant (or average) time delay between the end of transmission for a block and the beginning of transmission for the next block (seconds).

R - Error Rate: The average arrival rate of events which cause data transmission errors (errors/second).

T - Throughput: The expected average rate of transfer of usable data across the transmission line (bytes/second).

ASSUMPTIONS

a) Error events arrive with a Poisson distribution.

b) The arrival of one or more error events during the transmission of a block will result in the retransmission of that block.

The probability P of successfully transmitting a single block is,

$$P = e^{-RS/V} \quad . \qquad (1)$$

Let N be the number of data blocks to be transmitted; and let N' be the expected number of actual block transmissions required to successfully transmit N blocks ($N' \geq N$). Then,

$$N' = N + N'(1 - e^{-RS/V} ,$$

or,

$$N' = Ne^{RS/V} . \tag{2}$$

The expected throughput T is expressed as,

$$T = \frac{NS}{N'(C + S/V)} . \tag{3}$$

Combining expressions (2) and (3),

$$T = \frac{NS}{Ne^{RS/V}(C + S/V)} ,$$

or,

$$T = \frac{SVe^{-RS/V}}{S + CV} . \tag{4}$$

The variation of throughput with respect to change of block size is expressed as,

$$\frac{dT}{dS} = \frac{(CV^2 - CVRS - RS^2)e^{-RS/V}}{(S + CV)^2} . \tag{5}$$

The optimum block size S(opt) is that block size for which expected throughput is maximum. S(opt) can be found by equating to zero the quadratic in the numerator of expression (5) above and solving for S,

$$S(opt) = (CV/2)((1 + (4/CR))^{1/2}-1) \qquad CR > 0 \quad , \quad V > 0 . \tag{6}$$

In many ordinary situations, the product CR is positive and small. For purposes of simplifying calculation of S(opt), the following approximations to S(opt) may be useful,

$$S(\text{opt})(\text{approx.}) = CV((CR)^{-1/2} - 1/2) \qquad 0 < CR < .5 \quad , \tag{7}$$

and,

$$S(\text{opt})(\text{approx.}) = V(C/R)^{1/2} \qquad 0 < CR < .05 \quad . \tag{8}$$

The Hahn-Meitner-Institut Computer Network

W. Lehmann-Bauerfeld, and H.W. Strack-Zimmermann

1. INTRODUCTION

The Hahn-Meitner-Institut uses a large (≈ 20) number of process-control computers from different manufacturers for its research in nuclear physics and radiation chemistry. In addition, a Siemens 4004/151 is operated as the central computing facility.

Originally data were carried by tape from the process central machines to the central facilities in a "bicycle on-line" fashion.

Under the network project, the scientist is allowed to send samples of his data via fast private or slower postal lines to the central machine. Partial results can be computed on-line to reduce experiment plus data aquisition time. The project is partly funded by the Ministry for Research and Technology of the Federal Republic of Germany (Grant No. 411-5939-DV 2.019) in order to enhance the national know-how in data communication and message switching. The development is carried out as a joint effort between the Hahn-Meitner-Institut and Siemens AG.

2. BASIC NETWORK SPECIFICATIONS

2.1 User Facilities

The network offers to the users of the small process control computers enhanced and new facilities such as central data storage and on-line access to larger computing capacity. The new facilities can be grouped as follows:

DIALOG:	participation in the time-sharing service using local process control computer terminals;
EITC:	extended intertask communication between programs running on different computers;
RJE:	remote job entry from process control computer peripherals;

SPOOL-OUT:	remote spool-out to process control computer peripherals;
RFA:	remote access to the central file base.

Transfers between different process control computers are also possible; this eases the implementation of physics experiments in which more than one local machine is needed for control and data acquisition.

2.2 Network Topology

As may be seen in Figure 1, for the first stage, a simple star layout is foreseen; the higher level protocols are designed, nevertheless, to allow for a more complex configuration in the future. There is no special host computer in the entire network. Larger machines such as the Siemens 4004/151 are only connected via a faster data transmission channel, they offer a wider variety of services, and are always kept on-line.

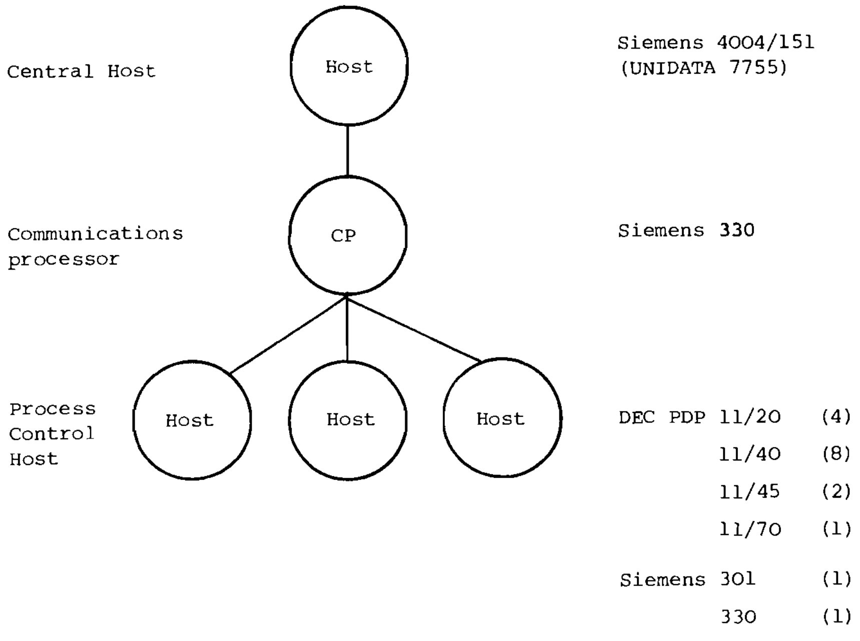


Figure 1.

3. HARDWARE OVERVIEW

A Siemens 330 computer is used as a store and forward message handler. The machine is equipped with an independent I/O processor and 64 K words (a 16 bit) of core memory. A disk and a magnetic tape unit are included for bootstrapping, event logging, and development purposes. A teletype is used to enter commands and to retrieve snapshots of current activities.

The central host is currently a Siemens 4004/151, which will be exchanged mid-1976 against a UNIDATA 7755. The Siemens 4004/151 mainly provides a time-sharing service for a large number of interactive users but is also capable of processing background batch jobs. Some real-time facilities are implemented in connection with this project.

All of the process control machines are large enough (32 K words, disk) to run under a multitasking real-time operating system. Most process control tasks are handled locally by these machines. The connection to the central machine is used if additional computing power and file storage are needed or if link modules are drawn out of the centrally kept process control library.

3.1 Data Transmission Equipment

The central host and the communication processor are coupled on a multiplexor channel interface level with a transfer speed of 216 K byte/s.

The process control machines on the site are connected via HMI build DMA interfaces and 200 K bit/s serial data communication channels provided by Siemens. For outside connections, MODEMs and a 4800 bit/s variant are used. Since the private lines are extremely noise free, only simple byte parity is used for error detection.

All data transmission if fully transparent and 8-bit oriented. Control information is passed on the hardware level by preceding each byte with two command bits.

4. SOFTWARE STRUCTURE

Communication within a network of autonomous computers can be imagined as a continuum of communicating processes.

In the HMI network implementation, the communicating processes can be classified in a hierarchy of four different levels. On each level, specific control information (header) and transparent data are exchanged. The data of a lower-level transfer may contain control information for the higher-hierarchy levels. For the data exchange between different levels in one machine, the standard interprocess communication of the local operating system is used as far as possible. Communication between processes on the same level located on different machines are controlled by the communication protocols and procedures.

On each level, a maximum size for a data exchange was introduced. An extremely modular design, by which each level performs segmentation and reassembly of communication blocks for the next lower and higher level and has its own error procedures, will help to adapt the network to future enhancements and gateways to nonlocal networks. Figure 2 demonstrates the level concept.

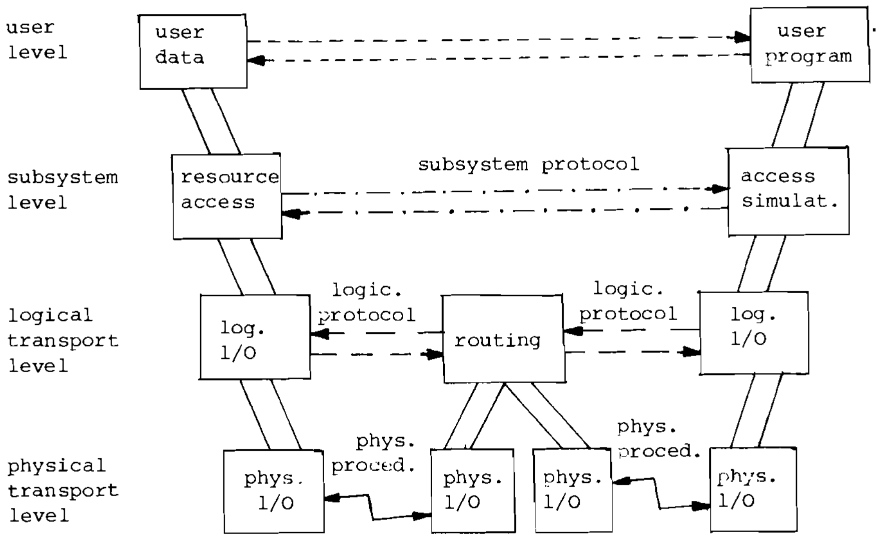


Figure 2.

4.1 Physical Transport Level

The I/O handler services all physical resources such as line, interface, and I/O buffer. All actions necessary to transmit a physical packet of data via a point-to-point connection, which are not performed by hardware, are handled on this level.

In the current stage of implementation, the Siemens NEA 2 half-duplex procedure is used for all connections even though a variety of transmission hardware is in operation.

4.2 Logical Transport Level

The communication access level governs the network as multipoint connections. It organizes data routing and administers the resources used by the physical transport level. The communication access method resides in all connected machines and must be capable of analyzing the standard logical control header. The communication access level sends without prior announcement flow control logic is therefore provided to slow down the sending communication access process in case of congestion.

Currently the Siemens NEA 2 station protocol is used on this level.

4.3 Subsystem Level

If a user process or terminal would like to use resources on a nonlocal machine, the subsystem level serves as an intermediary. The local subsystem acts as a simulator while the communicating subsystem on the other machine does the actual control of the resources. In general, the actual data exchange is invisible to the user process.

Currently the following subsystems are implemented:

DIALOG:	timesharing access to a nonlocal machine;
EITC:	intertask communication between tasks on different machines;
ADMINISTRATION:	general network administration and information service.

The following subsystems are planned:

RJE:	remote job entry;
SPOOL-OUT:	spool-out to nonlocal peripherals;
RFA:	remote file access;
RLA:	remote library access.

4.4 User Level

The user must be able to demand access to all network resources just by adding some additional parameters to his operating system interface. In the first stage, this will be provided on an assembler and job control macrolevel.

It is hoped that some work on a common job control language can be done in the future. In addition, a totally compatible interface to all services of the intertask communication on a FORTRAN call level is under implementation, which is of extreme importance in a community of physicists.

5. CURRENT STAGE OF IMPLEMENTATION

The network is currently under testing. It is hoped to start a partial user service in early 1976.

Acknowledgements:

The design and implementation of the network is carried out by Siemens, ZN, Berlin; Siemens, DV WS SP, Munich; Siemens, E STE 213, Karlsruhe; Hahn-Meitner Institut, Berlin.

Special thanks to Mr. Goergens, project leader Siemens, and to Professor Töpfer and Dr. Baum, HMI, who initiated the project and guided it during the initial difficult time.

Interprocess Communication in the HMI Network: Description of an Implementation

F. Vogt

1. INTRODUCTION

Interprocess communication is defined as data exchange between different processes. In order to exchange data, processes must be addressible. A process in a network environment is addressible by the concatenation between the communication name and the host computer identification. Data are transmitted in discrete portions called messages or, more exactly, transport elements. To exchange transport elements between producer and consumer, high-level transport protocols must be defined. In addition, a lower-level link protocol is needed between neighboring hosts. In the HMI network, a Siemens standard protocol, called NEA2 (NEtwork Adaptation), is used on the link level.

Date exchange between different processes is performed by a SEND and a RECEIVE directive. The network can be thought of as a virtual machine which is capable of performing the interprocess communication on the SEND-RECEIVE level. The first part of this paper is a description of the communication on the virtual machine level (Figure 1).

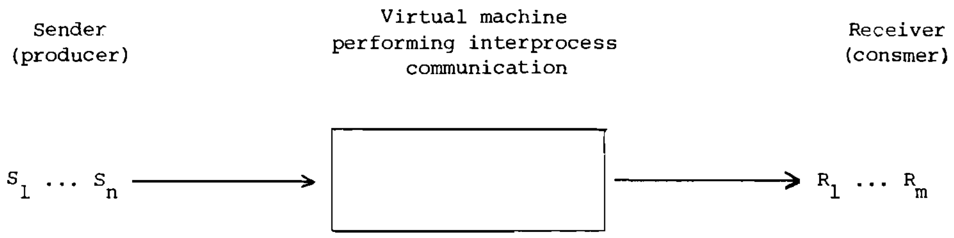


Figure 1.

The second part describes some implementation details on which the virtual machine is based.

2. DESCRIPTION OF THE INTERPROCESS COMMUNICATION ON THE VIRTUAL MACHINE LEVEL

2.1 Requirements

A sending process (called sender) can produce a message at a time when the receiving process (called receiver) is not yet ready to accept it (asynchronous machine). This requirement makes it necessary to use temporary storage for data buffering until the receiver is able to accept. The temporary storage is called mailbox and is located in the receiver host computer.

2.2 Limitations

If S is the set of data sent by all senders S_i to one process and R the set of data to be received by this process, two limitations can be formulated. If M is the maximum size of the mailbox, $0 \leq S - R \leq M$ defines the finite capacity of the mailbox in relation to S and R. Second, the receiver cannot consume faster than data are produced by the senders $(0 \leq R \leq S)$.

2.3 Synchronizing Rules (Figure 2)

B \ A	S	R	K	U
S	$X^{1)}$	$X^{2)}$	$X^{4)}$	$O^{6)}$
R		$X^{3)}$	$X^{5)}$	$O^{7)}$

A,B:	different processes
S:	process is a sender
R:	process is a receiver
K:	process is inactive but known
U:	process is unknown

Figure 2.

ad 1) The limitation in this case is that the mailbox capacity of both is limited $(S_A \leq M_B, S_B \leq M_A)$. If one of the mailboxes becomes full, the involved sender will get back a negative acknowledgement.

ad 2) To receive data means to get data out of one's own mailbox. No endless waiting can occur because the receiver doesn't wait for data from the sender; he only asks if there are data in his mailbox. In some special cases, it may be necessary to do exchange data between synchronizing processes (mailbox capacity, transfer rate). The synchronous machine is a subset of the asynchronous machine; this case is, therefore, also supported.

ad 3) R_A —— R_B is possible if there are still data in the mailboxes (M_A, $M_B \neq \emptyset$).

ad 4) S_B fills the destination mailbox M_A.

ad 5) $R_B \leq M_B$.

ad 6) If data are sent to a nonexisting process, it will be transmitted through the network, but cannot be queued in a destination mailbox. A negative acknowledgement gets back to the sender. By using information messages, this case can be exchanged.

ad 7) M_B does not exist.

2.4 Message Segmentation

On sender and receiver level, it is possible to make a process-specific segmentation of the set of data to be exchanged. This segmentation is totally independent of the size of transport elements (defined in the transport protocol).

2.5 Parameters for the SEND directive

Data buffer description;

Destination address;

End of data indicator (end of segment, end of data);

Acknowledgement required for each segment/not required;

Code description;

Mailbox type (core, disk, synchronous data exchange);

Segment number (to identify the acknowledgement).

2.6 Parameters for the RECEIVE directive

Data buffer description;

Source address.

2.7 Other Requirements to Perform SEND - RECEIVE

A general control block is required. It is created by an additional macrocall. The initial values of this control block can be created at assembly or at run time.

3. IMPLEMENTATION IN A DEFINED OPERATING SYSTEM ENVIRONMENT

3.1 General Aspects

To integrate new network facilities into an operating system, one must take care not to build another system "besides" the existing one; the aim has to be a full integration of the network facilities. Two different operating systems are to be adapted in the HMI network. One is a big time-sharing system (BS 2000 on a Siemens 4004/151), the other is a real-time operating system (RSX on DEC PDP 11) used for process control purposes. Both have a local intertask communication--a good base to do a network adaption. While the implementation in BS 2000 is done by the Siemens AG,[1] the HMI is responsible for the network adaption of RSX.

To discuss the software structure of the virtual machine, it is helpful to imagine the sending and receiving processes as a task using SEND-RECEIVE directives.

3.2 Software Structure of the SEND Directive

The SEND procedure (common code) creates an information block containing all necessary SEND parameters (see section 2.5); it then hands this block over to the IPOT (Figure 3). IPOT maps the address space of the data buffer into its own space and performs the segmentation into transport elements (with all the necessary header information). The assembly of transport elements to a link packet and the physical "data" transmission over the line is done by the output handler.

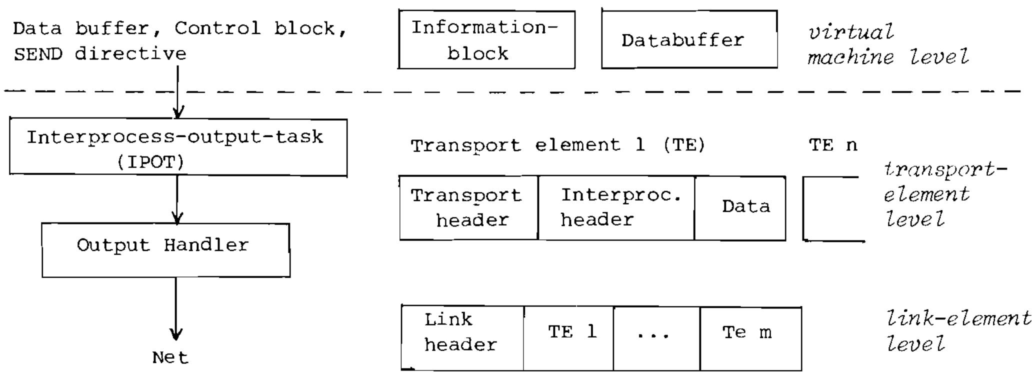


Figure 3.

[1]The design and implementation is carried out by Mr. Wentzien from Siemens, Berlin.

3.3 Software structure of the RECEIVE Directive

The mailbox contains all information which was sent to it by any task in the network. A mailbox is implemented as a queue of all transport element description blocks for a certain task and the associated transport element data buffers. These buffers can be located in core or on a disk file. The RECEIVE procedure dequeues the transport element description blocks and reads data in the buffer of the receiving task (Figure 4). The direct access RECEIVE does not need a mailbox; the task waits until data can be moved directly from the interprocess input task into the specified data buffer (special synchronizing is necessary).

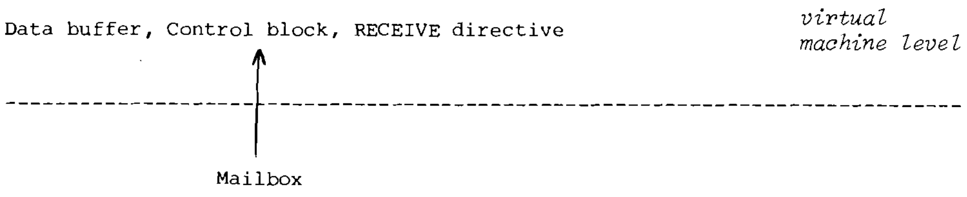


Figure 4.

3.4 The Structure of Mailbox Input (Figure 5)

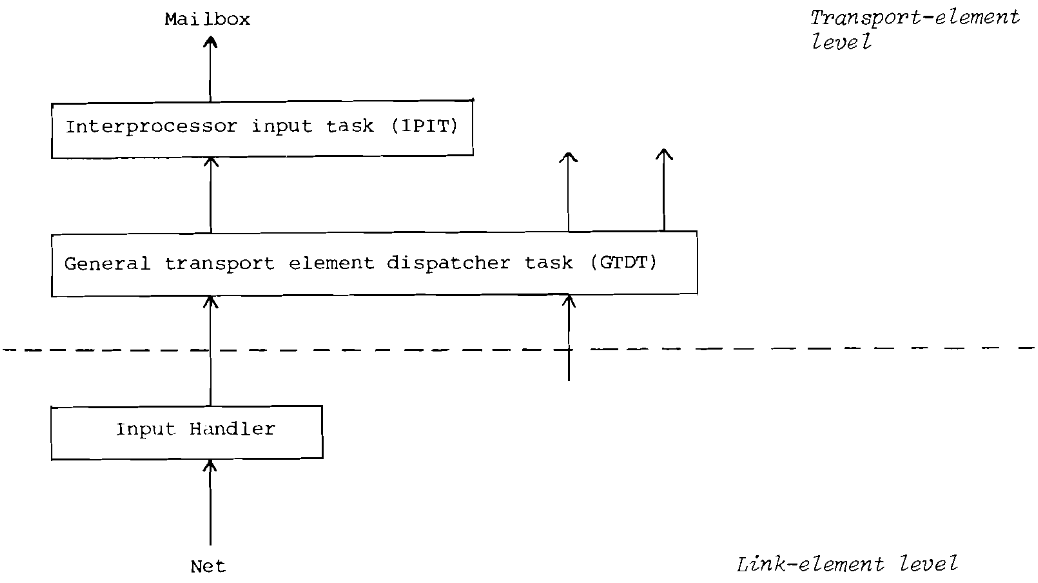

Mailbox
Transport-element level
Interprocessor input task (IPIT)
General transport element dispatcher task (GTDT)
Input Handler
Net
Link-element level

Figure 5.

The GTDT is always waiting for a transport element to arrive from an input handler; the GTDT distributes the elements by transferring only the addresses to the associated subsystems (in this case IPIT) and queues these. IPIT then maps the transport-elements into the proper receiver task mailbox. In the case of direct access, RECEIVE, IPIT maps the transport-element data directly from GTDT to the specified buffer of the receiver task.

The general transport-element dispatcher (GTDT) is not restricted to interface to a special type of network because it is designed to handle transport elements from different input handlers and can cooperate with any subsystem.

Stability and Control of Packet-Switching Broadcast Channels

G. Fayolle, E. Gelenbe, and J. Labetoulle

1. INTRODUCTION

Computer networks using packet-switching techniques have been implemented [1,3,5,8,12,13] in order to allow a large community of communicating users to share and transmit data and to utilize excess computing power which may be available at remote locations in an efficient manner. In this paper, we shall be concerned with packet-switching networks using radio channels similar to the ALOHA system [1].

We consider a large set of terminals communicating over a single radio channel in such a way that a packet is successfully transmitted only if its transmission does not overlap in time with the transmission of another packet; otherwise, all packets being simultaneously transmitted are lost. A terminal whose transmission is unsuccessful is said to be *blocked*; it has to repeat the transmission until it achieves success. A terminal which is not blocked is either *active* or it is transmitting a packet. The operation of the system is shown schematically in Figure 1 where the different state transitions of a terminal are shown. Since the only means of communication between terminals is the channel itself, it is not easy to schedule transmissions so as to avoid collisions between packets. It is also obvious that a terminal would in no case transmit more than one packet simultaneously.

Various methods for controlling the transmission of packets have been suggested. The simplest is to allow terminals to transmit packets at any instant of time. The second method, known as the *slotted ALOHA* scheme has been shown to increase channel throughput over the first method [2]. Here time is divided into "slots" of equal duration; each slot can accommodate the transmission time of one packet, and packets are all of the same length. Packet transmission is synchronized, so as to be initiated at the beginning of a slot for any terminal, and it terminates at the end of the same slot. Other methods have been suggested elsewhere [9].

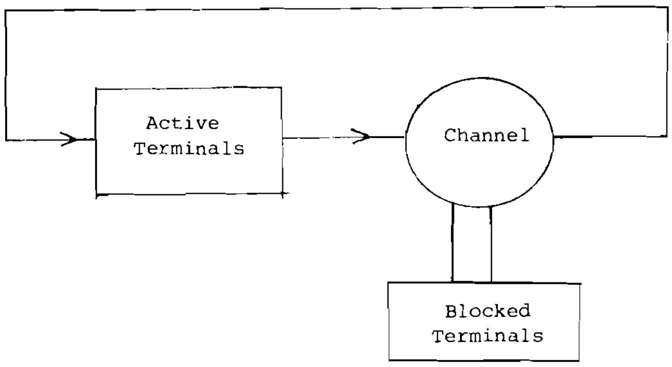


Figure 1.

Kleinrock and Lam [10] have discussed the stability problem of the slotted ALOHA channel. They give qualitative arguments and results based on simulations indicating that the channel becomes saturated if the set of terminals is very large, independently of the arrival rate of packets to the channel, saturation being the phenomenon whereby the number of blocked terminals becomes very (or arbitrarily) large. They also compute the expected time to attain a given level of saturation. In reference [11], policies designed to optimize the throughput of the channel, defined as the expected number of successful transmissions per slot, are presented.

The purpose of this paper is to give a theoretical treatment of some control policies which can be applied to the broadcast channel in order to stabilize it and to maximize its performance. We first recall the proof of instability in [6], extending it to the finite source model taken in the limit as the total number of terminals becomes very large, and showing that channel instability implies that the equilibrium value of the throughput is zero. Two simple control policies are then presented and necessary and sufficient conditions for stability of the controlled channel are derived. Bounds for the equilibrium value of the channel throughput with these policies are obtained. Finally, we give a simple algorithm for the approximate implementation of this policy and exhibit some simulation results showing its performance.

2. A MATHEMATICAL MODEL

A precise definition of stability can only be considered in the context of a model of the behavior of the broadcast channel. In this section, we present a model identical to the one we have considered in an earlier paper [6], except that we shall take into account here both finite and infinite source systems.

Assuming that the slot and the time necessary to transmit a packet are of unit length, consider N(k) the number of blocked terminals at the instants $k = 0,1,2,\ldots$ when a slot begins. Let X_k be the number of packets transmitted by the set of active terminals during the k-th slot, and denote by Y_k the number of blocked terminals transmitting during the k-th slot. In the *infinite source model*, (X_k) is the sequence of independent and identically distributed random variables with common distribution given by,

$$\Pr(X_k = i) = c_i, \quad i \geq 0 \quad . \tag{1}$$

In the *finite source model*, assuming that the total number of terminals in the system is M, we let the event $(X_k = i/N(k) = j)$ be independent of values of X_t for $t < k$; its probability is given by

$$q_j(n) = \Pr(X_k = j/N(k) = n) = \binom{M-n}{j} b^j (1-b)^{M-n-j} \quad , \tag{2}$$

for $0 \leq j \leq M-n$, where b is the probability that any one active terminal transmits a packet during a slot.

For both models, we shall denote by f the probability that any one blocked terminal transmits a packet during a slot. We then define,

$$g_i(n) = \Pr(Y_k = i/N(k) = n) \quad , \tag{3}$$

where we assume that the event $(Y_k/N(k))$ is independent of Y_t for $t < k$. Therefore,

$$g_i(n) = \binom{n}{i} f^i (1-f)^{n-i} \quad , \tag{4}$$

and more particularly,

$$g_0(n) = (1-f)^n, \; g_1(n) = nf(1-f)^{n-1} \quad . \tag{5}$$

Definition 1

The infinite source broadcast channel is *unstable* if, for $k\to\infty$, the probability $\Pr(N(k) < j) \to 0$ for all finite values of j; otherwise it is *stable*. For the finite source model, the system is unstable if the above condition is verified as we let $M\to\infty$, $b\to 0$, $M.b\to d$, where d is a constant.

The definition given here simply states that instability is verified if (with probability one) the number of blocked terminals becomes infinite as time tends to infinity.

Theorem 1

The broadcast channel is unstable both for the finite and infinite source model.

Proof

Let us first consider the infinite source model. The proof given here is identical to the one we presented in [6]. Let $p_n(k)$ denote the probability that $N(k) = n$. The following transition equation may be written for the infinite source model[1]:

[1]Equation (6) is valid for all $n \geq 0$ if we adopt the rule that $P_i(k) = 0$, $i < 0$.

$$p_n(k+1) = \sum_{j=2}^{n} p_{n-j}(k)c_j + p_{n+1}(k)g_1(n+1)c_0$$
$$+ p_n(k)(1-g_1(n))c_0 + p_n(k)g_0(n)c_1 \qquad (6)$$
$$+ p_{n-1}(k)(1-g_0(n-1))c_1 \quad .$$

On the right-hand side of Eq. (6), the first term covers the cases where two or more packets have been transmitted by the active terminals during the k-th slot; the second term covers the case in which exactly one blocked terminal has transmitted while no active terminal has done so. Notice that $\{N(k); k = 0,1,\dots\}$ is a Markov chain and that it is aperiodic and irreducible. It is ergodic if an invariant probability measure $\{p_n; n = 0,1,\dots\}$ exists satisfying Eq. (6) such that $p_n > 0$ for all n and where $p_n = \lim_{k\to\infty} p_n(k)$. To show that $\lim_{k\to\infty} \Pr(N(k) < j) = 0$ for all finite values of j, it suffices that the Markov chain representing the number of blocked terminals be not ergodic. Substituting p_n for $p_n(k)$ and $p_n(k+1)$ in Eq. (6), we obtain,

$$p_n = \sum_{j=0}^{n} p_{n-j}c_j + p_{n+1}g_1(n+1)c_0 + p_n(g_0(n)c_1 - g_1(n)c_0)$$
$$- p_{n-1}g_0(n-1)c_1 \quad . \qquad (7)$$

Let,

$$S_N = \sum_{n=0}^{N} p_n \quad . \qquad (8)$$

We then have for any $N \geq 0$,

$$S_N = p_{N+1}g_1(N+1)c_0 + p_Ng_0(N)c_1 + \sum_{n=0}^{N} S_{N-n}c_n \quad , \qquad (9)$$

or,

$$S_N(1-c_0) = \sum_{n=1}^{N} S_{N-n}c_n + P_{N+1}g_1(N+1)c_0 + P_N g_0(N)c_1 \quad , \tag{10}$$

or equivalently,

$$P_N(1-c_0) \le P_{N+1}g_1(N+1)c_0 + P_N g_0(N)c_1 \quad . \tag{11}$$

But then, from Eqs. (5) and (11), we have,

$$\frac{P_{N+1}}{P_N} \ge \frac{1 - c_0 - (1-f)^N c_1}{(N+1)f(1-f)^N c_0} \quad , \tag{12}$$

for any nonnegative integer N. This implies that the ratio $(P_{N+1}/P_N)\to\infty$ as $N\to\infty$, so that the sum S_∞ can only exist if $P_N = 0$ for all finite values of N; otherwise S_∞ is divergent, and this cannot be the case since the P_N, $N \ge 0$ define a probability distribution. Thus, the Markov chain representing the number of blocked terminals is not ergodic, and the broadcast channel under the infinite source assumption is unstable.

Now consider the finite source model. Using the rule that $P_i(k) = 0$ for $i < 0$, the transition equation for $0 \le n < M$ is,

$$\begin{aligned} P_n(k+1) = {} & \sum_{j=2}^{n} p_{n-j}(k)q_j(n-j) + p_{n+1}(k)g_1(n+1)q_0(n+1) \\ & + p_n(k)(1-g_1(n))q_0(n) + p_n(k)g_0(n)q_1(n) \\ & + p_{n-1}(k)(1-g_0(n-1))q_1(n-1) \quad . \end{aligned} \tag{13}$$

Defining, for $0 \le N < M$, the sum S_N as in Eq. (8) for the finite source model we obtain from Eq. (13) and substituting the stationary probability P_n:

$$\begin{aligned} & S_N + P_{N+1}g_1(N+1)q_0(N+1) + P_N g_0(N)q_1(N) \\ & + \sum_{n=0}^{N} \sum_{j=0}^{n} P_{n-j}q_j(n-j) \quad . \end{aligned} \tag{14}$$

Now take limit as in Definition 1: $M\to\infty$, $b\to 0$, $M \cdot b\to d$; we obtain $q_j(n) = \frac{d^j}{j!} e^{-d}$ for any j and n. Therefore, in the limit,

$$S_N = P_{N+1}g_1(N+1)q_0(N+1) + P_N g_0(N)q_1(N) + \sum_{j=0}^{N} \frac{d^j e^{-d}}{j!} S_{N-j} \quad , \tag{15}$$

and an argument identical to the one for the infinite source model can be now used to complete the proof of instability.

We note in passing that the finite source model in the limit (as we let the total number of terminals tend to infinity) and the infinite source model are not identical; in the infinite model, there is a nonzero probability of a transmission from active terminals in each slot even when we let $k\to\infty$; while for the finite source model in the limit as $M\to\infty$ no active terminal will transmit as $k\to\infty$.

In the context of this study, another measure of interest is the throughput of the broadcast channel. Indeed, this may well be the primary performance measure for the system under consideration.

Definition 2

The *conditional throughput* $D_n(k)$ of the broadcast channel is the conditional probability that one packet is successfully transmitted during the k-th slot given that $N(k) = n$.

Clearly, the conditional throughput cannot exceed one; it can also be defined as the expected value of the number of successful transmissions during the k-th slot conditional of there being n blocked terminals at the beginning of that slot.

Definition 3

The *throughput* of the broadcast channel is defined as,

$$D = \lim_{k\to\infty} \sum_{n=0}^{\infty} D_n(k)p_n(k) \quad . \tag{16}$$

The conditional throughput is,

$$D_n(k) = c^0 g_1(n) + c_1 g_0(n) \quad , \tag{17}$$

for the infinite source model; for the finite source model, we replace c_0 and c_1 by $q_0(n)$ and $q_1(n)$, respectively. This quantity is obviously independent of k; therefore, in the following we shall simply write D_n instead of $D_n(k)$.

Theorem 2

For $f > 0$, the throughput of the broadcast channel is zero for the infinite source model and for the finite source model as we let $M\to\infty$, $b\to 0$, $M \cdot b\to d$.

The proof is straightforward and not presented here.

3. CERTAIN CHANNEL CONTROL POLICIES

Various control policies for the broadcast channel have been discussed in [11] where these have been classified, roughly speaking, into three groups: policies which regulate access to the channel from the active terminals, those which regulate access from the blocked terminals, and mixed policies. In this section, we discuss two policies in some detail and give a definition of stability in each case. We see that this definition will be a variant of (or identical to) the definition given above. The first control policy which we shall describe typifies the first group of policies, and it may well be impossible to implement; the second policy is of the second group and has a better chance of being realizable.

3.1 A Threshold Control Policy

An input control policy as defined by LAM [11] is one which limits access to the channel from the active terminals depending on the present state and past history of the channel. Borrowing the terminology of Markov decision theory [7], a policy is said to be *stationary* if it only depends on the present state of the system.

The first policy we present is described in Figure 2. If the number of blocked terminals exceeds Θ, the threshold, an active terminal which wishes to initiate the transmission of a packet is not allowed to transmit and joins the *impeded set*; if not, the transmission takes place as in the uncontrolled channel. As soon as the number of blocked terminals decreases below Θ (this can only take place in steps of one), an impeded terminal joins the blocked set; thus, the number of blocked terminals can be less than Θ only if there are no impeded terminals. The retransmission rate of blocked terminals is constant. We shall refer to this scheme as the *threshold control policy*.

In this context, stability must be defined in terms of the number of impeded plus blocked terminals.

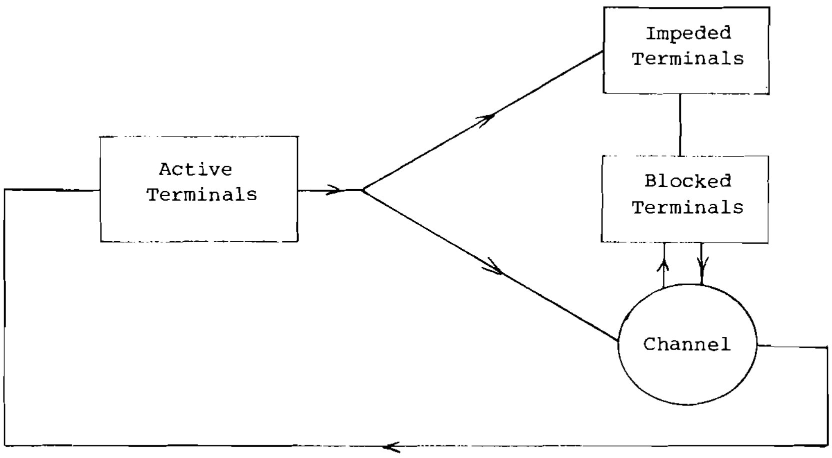


Figure 2.

Definition 4

Let U(k) be the number of blocked plus impeded terminals at the beginning of the k-th slot for the threshold control policy. The channel, with this control scheme, is unstable if the limit as $k\to\infty$ of $\Pr\{U(k) < j\}$ is zero for all finite values of j for the infinite source model; for the finite source model, the same definition is used as $M\to\infty$, $b\to 0$, $M \cdot b\to d$. Otherwise the channel is stable.

The following equations, which must be satisfied by the equilibrium probabilities p_n for the number n of blocked plus impeded terminals at the beginning of a slot, may be derived.

<u>$n \leq \Theta$</u>

$$p_n = \sum_{j=0}^{n} p_{n-j}c_j + p_{n+1}A_1(n+1)c_0 + p_n[c_1g_0(n) - c_0g_1(n)] \tag{18}$$
$$- p_{n-1}g_0(n-1)c_1 \quad .$$

<u>$n \geq \Theta + 1$</u>

$$p_n = \sum_{i=n-\Theta}^{n} p_{n-i}c_i + \sum_{i=1}^{n-\Theta-1} p_{n-i}c_i[1-A_1(n-i)] \tag{19}$$
$$+ \sum_{i=1}^{n-\Theta} p_{n-i+1}c_iA_1(n-i+1) + p_{n+1}c_0A_1(n+1) + p_n[1 - A_1(n)]c_0$$

where,

$$A_1(n) = \begin{cases} g_1(\Theta) & \text{if } n > \Theta \\ g_1(n) & \text{if } 0 \leq n \leq \Theta \end{cases} \quad .$$

Equation (19) may be rewritten as:

$$p_n = \sum_{i=0}^{n} p_{n-i}c_i + \sum_{i=0}^{n-\Theta-1} p_{n-i}A_1(n-i)[c_{i+1} - c_i] + p_{n+1}c_0A_1(n+1) \quad . \tag{20}$$

We obtain the following result concerning the stability of the threshold control policy. For simplicity let $A = g_1(\Theta)$.

<u>Theorem 3</u>

If the expected arrival rate of active packets $\lambda = \sum_{i=1}^{\infty} ic_i$ for the infinite source model is less than A, then the broadcast channel with a stationary threshold control policy is stable; otherwise it is unstable. The proof is given in Appendix A.

The threshold control policy may be quite difficult to implement in practice. It has a major advantage, however, with respect to the retransmission control policies we shall study in section 3.2: the maximum achievable channel throughput is *not* limited to e^{-1}. In fact, the throughput may be arbitrarily close to one if $\Theta = 1$ since it suffices to set $f = 1$ in this case. In general, for $\Theta \geq 1$, A is maximized by setting f equal to $f^* = \Theta^{-1}$. We then have $A(f^*) = (1 - \Theta^{-1})^{\Theta-1}$ which, for $\Theta >> 1$ is $A(f^*) \simeq \exp(-1 + \Theta^{-1}) > e^{-1}$. We see here that e^{-1} is a *lower bound* to the maximum achievable throughput. This does not depend on the Poisson assumption of packet arrivals to the channel.

3.2 A Retransmission Control Policy

A retransmission control policy is one which regulates access to the channel from the set of blocked terminals as a function of the past and present state of the system. We consider such a policy which only uses information concerning the present state (it is stationary) to regulate the retransmission rate of the ensemble of blocked terminals. The appropriate definition of stability (for this case) is then that given in Definition 1, and the equations for the controlled system are Equation (6) for the infinite source model and Equation (7) for the finite source model with the following modification. The parameter f which determines $g_i(n)$ (see Eqs. (3) and (4)), given the probability that a blocked terminal retransmits a packet during a slot, will be a function of n which we denote f(n) so that,

$$g_i(n) = \binom{n}{i}[f(n)]^i[1 - f(n)]^{n-i} \quad . \tag{21}$$

The following result can then be established.

Theorem 4.

A stationary retransmission control policy yields a stable broadcast channel if,

$$\lambda = \sum_{i=1}^{\infty} ic_i < d \quad , \tag{22}$$

and an unstable one if $\lambda > d$, where $d = \lim_{n\to\infty}[c_1g_0(n) + c_0g_1(n)]$.

The proof of this result is given in Appendix B. We do not have a proof of instability for $\lambda = d$ except for a special case; however, the question is only of mathematical interest.

Remark

In fact, Theorem 1 is a corollary of Theorem 4 since, if (as in the case for the uncontrolled broadcast channel) f is independent of n, we have $d = 0$.

Another consequence of Theorem 4 concerns the form which the function $f(n)$ must take to ensure stability.

Theorem 5

For the broadcast channel under stationary retransmission control to be stable, it is necessary that,

$$\lim_{n\to\infty} f(n) = 0 \quad \text{and} \quad \lim_{n\to\infty} nf(n) > 0 \quad . \tag{23}$$

Proof

Clearly, if the first condition is not satisfied, we shall have $d = 0$ leading to the instability of the channel. Now suppose that the second condition is not satisfied, that is $\lim_{n\to\infty} nf(n) = 0$, but that the first condition is satisfied. Then $d = c_1$, and we cannot have $\lambda < d$; therefore, by Theorem 4 the system will be unstable, which completes the proof.

We see, therefore, by this last result that a stationary retransmission control policy (with expected time between attempts of a blocked terminal to retransmit given by $[f(n)]^{-1}$, may stabilize the channel only if $f(n)$ decreases with n but no faster then the function n^{-1}.

3.3 An Optimal Retransmission Control Policy

It is natural to seek retransmission control policies which will maximize the output rate of the channel; for a stabilizing policy, the maximum value will be d of Theorem 4 since the input rate will be identical to the output rate. Consider,

$$D_n(f) = c_1(1 - f)^n + c_0 nf(1 - f)^{n-1} \quad . \tag{24}$$

By deriving this expression with respect to f and setting the result equal to zero, we see that $D_n(f)$ is maximized by setting f equal to $f^* = (c_0 - c_1)(nc_0 - c_1)^{-1}$ for $n \geq 1$, or $f^* = (1-\alpha)(n-\alpha)^{-1}$ if $\alpha = c_1/c_0$, where we are restricted to $\alpha < 1$ (for instance, with a Poisson arrival process $\alpha = \lambda$). The maximum value of $D_n(f)$ is then $D_n(f^*) = c_0[(n-1)(n-\alpha)^{-1}]^{n-1}$. In the limit as $n \to \infty$, we shall obtain the throughput $d = \exp(\log c_0 + \alpha - 1)$. If the arrival process is Poisson, we obtain $d = e^{-1}$ as predicted by Abramson [1] and Kleinrock and Lam [10] for the maximum throughput of the channel.

In Figure 3, we present time series characterizing channel behavior obtained by Monte-Carlo simulation with a Poisson arrival process of packets from active terminals. In Figure 3a, we show the behavior of the uncontrolled broadcast channel; we see that if the number of blocked terminals is sufficiently high, the channel is unable to recover (i.e., it is unstable), and the total number of blocked terminals increases indefinitely while the channel throughput tends to zero. In Figure 3b, we see the channel behavior under identical conditions, except for the retransmission probability which is chosen to be f^*. The channel is now able to recover from an initial state with a large number of blocked terminals, and the throughput matches the input rate. The exact form of f chosen in the simulation results of Figure 3b is $f^+ = (1-\lambda)n^{-1}$, where denominator term of f^* has been simplified. There is a simple intuitive (but nonrigorous) explanation for the choice of f^+: when there are n-blocked terminals and n is very large, the set of blocked terminals will behave as a Poisson source of parameter $f^+ \cdot n = 1 - \lambda$; thus, the total input rate of packets to the channel will be $\lambda + f^+ n = 1$, which is the maximum rate it can admit.

The control policy f^* could be approximately implemented by simple statistical estimation of the number of blocked terminals. The estimate could be obtained by a specialized terminal (or by the data concentrator which receives packets and sends back the acknowledgement packets) which would deduce an instantaneous estimate of the number of blocked terminals by measuring the throughput. It would then send once in a

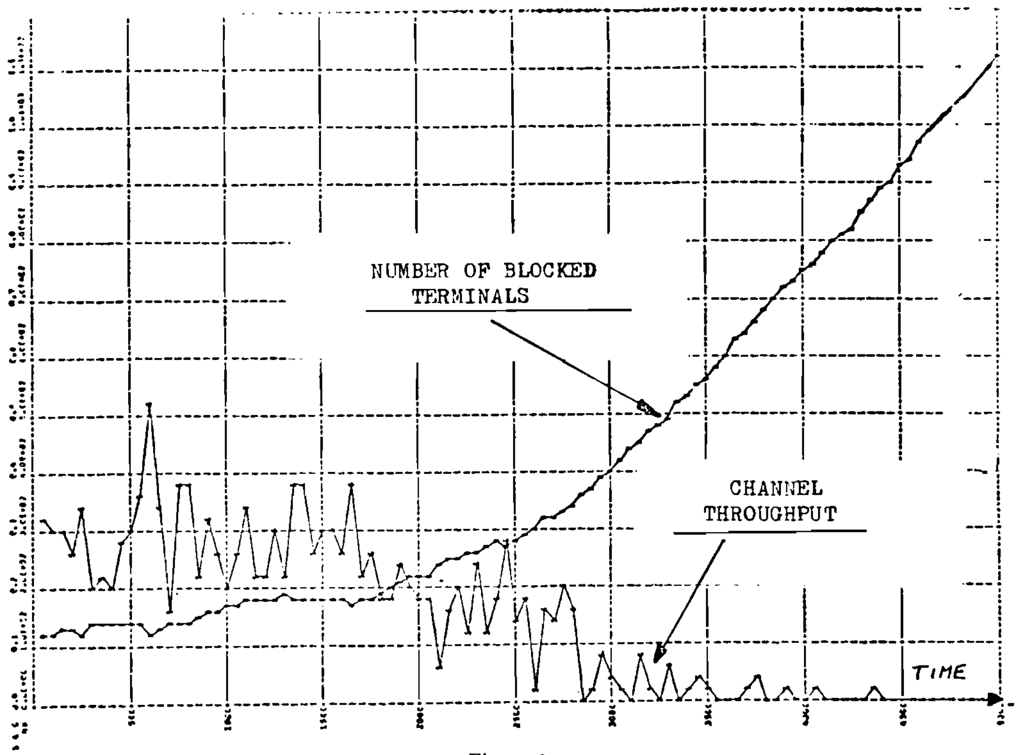


Figure 3a.

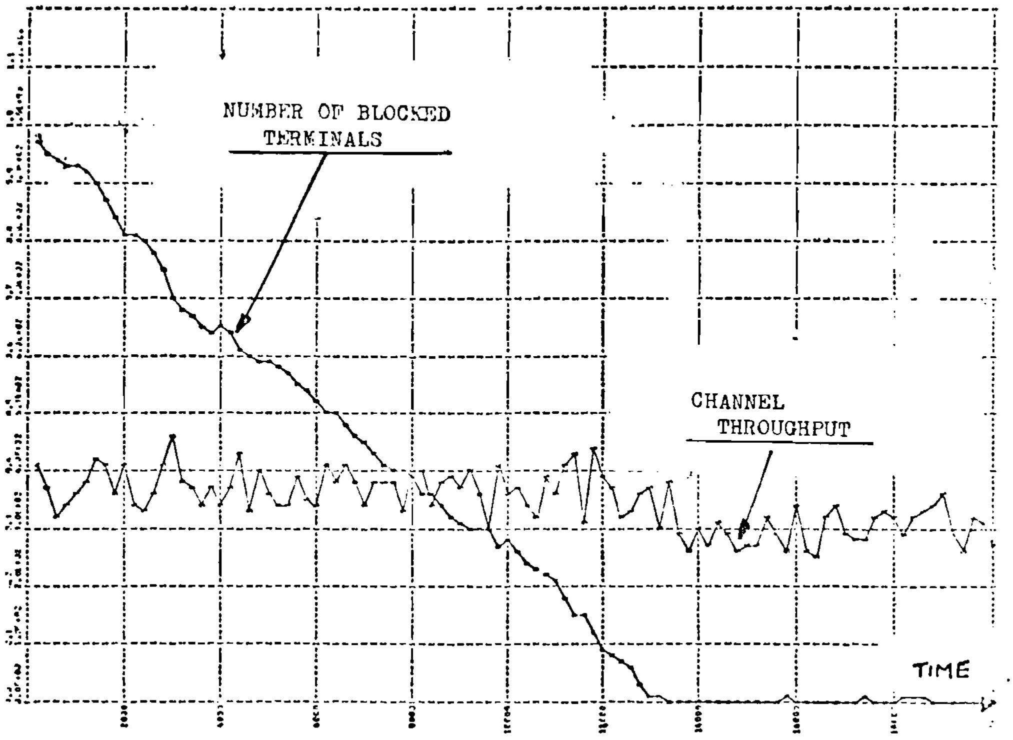


Figure 3b.

by measuring the throughput. It would then send once in a while an updated value of f^* on the frequency used for acknowledgement packets.

4. CONCLUSIONS

In this paper, we have given a theoretical treatment of some basic problems related to the packet-switching broadcast channel. Its inherent instability has motivated us to look into stabilizing control policies. The first policy examined has been one in which access to the channel is controlled by admitting active terminals which wish to transmit a packet into an impeded set. Necessary and sufficient conditions under which the number of impeded plus blocked terminals remains bounded are derived, and it is shown that, with this policy, it is theoretically possible to achieve a throughput which is arbitrarily close to one.

We have then examined control schemes based only on choosing the transmission probability of any blocked terminal as a function of the total number of blocked terminals. Sufficient conditions for stability and instability of the channel and necessary conditions which must be satisfied by the retransmission probability are derived for this scheme. We then obtain the optimal control policy for the channel which maximizes the throughput. This policy appears promising as a practical means of optimizing channel performance.

Acknowledgements:

The authors are grateful to Prof. L. Kleinrock for discussions which motivated this research.

References

[1] Abramson, N., The ALOHA System-Another Alternative for Computer Communications, Fall Joint Computer Conference, *AFIPS Conf. Proc.*, 37 (1972), 281-285.

[2] Abramson, N., Packet Switching with Satellites, National Computer Conference, *AFIPS Conf. Proc.* (1973), 695-702.

[3] Baran, P., et al., On Distributed Communications, Series of eleven reports, Santa Monica, California, Rand Corp., 1964.

[4] Cohen, J., *The Single Server Queue*, Amsterdam, North-Holland, 1969.

[5] Davies, D.W., The Principles of a Data Communication Network for Computers and Remote Peripherals, *Proc. IFIP Congress* (Hardware), Edinburgh, 1968.

[6] Fayolle, G., et al., The Stability Problem of Broadcast Packet Switching Computer Networks, forthcoming in *Acta Informatica*.

[7] Howard, R.A., *Dynamic Probabilistic Systems*, 2, New York, John Wiley, 1971.

[8] Kleinrock, L., Resource Allocation in Computer Systems and Computer-Communication Networks, *Proc. IFIP Congress*, Stockholm, 1974. Rosenfeld, J.L. (ed.), Information Processing 1974, Amsterdam, North Holland, 1974.

[9] Kleinrock, L., and S. Lam, On Stability of Packet Switching in a Random Multi-Access Broadcast Channel.

[10] Kleinrock, L., and S. Lam, Packet Switching in a Slotted Satellite Channel, National Computer Conference, *AFIP Conf. Proc.* (1975), 703-710.

[11] Lam, S., Ph. D. Thesis, Computer Science Dept., University of California, Los Angeles, 1974.

[12] Pouzin, L., Presentation and Major Design Aspects of the Cyclades Computer Network, Third Data Communications Symposium, St. Petersburg, Florida, November 1973.

[13] Roberts, L.G., Multiple Computer Networks and Inter-Computer Communications, ACM Symposium on Operating Systems. Gatlinburg, Tenn., 1967.

[14] Roberts, L.G., Dynamic Allocation of Satellite Capacity Through Packet Reservation, National Computer Conference, *AFIPS Conf. Proc.*, New York City, June 1973, 711-716.

APPENDIX A

Proof of Theorem 3

The theorem is easily established for $\Theta = 1$. It suffices to notice that in this case the system is equivalent to a single-server queue with binomial service (with parameter $A = f$) and mean service time $1/f$; the arrival process is independent in each service interval. It is easily shown that the model has an equilibrium distribution of queue length (corresponding to the numbers of impeded terminals) if and only if $\lambda < A$. Now consider the case $\Theta > 1$. Let π_n denote the equilibrium probability that the number of impeded terminals is n for $n \geq 1$; q_j will denote the equilibrium probability that there are zero impeded terminals and j blocked ones. The equilibrium probabilities satisfy for $n \geq 1$,

$$\begin{aligned}\pi_n = \pi_{n+1}C_\Theta A + \sum_{i=0}^{n-1} \pi_{n-i}[C_i(1-A) + C_{i+1}A] \\ + \sum_{j=0}^{\Theta} q_j C_{n-j+\Theta} - q_\Theta C_{n+1}A \ .\end{aligned} \tag{25}$$

Define the generating function $G(x) - \sum_{n=1}^{\infty} \pi_n x^n$. Then,

$$\begin{aligned}G(x) = \frac{C_0 A}{x}(G(x) - \pi_1) + G(x)[G(x)(1-A) + \frac{A}{x}(C(x) - C_0)] \\ + \phi(x) - q_\Theta A \frac{C(x) - C_0}{x} \ ,\end{aligned} \tag{26}$$

where,

$$\phi(x) = \sum_{n=1}^{\infty} \sum_{j=0}^{\Theta} q_j C_{n-j+\Theta} x^n \ , \tag{27}$$

yielding,

$$C(x) = \frac{x\phi(x) - q_\Theta A(C(x) - C_0) - C_0 A\pi_1}{x - xC(x)(1 - A) - AC(x)} \quad . \tag{28}$$

Notice that $\pi_n = P_{\Theta+n}$, $n \geq 1$, $q_j = P_j$, $0 \leq j \leq \Theta$, of Eqs. (19) and (20). By Foster's theorem (4), the Markov chain representing the number of blocked plus impeded terminals will be ergodic (and the channel will be stable) if there exists a positive solution to Eq. (25), of finite sum since the Markov chain is irreducible and aperiodic.

Suppose $q_0 > 0$; it can be easily shown that $q_j > 0$, $1 \leq j \leq \Theta$, and $\pi_n > 0$, $n \geq 1$. Denote by F(x) the numerator of Eq. (28); we first show that if $q_0 > 0$, then $F'(1) > 0$. We have,

$$F'(1) = \phi(1) + \phi'(1) - q_\Theta A \sum_{n-1}^{\infty} nC_n$$

$$= \sum_{n=1}^{\infty} \sum_{j=0}^{\Theta} q_j C_{n-j+\Theta} (n+1) - Aq_\Theta C_n n) > 0 \quad , \tag{29}$$

since we have $A \leq 1$.

Now take $\lim_{x\to 1} G(x)$. After applying l'Hôpital's rule, we remain with,

$$\lim_{x\to 1} G(x) = \frac{F'(1)}{A - C'(1)} = \frac{F'(1)}{A - \lambda} \quad . \tag{30}$$

Clearly, if $A = \lambda$ and $q_0 > 0$, then G(1) does not exist and the channel is unstable. Similarly, if $A > \lambda$ and $q_C > 0$, then $G(1) < 0$ which is a contradiction so that again the channel is unstable. The case $\lambda < A$, however, remains to be considered; taking any finite $q_0 > 0$, we see that since $G(1) < \infty$ due to the fact that F'(1) is bounded, the sum,

$$G(1) + \sum_{j=0}^{\Theta} q_j < \infty \quad , \tag{31}$$

and Foster's theorem is satisfied. Therefore if $\lambda < A$, the broadcast channel with the threshold control policy is stable.

APPENDIX B

Proof of Theorem 4

Let us first determine that the channel is unstable if $\lambda > d$. If the limit defining d exists, then for each $\varepsilon > 0$ there exists an integer n_0 such that for all $n \geq n_0$,

$$|g_1(n) - a| \leq \varepsilon \text{ and } |g_0(n) - b| \leq \varepsilon \quad , \tag{32}$$

where a, b are constants such that,

$$d = c_0 a + c_1 b \quad . \tag{33}$$

Let $P(z) = \sum_{n=n_0}^{\infty} p_n z^n$, $Q(z) = \sum_{n=n_0}^{\infty} S_n z^n$. Then, from Eq. (9) and the discussion above, we have,

$$S_n - \sum_{j=0}^{n} S_{n-j} c_j \leq (a+\varepsilon) p_{n+1} c_0 + (b+\varepsilon) p_n c_1 \quad , \tag{34}$$

and,

$$S_n - \sum_{j=0}^{n} S_{n-j} c_j \geq (a-\varepsilon) p_{n+1} c_0 + (b-\varepsilon) p_n c_1 \quad , \tag{35}$$

for all $n \geq n_0$. Thus from Eq. (34), we derive,

$$Q(z) - \sum_{n=n_0}^{\infty} \sum_{j=0}^{n} S_{n-j} c_j z^n \leq \frac{(a+\varepsilon)_0}{z} (P(z) - z^{n_0} p_{n_0} + (b+\varepsilon) c_1 P(z)) \quad . \tag{36}$$

Notice that,

$$\sum_{n=n_0}^{\infty} \sum_{j=0}^{n} s_{n-j} c_j z^n = \sum_{n=n_0}^{\infty} \sum_{j=0}^{n-n_0} s_{n-j} c_j z^n + \sum_{n=n_0}^{\infty} \sum_{j=n-n_{0+1}}^{n} s_{n-j} c_j z^n \quad . \tag{37}$$

Therefore, if we denote $C(z) = \sum_{j=0}^{\infty} c_j z^j$,

$$Q(z)(1 - C(z)) \leq \frac{(a+\varepsilon)c_0}{z} (P(z) - z^{n_0} p_{n_0} + (b+\varepsilon)c_1 P(z)) + \sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} s_{n-j} c_j z^n \quad . \tag{38}$$

The following relationship may be verified:

$$Q(z)(1-z) = z^{n_0} s_{n_0} + P(z) - z^{n_0} p_{n_0} = P(z) + z^{n_0} s_{n_0 - 1} \quad , \tag{39}$$

yielding after substitution in Eq. (38) and combining terms:

$$P(z)\left[\frac{1 - C(z)}{1 - z} - \frac{(a+\varepsilon)c_0}{z} - (b+\varepsilon)c_1\right] \leq - z^{n_0}\left(\frac{1 - C(z)}{1 - z}\right) s_{n_0 - 1} - z^{n_0 - 1}(a+\varepsilon)c_0 p_{n_0} + \sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} s_{n-j} c_j z^n \quad . \tag{40}$$

However,

$$\sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} s_{n-j} c_j z^n \leq s_{n_0-1} \sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} c_j z^n \quad , \tag{41}$$

and,

$$\sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} c_j z^n \leq \sum_{n=1}^{\infty} \sum_{j=n}^{\infty} c_j z^n = F(z) \quad , \tag{42}$$

where $\lim z \to 1 \; F(z) = \lambda$.

Returning to Eq. (40), we obtain,

$$P(z)\left[\frac{1-C(z)}{1-z} - \frac{(a+\varepsilon)c_0}{z} - (b+\varepsilon)c_1\right]$$

$$\leq S_{n_0-1}\left[-z^{n_0}\left(\frac{1-C(z)}{1-z}\right) + \sum_{n=1}^{\infty}\sum_{j=n}^{\infty} c_j z^n\right]$$

$$- z^{n_0-1}(a+\varepsilon)c_0 p_{n_0} \quad . \tag{43}$$

Now we take the limit as $z \to 1$ of both sides in Eq. (43). We obtain,

$$P(1)[\lambda - (a+\varepsilon)c_0 - (b+\varepsilon)c_1] \leq - c_0 p_{n_0}(a+\varepsilon) \quad . \tag{44}$$

Therefore, if $\lambda > d$, choosing n_0 sufficiently large so that $\lambda - d > \varepsilon(c_0+c_1)$, we have that either $p_{n_0} = 0$ and $P(1) \leq 0$ or $p_{n_0} > 0$ and $P(1) < 0$; in both cases it implies that the balance equations satisfied by the equilibrium probability distribution $\{p_n\}$ do not possess a positive solution. Thus the Markov chain representing the number of blocked terminals at the beginning of each slot is not ergodic and the channel is unstable if $\lambda > d$.

Starting with Eq. (35) and proceeding by arguments similar to the ones used above, we can obtain,

$$P(1)[\lambda - (a-\varepsilon)c_0 - (b-\varepsilon)c_1] \geq -\lambda S_{n_0-1} - (a-\varepsilon)c_0 p_{n_0} \quad .$$

$$+ \sum_{n=n_0}^{\infty}\sum_{j=n-n_0+1}^{n} S_{n-j}c_j \quad . \tag{45}$$

The last term on the right-hand side of Eq. (45) cannot exceed λS_{n-1}, therefore, assuming p_{n_0} is positive, we may write,

$$P(1)[\lambda - (a-\varepsilon)c_0 - (b-\varepsilon)c_1] \geq - \alpha(n_0) \quad , \tag{46}$$

where,

$$0 < \alpha(n_0) = \lambda S_{n_0-1} + (a-\varepsilon)c_0 P_{n_0} - \sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} S_{n-j}c_j \quad , \tag{47}$$

since, by choosing n_0 sufficiently large, we know that $a > \varepsilon$. Therefore, if $\lambda < d$, then,

$$P(1) \leq \frac{a(n_0)}{d - \lambda - \varepsilon(c_0 + c_1)} \quad , \tag{48}$$

assuming that n_0 is large enough so that $d - \lambda > \varepsilon(c_0 + c_1)$. From Eq. (7), we notice that we may write for any $n \geq 0$,

$$p_n = k(n)p_0 \quad , \tag{49}$$

where $k(n) > 0$; thus,

$$P(1) \leq \frac{\lambda \sum_{j=0}^{n_0-1} k(j) + (a-\varepsilon)c_0 k(n_0)}{d - \lambda - \varepsilon(c_0 + c_1)} \quad . \tag{50}$$

Notice that p_{n_0} is positive if and only if p_0 is positive. We can now invoke Foster's theorem (4) which implies that the Markov chain is ergodic if there exists a positive solution to the equilibrium Eq. (7) such that $P(1) < \infty$. Setting $p_0 = 1$ (or any positive constant), Eq. (49) represents a positive solution of Eq. (7); by Eq. (50), we will have $P(1) < \infty$; therefore, we have satisfied Foster's condition completing the proof that the channel is stable if $\lambda < d$. We now have to consider the case $\lambda = d$.

For $n \geq n_0$, we may write,

$$g_1(n) = a + u_n \quad , \quad g_0(n) = b + v_n \quad , \tag{51}$$

so that from Eq. (9) we obtain,

$$Q(z)[1 - C(z)] = \frac{ac_0}{z}[P(z) - z^{n_0}p_{n_0}] + bc_1P(z)$$

$$+ \frac{c_0}{z}[U(z) - z^{n_0}u_{n_0}\,P_{n_0}] \tag{52}$$

$$+ c_1V(z) + \sum_{n=n_0}^{\infty} \sum_{j=n-n_0+1}^{n} s_{n-j}c_jz^n \quad ,$$

where,

$$U(z) = \sum_{n=n_0}^{\infty} u_np_nz^n \quad , \quad V(z) = \sum_{n=n_0}^{\infty} v_np_nz^n \quad , \tag{53}$$

yielding,

$$P(z) =$$

$$\frac{-z^{n_0}s_{n_0-1}\frac{1-C(z)}{1-z} - C_0(a+u_{n_0})p_{n_0}z^{n_0-1} \leftarrow \frac{C_0}{z}U(z) + C_1V(z) \sum_{n=n_0}^{\infty} \sum_{j=n=n_0+1}^{n} s_{n-j}c_jz^n}{\frac{1-C(z)}{1-z} - \frac{aC_0}{z} - bc_1} \quad . \tag{54}$$

For $\lambda = d$, the denominator of $P(1)$ vanishes. Instability for $\lambda = d$ will be verified if we can show that the numerator of $P(1)$ does not vanish or that the numerator of $P(z)$ tends to zero more slowly than the denominator for $\lambda = d$ as $z \to 1$. If $c_0g_1(n) + c_1g_0(n) < d$ for all $n \geq n_0$ (i.e., if D_n tends to d from below), then $c_0U(1) + c_1V(1) < 0$, and clearly the numerator of $P(1)$ is negative for $p_n > 0$, and $P(1)$ does not exist. Under this condition, the channel is unstable for $\lambda = d$. In general, however, even though we conjecture that the channel is unstable when $\lambda = d$, we have no proof of this.

IIASA Data Communication Network

A. Butrimenko, J.H. Sexton, and V. Dashko

1. BRIEF DESCRIPTION OF PLAN

In March of this year, a formal proposal to construct a data communication packet-switching network linking IIASA to computer centers in National Member Organizations (NMO's) was circulated. This step had been preceded by long range preliminary contacts with various projects in this institute and some computer centers in the NMO's. Circulation of this paper was also a demonstration of our *belief* that the communication between scientists involved in in-house research, and their counterparts in NMO's is essential to the success of this unique institution.

The initial plan is to connect the following centers: IIASA; The Computer and Automation Institute, Budapest; The Computing Research Center, Bratislava; The Institute of Control Science, Moscow; The Cybernetics Institute, Kiev; and the Technical University, Vienna. Some schemes for doing this are shown in Figure 1.

Bulgarian and Polish Academies of Sciences have also expressed their willingness to join the IIASA network and provide needed computer facilities and man power. Exploratory contacts have been made with EIN and UN authorities on possible cooperation.

2. PURPOSE

In order to understand IIASA's interest in data communications, it is necessary to know something of how the institute functions. This institute is characterized by a large variety of research fields and disciplines. We have here 11 projects in which the scientists of 14 National Member Organizations are involved. The institute relies inevitably on the support and intellectual supply from the national institutions. Scientists come to IIASA for periods of a few weeks to several months.

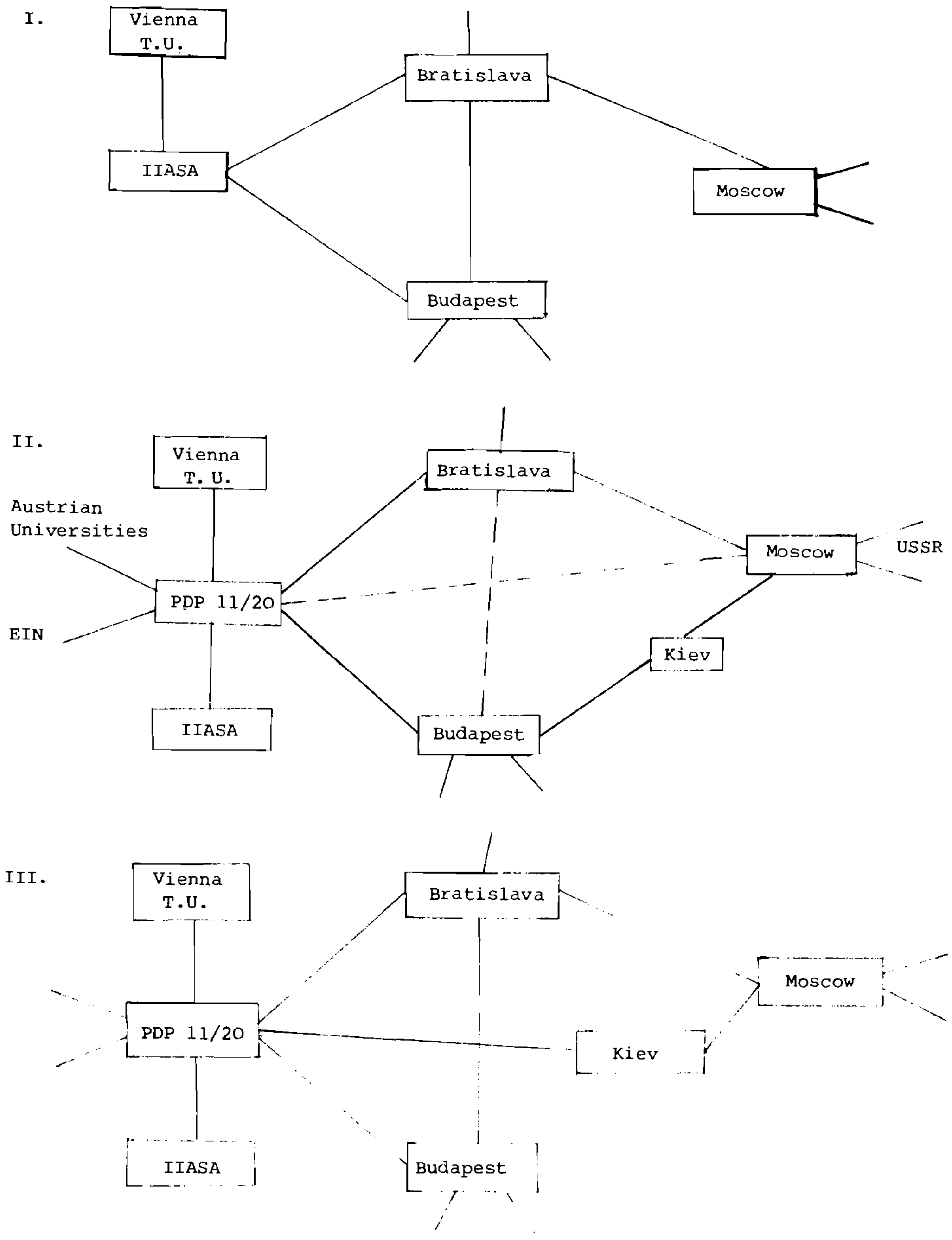


Figure 1. Some possible alternative configurations for the IIASA network.

This continuing inflow and outflow leads to working relationships being established between IIASA and the national institutions providing these scientists. As a result, IIASA is in a unique position to stimulate cooperative research, but joint projects require adequate communication facilities. Because of the computer intensive nature of most of our projects, all of which involve the applications of systems analysis to problems of global interest, we firmly believe that data communication forms an important part of such facilities. We also firmly believe that improved communication facilities will both stimulate international research and also provide a basis for collaborative research of a type which is practically impossible with the small number of people that can be brought together at this institute.

At the same time, international computer networking is a developing problem of global interest and therefore appropriate for study at the institute. In brief, the goals of the proposed IIASA network are:

a) to provide data communication links between IIASA and cooperating national and international institutions;

b) to provide access for IIASA scientists to specialist data base and computing facilities, particularly those at the home institutions;

c) to provide a practical base for research in computer internetworking.

In a later section of this paper, we shall draw your attention to some topics of basic research conducted by this project.

3. THE GENERAL FRAMEWORK

When considering the implementation of our network, we have to take into account a number of constraints and limitations. We also have to bear in mind how these will change with time and how our plans may develop.

We give here some details concerning hardware, topology, type, etc. There are some very heavy constraints imposed on the development of the IIASA network. Because of the nature of this project, we cannot expect an entirely homogeneous network even at the level of switching nodes. There have been and partly remain some differences in understanding the goals and scope of the project between people from various computer centers, and time has been needed to reach an understanding and achieve a common terminology.

At IIASA, we have one in-house machine, a PDP 11/45 which we run in time-sharing mode using the Bell Labs UNIX operating system. There are currently eight terminals attached. Because of the heavy demand on this machine as a general purpose, computing resource, it is planned not to include packet-switching functions but rather to attach it to our network as a host. At Budapest, a TPA 70 minicomputer will be used as a node (packet switch), and similar arrangements are proposed for Bratislava using a NORD 20 and in Moscow using a PDP 11/20. In these three cases, it is also proposed to attach some local general-purpose computers as hosts. These include an ICL system 4 and a Siemens 4004 in Moscow and large R series machines in Budapest.

One installation of special interest is the Cyber 74 of the Vienna Technical University, which has especially good communication facilities built in. From IIASA, we have been using this machine for some time, but now we want to include it directly in our network. One possible solution to this is to introduce another node, perhaps a PDP 11/20 between IIASA and Vienna, to which each site could attach its hosts. Such an arrangement is particularly attractive since it would permit the connection of other centers in Austria and also (something of great importance to us) would facilitate connection to other networks in Western Europe, such as EIN.

4. LINKS

For the links, we intend to use the telephone service. We are starting with dialed-up lines and are trying speeds of 600 and 1200 bits/sec, but, as soon as possible, we shall upgrade these connections by changing to leased lines and speeds of 2400 bits/sec. For higher speeds, conditioned lines will be required. The primary constraint in this particular is the question of cost, and a need to justify new expenditure by demonstrating feasibility. Furthermore, we do not expect very heavy usage of the computer network in the early stages. The initial goal is to provide the physical capability to establish the needed connection on demand for relatively short periods of time, and not to run this connection permanently.

For the same reason, it has been agreed to operate the links with Bratislava and Moscow using asynchronous MODEMs, but it is hoped that some of the links will be operated in synchronous mode, and the line protocol has been designed with this in mind.

5. PROTOCOLS AND SOFTWARE

It is proposed to take a fairly conventional approach to the communication protocols, at least at the lowest levels. That is to say software will be written to form a series of levels. These are: line protocol, packet protocol, and host-to-host protocol. However, we intend to rationalize this approach as much as possible and avoid duplication or provision of little used facilities. We feel strongly that there is a real need to reduce the communication software to a small volume and avoid excessive use of precious core store.

6. LANGUAGE AND OPERATING SYSTEMS

We are making a study of necessary operating system features and available systems programing languages. On the IIASA PDP 11, we use a language called "C" which is almost identical with BCPL. A language which is available to all centers is PL 11. We believe that by being careful about programing methodology, we can make our software better, smaller, and faster to implement. At IIASA, we are constrained to work within the framework of UNIX, but at the other centers, especially nodal centers, we expect to have to build an operating system. We intend to make it as small as possible, consistent with provision of basic essentials. Some desirable features are:

a) parallel processing,

b) inter-process communication,

c) process priority scheduling system,

d) automatic restart.

We intend to build a modular system and, wherever possible, to aim for transportability, so that modules developed at one center can be distributed and used at others. Systematic documentation will form an important part of this.

7. DATA LINK PROTOCOL

We have made a survey of existing and developing protocols in this area and are dismayed to see how many separate developments there are. It seems that every new entry to the networking field develops its own line protocol. At IIASA, we have decided to forgo this pleasure and to adopt the emerging European standard HDLC. Having said that, we have to admit that it is not quite what we need, and we have had to tinker with it a little. Firstly, it is intended for synchronous communication, and some of our lines will be operated asynchronously. This, however, is no problem, and we simply intend to construct a frame for synchronous transmission and allow the interface to segment this and add the start and stop bits. In this way, we avoid a distinction between these modes of operation and also avoid having to alter our software for synchronous communication at a later date.

For us, a more serious problem arises because HDLC employs a bit stuffing technique, which would be very inefficient to do by software. In the absence of hardware, we have, therefore, been forced to rethink this and have decided to adopt the HDLC frame structure but to employ the oder "synchronization character, data-link-escape" technique. Our frames will, therefore, have the structure shown in Figure 2.

SYN	SYN	ADDRESS	CONTROL	INFORMATION FIELD	CRC	PAD
1 byte	1 byte	1 byte	1 byte	0 or more bytes	2 bytes	1 byte

Figure 2.

When it exists, the information field will be terminated by ETB. To achieve transparency and avoid premature termination due to the accidental occurrence of ETB in the data, all occurrences of SYN, DLE, and ETB in the data will be immediately preceded by DLE.

From here on, we adapt HDLC without change. However, it is clear that we only need a subset, and we have decided on the following choice of features.

8. HDLC Subset

We shall use only normal response mode in half-duplex. Supervisory and unnumbered commands will be limited to the following set: RR, RNR, SNRM, DISC; and supervisory and unnumbered responses to: RR, RNR, UA, CMDR.

We shall not use the extended modes. In agreement with ISO standard DIS 3309, we shall use the address field to identify the secondary, but we shall divide that field into two subfields as shown in Figure 3.

4 bits	4 bits
Center Number	Secondary Number

$\longleftarrow$ Address byte $\longrightarrow$

Table of Center Numbers

Center	Number
IIASA	1
Bratislava	2
Budapest	3
Kiev	4
Moscow	5
Vienna	6

Figure 3.

If FILL characters are transmitted between frames to maintain an active channel state, that character will be SYN.

To overcome the unsatisfactory asymmetry between primary and secondary stations, we propose the following solution. On each link there will be two logical channels with a primary at each end (see Figure 4).

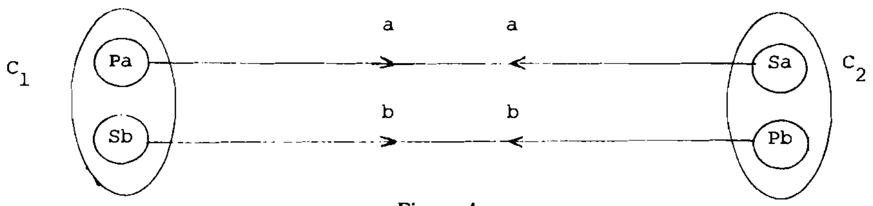


Figure 4.

Primary Pa and secondary Sa will communicate with one another using the center number for C_2, while primary Pb and secondary Sb will communicate using the center number for C_1. The secondary numbers for such a pair will be the same. The send and receive sequence numbers will be distinct for each channel in accordance with the standard. In order to avoid confusion between the traffic on the two logical channels, the primary and secondary at a given center will cooperate according to the following rule, which applies to the state where both channels have been initialized in normal mode.

If there is information to be sent, then it will be sent on the channel last used, unless the line has been quiescent for a certain time (a constant yet to be fixed). When the line has been quiescent for this time, a station wishing to restart activity will do so by transmitting from its primary, which may or may not be the same logical channel as before. The clash which can occur in this quiescent situation will be resolved by having a different time out for each primary.

Cross links are easily detected in this arrangement, but, for the purpose of local testing, cross linking could be allowed by forcing the channel numbers to be equal.

Initialization, disconnection, and recovery from a CMDR will be symmetrical, but each channel can, in principle, be initialized and disconnected separately so that at times (and in some cases) there will only be one available channel (or none), and in such situations one station will function solely as the primary and the other as the secondary. This will allow symmetrical testing and progressive activation of a communication link.

In deciding on the above implementation of HDLC, we considered adopting the SESA-LOGICA solution for the European Information Network. However, we abandoned it because it is not in accord with HDLC in that it does not use either the address or control fields and because it does not represent any real advantage. The argument that using eight logical channels makes better use of a half-duplex protocol is just false since each cycle is reduced to two frames, rather than the cycle of eight available with the three bit HDLC sequence numbers. In either case, at most eight frames can be transmitted before an acknowledgement is received. Whether one should have several logical channels or not is entirely irrelevant at the line protocol level and is an example of the needless complication in communication protocols referred to earlier.

9. PACKET SWITCHING

The description of the line protocol above is rather extensive because this is the area where we have, so far, spent most of our time studying, but we are also beginning to think about the packet structure and protocol and host-host protocols. All that can be positively stated at this stage is that we shall probably use the "D" format packet header as proposed by Pouzin and packets of 0-255 data bytes. Uniformity in computer networking standards is something which we feel to be important, and this is one reason for preferring the D format, but another is that it is one of the few that provide adequate address space for inter-network communication. As has already been stated, our network cannot be entirely homogeneous, and it is quite probable that we shall have to connect to a number of existing networks. Indeed, we plan to do this in the case of some existing networks in Western Europe.

10. HOST-HOST AND HIGHER-LEVEL PROTOCOLS

Our present timetable includes the development of a host-host protocol, but, unlike the lower levels, we are at last faced with contact with the end users. Since we plan to participate in internetworking, it is clear that we cannot develop such a protocol in isolation. We may then consider ourselves obliged to adopt an international standard if one then exists, or whatever is being widely used on the networks that we wish to connect to. It might be feasible to adopt a subset or to ensure a sufficient intersection, but it is clearly highly desirable to support standardization in this area, while at the same time keeping enough room for experiment, bearing in mind our goal of reducing the volume of software needed to support communication.

As to higher-level protocols, we expect to participate in the studies that are currently being made and hope to contribute new ideas especially where they assist in standardization and the widening of useful resources available over a network.

11. DEVELOPMENT TIMETABLE

A program of work has been drawn up for the period to the end of August 1976. It includes the development and implementation of communication protocols to the level of host-host communication between at least two centers. We have tried to distribute the development between the centers to avoid duplication as much as possible and to exploit the different areas of expertise available at each center. Inevitably though, some duplication cannot be avoided and is even desirable in order that each center should understand the complete system. In particular, it would be wrong to maintain a complete dependence on IIASA since staff members come and go, and the same is true for IIASA projects. Further, it is very desirable that centers should establish working projects with one another, and, in order to realize one of our principal goals, it is necessary for national centers to continue the growth of the network by connections to research institutes in their own countries. At the same time, IIASA will continue to take the initiative, and we have recently proposed the formation of a Network Management Committee to unite and guide the growth of the network.

The timetable calls for the implementation and testing of the line protocol to be completed between IIASA and one or two centers by the end of January 1976. To this end, the Czechoslovak center is proposing to bring its NORD 20 to IIASA from mid-November. This will materially assist the human communication and the solution of the development problems which inevitably occur at this stage of a network project. The experience which is gained at IIASA of its own limitations and local problems will make the connection to other centers a much smoother process.

12. RELATION TO OTHER NETWORKS

As has been mentioned above, we are very interested in the possibilities of connecting to other research networks. For some time, we have been exploring the available opportunities and are considering connections to EIN, RPC in Italy, and CYCLADES. That is not to say that we shall connect to all three with direct links, but are looking for ways of connecting to centers on these networks, especially those which are doing work of relevance to IIASA projects.

We are currently engaged in a joint feasibility study with Austria to consider Austrian participation in international networking. One possible outcome of this study could be that Austria would decide to join EIN. In that case, of course, we should link to the Austrian node.

One goal that we have had for some time is to obtain a connection to the World Health Organization and the International Computing Center in Geneva. Unfortunately, there are no plans at the moment for a computer network in Switzerland, so our only route would appear to be via EIN and CYCLADES. Such a connection would lead to some interesting problems in gateway development.

13. EXPLOITATION

The uses that we shall make of our networking capability remain to be more fully explored, but IIASA is in a unique position to function as a real user of international networks, in the sense that our research projects have a genuine need to get at sources of data and remote computing resources. This arises just because our work is of an international character and our staff have links with home institutions and need to be able to access their home computers and data-bases which they know well. In the short periods that some scientists spend here, it is unrealistic to expect them to learn a whole new operating system, languages, and system peculiarities. In any event, their files and data are at home and not at IIASA.

It follows that IIASA has an important contribution to make to international computer networking especially in the scientific research area. It is able to provide existing and developing networks with grist for their mill, where previously they have only been used as tools for research upon themselves. It is probably time for the individual networks projects, at least throughout Europe, to unite their efforts with potential users to establish international networking as a tool for data communication in scientific research.

14. RESEARCH AREAS

While we view our task primarily to be that of the provision of a needed facility, it is clear that many interesting questions will arise in the course of our work, and the network can function as a vehicle for some experiments in data-communication. The areas which are of especial interest to us include:

a) accounting and network performance,

b) access to heterogeneous information systems,

c) the problems of small systems and compatibility of existing operating systems with data communication demands,

d) the provision of network services,

e) the special problems of internetworking.

Concerning fundamental research, a study is under way on adaptive job allocation in large computer networks. The methods are aimed at combining in one process job allocation, routing, and flow control, and are based on the concept of so called "pay functions" for the performance of a job, which depend on response time.

A technique is being developed which will allow very quick calculation of network (packet switching, or channel switching) performance. The method is based on a step-by-step modification of a computed network model from a very simple one to the desired one, and is constructed as an iterative process, which converges to the solution, representing the network performance.

One member of our group is making a detailed study of IIASA requirements in information systems and looking at ways of reconciling these with access to existing data-bases.

Internetworking brings many new problems. Gateways between networks are a topic of intensive discussion, especially where different packet conventions, lengths, etc., and protocols are in use. At IIASA, we would support moves toward an international standard host-host protocol, accepting that in many cases a protocol translation will have to be performed, and a one-one translation is much preferred to one-many translations. In our case, at least we shall have to give a lot of thought to questions of privacy and right of access, and mechanisms will probably be included in gateways to determine the right of passage.

Network services will need a lot of attention. More than most, we shall have to consider methods of uniform job control and file access. We also propose the provision of a network information service, whose job it is to provide information about the network, its resources and capabilities, the methods of use, and the rules of procedure. Distributed processing is an area of special interest. One can imagine an assembly line technique in which messages are processed by one host and then dispatched to another for further processing and so on until a result is finally returned or delivered to its destination. In this way, one could exploit the specialist capabilities of different systems to achieve some grand plan. Generally one thinks of such services as being provided by hosts on the network and not by the network itself, but gateways and protocol conventers are special cases of sequential network servers. It is important to see data communication as a means of interprocess communication and all that this implies.

15. CONCLUSION

We recognize that we have an ambitious project but are constantly being surprised by the interest being shown and new requests to participate. It is clear that at IIASA we are in a good position to unify networking developments, and this fits well with the spirit of our charter.

It should be mentioned that, during the past year, a number of successful experiments have been made including communication tests with Moscow conducted by V. Dashko and with

Budapest by P. Darvas. These experiments have paved the way to our present plans and shown the feasibility, at least in principle, of the project.

During the latter part of this year, we were joined by W. Orchard-Hays from the USA, and Y. Masunaga from Japan. Orchard-Hays has written a series of working papers on "User-Oriented Networks," and we expect these to have a strong influence on the provisions of our network. Masunaga's experience in the computer network for the Tohoku District of Japan will be particularly valuable.

To all of these people and many others the authors are indebted for a lot of advice and ideas.

References

[1] The Computer Communications Group of the Trans-Canada Telephone System, Datapac Standard Network Access Protocol, RES. 089, 1974.

[2] International Organization for Standardization, High Level Data Link Control Procedures, Proposed Draft International Standard on Elements of Procedures, ISO/TC97/SC6/1005, May 1975.

[3] International Organization for Standardization, High Level Data Link Control Procedures--Frame Structure, ISO/TC97/DIS3309.

[4] Mackenzie, A., Internetwork Host-to-Host Protocol, INWG General Note 74, 1974.

[5] Masunaga, Y., The Local Computer Network in Tohoku District, Japan, 1975. Internal paper, International Institute for Applied Systems Analysis, Laxenburg, Austria, 1975.

[6] Orchard-Hays, W., User-Oriented Networks: A Series, Parts 1-4, internal paper, International Institute for Applied Systems Analysis, Laxenburg, Austria, 1975.

[7] Pouzin, L., A Symmetrical Point-to-Point Procedure with HDLC, Réseau CYCLADES, TRA 523, 1975.

[8] SESA-LOGICA, COST PROJECT 11, EIN., Specification of the Interface Between a Subscriber Computer and a Network Switching Centre, 7104/5, 2240-2000 1/01, 1974.

[9] Sexton, J.H., IIASA Network Project, CSN002, July, 1975.

[10] Zimmermann, H., and Elie, M., Transport Protocol, Standard Host-Host Protocol for Heterogeneous Computer Networks, SCH 519.1, 1974.

Distributed Applications on Heterogeneous Networks

J.C. Chupin, J. Seguin, and G. Sergeant

1. SCOPE AND DEFINITIONS

Present trends in general purpose, computer networks are toward two-level structures--communication and application.

In the following, we shall be only concerned with the application level and shall study how to perform an effective resource and/or data sharing.

A net work application is *centralized* if only one host is responsible for all controls and functions. We do not address this type of application but rather concentrate on the notion of *distributed application* (i.e., those where some controls and/or functions are distributed among the participating hosts).

We consider that every application presents two classes of objectives--environmental objectives which are usually application-independent and functional objectives. The second section is therefore dedicated to the presentation of a *logical network machine* (LNM) intended to play in the network the role assigned to the basic software in a general-purpose machine. The LNM should provide the designer with all the tools necessary for a *direct and easy* implementation of heterogeneous distributed applications.

The last section should be considered as an example illustrating how a large data base management system can be distributed by using the LNM as a support.

2. THE LOGICAL NETWORK MACHINE (LNM)

2.1 Overview

It is the responsibility of this level to take into account the following environmental objectives: heterogeneity, distribution of control, and use of standard existing operating systems.

As illustrated in Figure 1, the LNM is placed between the "naked network" and the various distributed applications. It should hide from the user the problems by the heterogeneity and the loose nature of the coupling (telecommunication links).

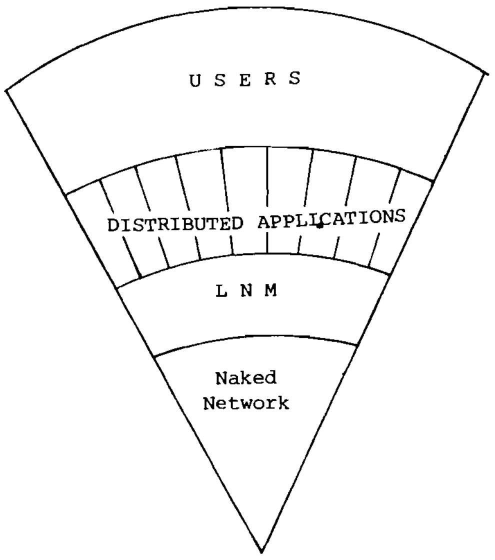


Figure 1. Hierarchical structure of a network.

The functions performed by the logical network machine can be classified into *referencing* and *scheduling*. Referencing includes the concepts of designation, retrieval, conversion, and transfer. This referencing function allows high-level addressing between objects located at different loosely coupled processors to be established. It is really a *network name space* which is built on top of the various local name spaces. New network objects such as *network jobs*, *network files*, etc., are defined and described in *network catalogs*.

As for scheduling, it is intended to submit and execute network jobs and network programs which may result in the simultaneous execution of *local jobs and processes* at the different locations. Therefore, the scheduling function includes the interfacing with the local operating system, the interpretation of a *network language*, and the intersite organization. There is such a lack of semantic interpretation in existing operating systems that one must add new software components to the hosts: these modules are called *control subsystems* whose collection forms the *control facility of the network*. Figure 2 shows the logical machine structure.

It is clear that the control is distributed among the different hosts. In order to achieve the global control, the different control subsystems must communicate with others. They do so by using the *transport station* which gives access to the communication subnetwork.

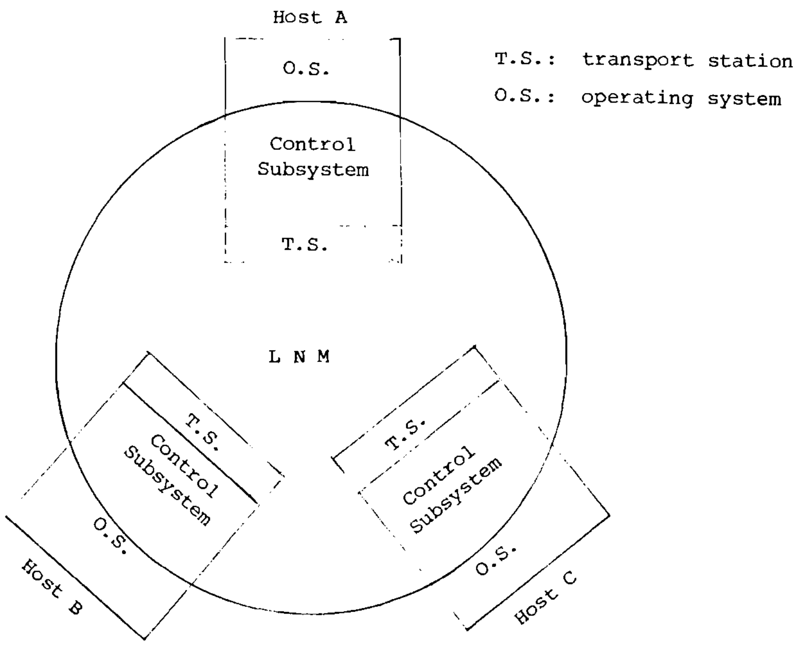


Figure 2.

2.2 An Approach to Network Scheduling

The heterogeneous nature of a network is a strong argument in favor of a uniform language for both command and programing. If not, since a network job (respectively, network program) results in the execution of several jobs (respectively, programs) under the control of dissimilar systems, every host should know the command languages (respectively, assembler codes) of every computer in the network.

It is therefore preferable to define a *unique* language that can be understood by all the participating machines. This language must be a pseudo-code to be interpreted by every host on reception. At this level, there is no reason why command and programing languages should be different from one another. An interpretative approach has been selected which provides a high degree of flexibility during the debugging stage.

Each host contains a general interpretor called IGOR [12] responsible for executing programs written in pseudo-code. The pseudo-code contains arithmetic logical, branch, and *if* instructions as well as normalized system calls, which permit the definition of algorithms.

We then obtain the remarkable facility of being able to uniquely define algorithms mixing command and programing languages which can be executed on any host in an interpretative manner. This provides a total transportability of modules, even of those containing system calls.

2.3 Structure of a Control Subsystem

A control subsystem is one of the tasks of the operating system on which it resides. Its overall structure consists of a process management kernel (PMK) controlling two types of processes--user and system processes.

The reentrant code of the interpretor (IGOR) is associated with user processes. The two system processes are a logger (responsible for contact establishment) and the local operator console handler, which provides some kind of network operator console.

The design of the process management kernel (called SYNCOP) is extremely important since it must take into account the heterogeneous nature of the network. It accepts standard creation/destruction and synchronization primitives regardless of the local operating system. It is a key element which provides the designer with a means of transporting control modules (i.e., those containing system calls).

2.4 Structure of a Distributed Application

So far, the study has been restricted to the application-independent objectives. We must now describe how to develop *functional subsystems* (FSS) whose union with the LNM constitutes a network distributed application. Clearly, most of the problems are specific to the type of application (nature of objects, naming, operating protocols, etc.). The next section will illustrate the case of a data-bank application.

Nevertheless, the global structure remains the same and is shown in Figure 3. Note that a control subsystem exists wherever a functional subsystem is present, though not necessarily vice versa. Note also the sharing of LNM between different higher-level applications.

2.5 Implementation of a Distributed Application and Transportability

Implementation of a distributed application requires that functional subsystems be implemented. One can simply write these subsystems in pseudo-code and leave the interpreter to do the rest. This could be a useful way during the debugging stage, but the overhead can become too expensive in the operational state. This is why it has been planned that local code be generated from pseudo-code.

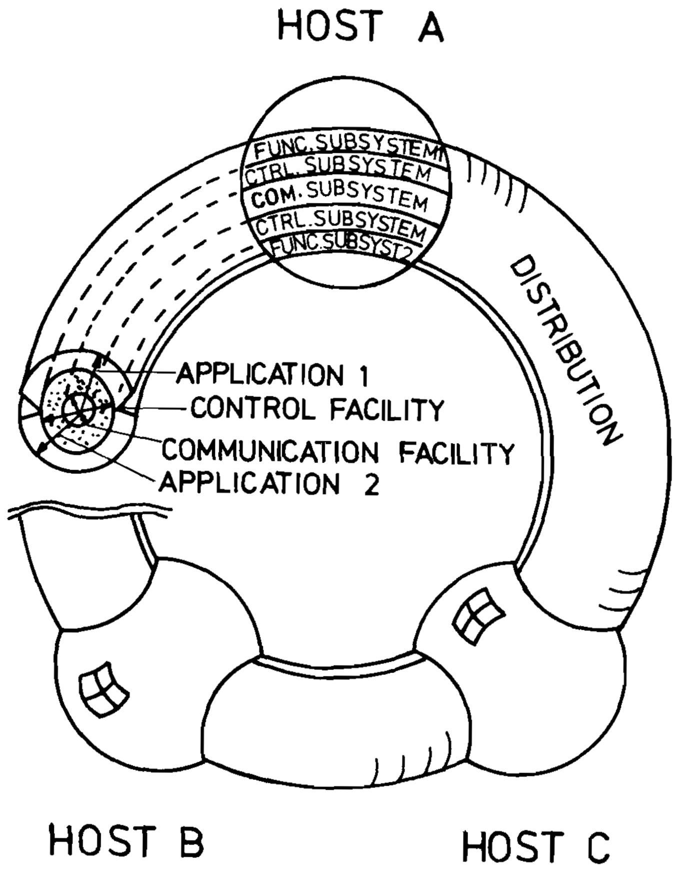


Figure 3. Structure of a distributed application.

This implies that a code be called for execution from a pseudo-code program. This facility is also very interesting if we wish to use "off the shelf programs" which have been previously developed on a given computer.

If we adhere to the following rules, the transportability of subsystem is automatic:

a) write as many programs as possible in pseudo-code;

b) use exclusively SYNCOP for all process creations and synchronizations;

c) provide a scrupulous functional layering to permit an eventual distribution.

3. APPLICATION TO DISTRIBUTED DATA-BASE MANAGEMENT SYSTEMS

3.1 Overview

Above-described concepts have been applied to the SOCRATE [6] data-base management system in order to ensure its distribution in the French CYCLADES network.

For us, a data-base management system consists of three sets: the *users* (be they terminals or batch oriented), the data-base management system *internal functions*, and the *physical data bases* (collection of files which are standard with respect to the housing operating system). Stated another way, we identify three major functional levels: terminal access method, logical access method, and physical access method.

A distributed data-base management system is then a data-base management system for which one or more of the access methods have been distributed.

The approach taken preserves the access method concept and ensures distribution by designing each level as a *heterogeneous distributed network application* (see section 2).

It is clear that such an approach, resulting in access distribution rather than data duplication, is opposed to the file transfer approach which leads inevitably to serious constistency problems (due to the presence of multiple copies of the same data set).

Finally, depending on which level is intended for distribution, we can observe several types of usage ranging from the simple user distribution up to a *distributed logical "back-end"*.

We shall now examine the distribution process of two significant levels with respect to the concepts developed in sections 1 and 2.

3.2 The Network Direct Access Method

As explained earlier, this access method deals with physical data bases (i.e., set of files in the usual sense). As opposed to a file transfer solution, the approach consists of *distributing access*, thus avoiding consistency problems. We shall first establish the correspondence between the concepts developed in section 2 and the network direct access method module [4,5].

A *user* (system or application program) is a process in the IGOR acceptance of the term. The associated algorithm consists in "threaded code" (IGOR instructions mixed with local code). The functional subsystems are represented by the direct access methods.

The *operating protocol* takes care of:

a) remote process creation/destruction,

b) synchronization,

c) interface with the different operating systems and the file management systems.

The replacement of the SOCRATE physical access method by network direct access method automatically provides a geographical distribution of physical data bases as well as the possibility of storing the bases on heterogeneous hosts (see Figure 4).

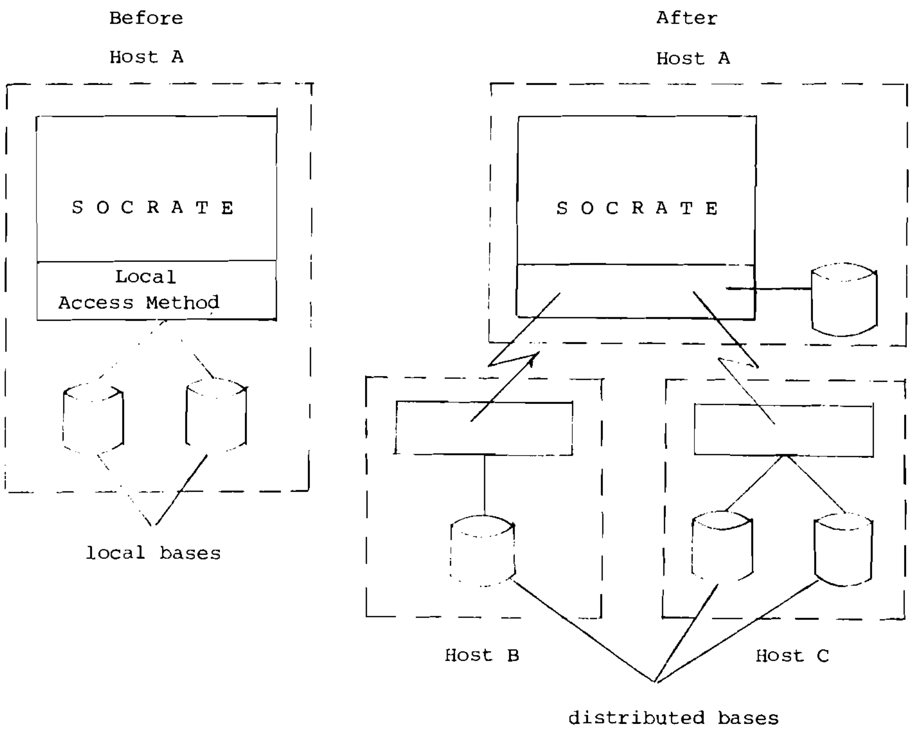


Figure 4. Physical file distribution.

3.3 Logical Access Method

This level concerns the manipulation of *structured data* possibly containing *interrelationships*. This means that we operate at the minimum *semantic* level allowing the *logical consistency* to be preserved during eventual up-date operations. It is very desirable to provide a certain independance from the data description language, so that the access method can be shared between several data-base management systems.

3.3.1 Choice of the minimum semantic level

This level must be:

a) independent of the user request language,

b) independent of the physical data organization,

c) hopefully independent of the access path.

3.3.2 Network modularity

In order to allow different functional levels to reside on different hosts, it is necessary to define a special kind of interface between the levels. The interface cannot use local addressing since such a concept is not transportable from one computer to the other. Besides, the functional layering must be performed with a great care in order to minimize network traffic and response time. In particular, updating or large tables should be avoided by two different levels if these levels are to reside on different hosts.

It looks particularly interesting to provide a means of sharing a given access method between different and rather independent applications. This implies that some sort of a "login mechanism" must be present at the access method level and that higher levels introduce themselves to the access method using a contact/connection protocol.

3.3.3 Distribution and implementation

In SOCRATE, the compilation of the user request language is performed in three phases: edition, pseudo-code generation, and finally code generation.

The SOCRATE pseudo-code contains a set of logical access method primitives separated by branch operations. Fortunately, this level matches the requirements indicated for a minimum semantic level. The distribution must take place at this level.

It has been decided to adopt the IGOR pseudo-code as a vehicle for the operating protocol. This means either that SOCRATE pseudo-code be translated into IGOR pseudo-code or that the SOCRATE compiler be modified so that IGOR pseudo-code be directly generated.

The only remaining problem is site selection. First, we must adopt for the data bases a mode of designation which is adapted to the network. *The network data base name* must contain, in one way or another, a site/host identification which will be made available to the SOCRATE compiler.

Secondly, we must structure the generated pseudo-code into blocks, each of them corresponding to a given host. This structuring can be performed according to different strategies: either we build for each user request a small pseudo-code block to be executed immediately, or we concatenate the blocks corresponding to the same host before interpretation. In the latter case, IGOR must provide two new pseudo-code operations in order to achieve synchronization: pseudo-WAIT (PSDWAIT) and pseudo-POST (PSDPOST).

4. CONCLUSION

The logical network machine provides a solution to the problem of heterogeneity while providing the designer with a means of implementing distributed applications. This should enhance the development of such applications, the lack of which is deeply resented by large-scale network users.

There remain now many outstanding problems in the fields of reliability and performance.

The most significant result is certainly the complete transportability provided by conjunction of a transportable programing language (FANNY), a normalized process management kernel (SYNCOP), and an interpretative execution (IGOR).

5. References

[1] Abrial, J.R., Data Semantics, Université Grenoble, 1973.

[2] Andre, B., et al., Distributed Data Base Management Systems, EEC Advanced Course, Serre Chevalier, 1974.

[3] Chupin, J.C., Control Concepts of a Logical Network Machine for Data Banks, IFIP Congress, Stockholm, 1974.

[4] Chupin, J.C., and J. Seguin, A Network Direct Access Method, Network Conf., Darmstadt, 1974.

[5] Chupin, J.C., and J. Seguin, MADRE, Note Technique, Centre Scientifique C.I.I., 1974.

[6] Manuel SOCRATE SIRIS 7/8, C.I.I., 4338 E/Fr, 1973.

[7] Du Masle, J., et al., Proposed Organization of an Interpreter Intended for the Implementation of High Level Procedures on a Computer Language, IFIP Working Conference on Command Languages, Sweden, 1974.

[8] Equipe Réseaux ENSIMAG, Equipe Réseaux C.I.I., SYNOOP, Système Normalisé de Commutation de Processus, Grenoble, 1975.

[9] Gien, M., and J. Seguin, FANNY, un langue d'écriture de systèmes portables, Note Technique, Centre Scientifique C.I.I. de Grenoble, 1973.

[10] Lambert, O., and C. Lambert, Verrouillage et partage d'informations au sein d'une méthode d'accès réseau, Project 3ème Année ENSIMAG, 1975.

[11] Sergeant, G., SOCYCRATE, Note Technique, Centre Scientifique C.I.I., 1975.

[12] Sergeant, G., and M.N. Farza, Machine Interprétative pour la mise en oeuvre d'un langue de commande sur le réseau CYCLADES, These, Université de Toulouse, 1974.

[13] Toan, Nguyen Gia, Fichiers Réseau/Protocoles Fichiers, Rapport D.E.A., Grenoble, 1974.

Facilities Rendered by Computer Networks to Developing Countries

T. Szentiványi

1. INTRODUCTION

Nowadays, a large number of computer networks are alive all over the world. Some have existed for several years now. Thus, we have various experiences at our disposal. A system for the designing of networks has been established, and the overwhelming majority of technical problems--such as simplification of use, economy of the system, extension, and normalization of services made available by the networks, in other words, unification of formal prescriptions for utilization (standards)--can be considered as solved. In addition, the connection of machines, cooperation of networks, and problems of secrecy and privacy have also been solved.

The services of the networks can be utilized by means of terminals, whose development is tending from passive terminals toward terminals including intelligent small computers. With this device, demands concerning the utilizer's training may be reduced. Concurrently, also terminals based on very simple telephone sets afford up-to-date facilities, in particular, in countries where the telephone network reaches practically almost every home. In any case, the number of terminals will increase at a very fast pace, accompanied also by a change of their types and forms of utilization. This means, of course, also an increased utilization of telephone lines. Today's average utilization of about 5 to 10% will increase to 30 to 50% with the extension of data transmission. Due to this, the traditional communication channels will no longer be able to afford the connection of required quality.

2. RELATION OF ECONOMICS TO COMMUNICATION PROBLEMS

Communication problems also may be considered from a different aspect, i.e., from the economic relation of the question. During the last 10 to 12 years, the price of computers for public use, consequently the per unit costs of processing, have decreased in a dramatic manner (according to some estimations, in the ratio of one hundred to one). In remote data processing, the proportion of the part of communication is not decreasing in such a scale. In fact, it represents a considerable proportion of the whole amount. In the highly industrialized countries, the characteristics of established computing networks, their structure and the services afforded by them, conform to the already developed computing centers and are based on them. In the case of developing countries, questions of an entirely different character have to be taken into account, as here the introduction of computing techniques is a condition of the economic upswing of the area and of the raising of its culture.

3. COMMUNICATION SYSTEMS AND THE DEVELOPING COUNTRIES

Developing countries have quite different possibilities and problems. In the first place, the fast rise of these countries has created an important demand for communication networks. From the point of view of technical development, it is favorable that the existing latest results be taken over, i.e., with new establishments, the momentary level of the highly industrialized countries can be taken on as a starting point. As no prior systems exist, no problem of adapting to them arises. Of course, the greatest problem arises precisely from the fact that these countries are in a state of being developed, and the low level of their technical culture is very closely related to the very low level of national income per inhabitant. Because of this circumstance, one can scarcely depend on technical expertise. Telecommunication and even transport connected with it does not exist on the necessary scale in most developing countries. As for the settlements themselves, they are widely scattered over enormous areas.

The question arises then, whether, under these circumstances, it is permissible to pose the problem of computing techniques, not to mention computer networks, or even to consider them as being demands of outstanding importance. In our opinion, it *is* permissible because such techniques can help in solving problems of capital importance--raising the cultural level, primary education, and supporting a health service. Naturally, outside computing techniques and networks cannot substitute for local expertise, but they can help in training the people of these developing countries. It results, therefore, that within developing countries demands arise in a new order, and the aim is not data processing or even the distribution of scientific information. The territorial distribution of the population does not show a concentration in towns as can be found in highly industrialized countries. Thus a computing service cannot be limited only to the larger settlements and towns but must be made available even in areas with a relatively low density of population.

Obviously, all this is not equally true for each developing country as there are very great differences among them. The quota per capita of the national incomes varies widely. This is why we are continuing our investigations and have chosen Iran as an example.

Iran's surface area is 1,6 million sq. kms; of its 33 million population, two-thirds are illiterate.

In fulfilment of its demands, the processing capacity can be afforded partly by computers installed in the country, and partly by the capacity provided by machines functioning over its borders. As the users of the machines (in this case the native users) are scattered throughout the country, numerous distant terminals that are connected by a network to the processing machines are necessary. The main concern is *where* to install the machines.

In the case of machines functioning across the border, it is expedient to realize the connection of telecommunication by a satellite in a packet broadcasting mode. In this case, it is not necessary that the machines be in the immediate neighboring state. For this purpose, every highly industrialized country in Europe or elsewhere can be considered as a site. According to recent investigations, from the economic point of view, the local costs are prevailing. But with stations installed in several rather large settlements and with terminal systems at small distances connected to them, there arise only the simple costs of investment for the ground transmitter-receiver station. At first approach, the operating expenses seem negligible; the costs of local communication are somewhat hidden.

In the case of computers installed within the country, the connection to the terminals by ground stations, first of all by cables, is preferred. The existing telephone network is unsuited for performing this task because of its limited capacity and its lines being only sparsely distributed throughout the countryside. The establishment of a new network is not only very expensive but its realization demands a very long time. Though the geographical situation of the country, with the high mountains surrounding it, creates rather favorable conditions for the establishment of a communication with microwaves, such a system demands considerably more experts to handle it than the former solution does, but its realization demands less time.

Analyzing both possible alternatives, we should like to mention the following considerations, presented here in a compact chart.

Computers installed in the country with terminals connected partly by ground and partly by radio connection	A system of machines installed across the border with terminals grouped around an independent ground station
a) Machines and investments are necessarily of great volume.	a) Only rent costs arise.
b) A large number of operating experts are necessary.	b) Personnel attending the terminals and ground stations is demanded.
c) The establishment of the network is a fundamental demand that can be realized in a continous, centrally coordinated system but the establishment demands long time.	c) Establishment of ground stations cause real establishment costs. Their communication expenses are lower than their processing costs. There can be established territorial units, realizable independently from one another, that can be started up in a short time.
d) Checking with foreign institutions is not necessary.	d) Requires an alignment in legal, etc., respect with machines installed elsewhere.
e) Secrecy and privacy probelms can be surmounted more easily.	e) Solution of secrecy and privacy problems problems is more difficult, but this would cause no difficulty in the initial period.
f) It provides a more favorable means for controlling central bureaux, state institutions, and governmental offices.	f) Government and central leading organizations would be subscribers to the system, and only be the proprietors of the terminals.
g) The comparatively fast moral obsolescence of computers is less negligible.	g) Such a network dynamically compensates for lack of capacity and is less sensitive to moral obsolescence of system elements.

4. CONCLUSIONS

In a developing country, the most important aims are the promoting of education, the raising of the cultural level, and the establishment of a health service. These demands can be aided considerably by a computing technique with its interactive mode of operation. The satisfaction of additional demands for processing or use of information systems can take place only after these initial demands have been satisfied. It is more expedient to base the service on the foreign computer capacity. This creates the base for the computer network in the country, contributes to the country's gradual further development, and assures the equalization of differences in mentality and attitude.

With the centers being established in big cities (those having, incidentally, a university), the problem of connecting the scattered settlements to the network is raised. In addition, there is the problem of the connection to other national or international networks. This problem is a solvable one. The lack of experts can be surmounted in this case, but its solution requires considerable effort.

In realizing these problems, a new attitude was adopted that signifies a step a) toward establishing a computer network all over the world, an information system assuring telephone service, and b) toward "compunication" to be established at length with the patronage of the UNO. Of course, a number of problems still await solution, beginning with standardization and ending with assuring the necessary number of experts.

References

[1] Abramson, N., Satellite Data Communications for the UN, *Manuscript*, 1974.

[2] Enslow, P.H., Nontechnical Issues in Network Design - Economic, Legal, Social and Other Considerations, *Computer* (1973), 21-30.

[3] Fano, R.M., On the Social Role of Computer Communications, *Proc. of the IFEE*, 60, 11 (1972) 1249-1253.

[4] Kuo, F.F., Political and Economic Issues for Internetwork Connections, *Proc. of the Second International Conference on Computer Communication*, Stockholm, 1974, 389-391.

[5] Roberts, L.G., TELENET: Principles and Practice, Eurocomp-Computer Networks, Brunel University Conference Proceedings (forthcoming).

[6] Samuelson, K., Communicating within a World System, *Proc. of the Second Internat. Conference on Computer Communication*, Stockholm, 1974, 361-366.

Man/Computer Communication: A Problem of Linking Semantic and Syntactic Information Processing

K. Fuchs-Kittowski, K. Lemgo, U. Schuster, and B. Wenzlaff

Through the development of more efficient computer networks, the computer will experience more economical usage, the technical reliability of the system will rise, and computer technology will be directly available to the user. General and special communication languages will be developed to serve as a bridge between the programing and control languages, thus allowing the user to formulate programs readily acceptable to the machine. In addition to this decisive trend of development, there is another aspect which, in our opinion, seems to be of growing significance, namely, the integration of these highly efficient soft and hardware systems into the complex of problem processing and solving by man. We are of the opinion that this integration cannot and must not be permitted to remain restricted to the mastering and application of predetermined more complex models and partial models, but, must also encompass the process of dealing with new data processing tasks arising directly from data processing. Thus, as we shall show in our paper, the essential basis for this is given in the formulation of tasks for data processing consisting of the transition from semantic to syntactic information processing. The capability of the computer operator to deal with technical systems cannot, therefore, be restricted to the handling of programing and communication languages, but should also include the creation of tasks and programs for use in processing. We believe that the ensuring process of opening completely new fields of application with dynamic phases of intersection of semantic and syntactic information processing is an issue which should not be considered a problem of the user alone.

Informatic, being more than a purely mathematical and technical discipline, involves analysis of the general structure of spheres to which automatic information processing can be applied as well as the conditions by which the hard and software systems at one's disposal can be used to the best advantage. We view this sphere of informatics as a form of organizing information processing, beginning with a scientific analysis of the interrelations between semantic and syntactic (human or mechanical) information processing from which general principles for the design of information systems can be derived.

With regard to the trends of development we have outlined here, we have gained experience in the establishment of information systems that has induced us to devote the same degree of attention to the trend toward more efficient man/computer systems with special organizational components.

Man/computer systems are complex information systems of purposeful interrelated syntactic and semantic processes of information processing. Therefore, they require detailed knowledge of the rationale of the transition from the semantic to the syntactic phase and vice versa. To leave these problems untouched would mean an even greater limitation of the range of application and the social effectivity of computer systems. In turn, the gap between theory and practice of mental work and analysis of social systems, which is already evident, will deepen even more.

If we understand by system analysis a method that is applied to solve organizational, control, and decision problems in social systems, then system analytical activity from the aspect of information and data processing always leads to the setting up of information systems, that is, to a designed cooperation of human and machine, or semantic and syntactic information-processing operations.

A special task in the design of information systems is to consider how the advantages and the special performance of specifically human, semantic information processing can be linked with the performance of syntactic information processing (structure-processing automats) so as to form a high performance overall system.

The analysis of complex information systems (for instance, a hospital consisting of various subsystems, such as the administration, the patient, documentation, medicine, etc.) shows that their set-up differs significantly.

a) Information systems are mixtures of semantic and syntactic information processing.

b) Information systems are efficiently aided by different types of automated information-processing systems.

c) Information systems according to points (a) and (b) have different forms of man/computer communication for allowing purposeful combination of human and machine information processing.

d) Theoretical problems of social organization and problems of man/computer communication may be rather similar, no matter whether a single (autonomous) computer or computer networks are used.

J.C. Chupin [2] also writes: "It is our main concern that the logical network machine appears to the user as a single computing facility."

Starting from a clear limitation of the object of automation to formalizable operation and from an analysis of the process of working upon problems, a distinction can be made between schematic and nonschematic tasks [6,9]. In this way, it becomes obviously clear that, on the one hand, a conception of full automation is mistaken and that, on the other hand, a restriction of automation to only processing schematic tasks is not a sufficient picture. It can be shown that modern computers, under certain organizational conditions, may also be used to process nonschematic tasks, that is, newly emerging, and normally once-occurring tasks. By appropriate organizational measures, it is therefore possible to obtain dynamic forms of linking human and machine-operated information processing without having to develop completely new foundations in programing technology. In this respect, the setting up of an information center, by which a special form of man/computer communication is carried out, plays an important role. This will be presented in greater detail.

The basic problem in our considerations about general informatics is, furthermore, the achievement of the clearest distinction possible between semantic and syntactic information processing and the differentiation of information into descriptions and date, that is, whether we are dealing with formalizable information that can be described in extensional terms or not. A sufficient but essential precondition for the use of structure-processing computers or of a computer network for rationalizing information processing consists in formalization of the information (that is, it must be available in the form of data) and of the operations (that is they must be available as algorithms).

From these distinctions emerges the problem of the effective coupling of semantic and syntactic information processing and the integration of machine operated, information-processing systems into the information-processing systems of the social reproduction process.

The exchange of syntactic information between systems is characteristic of communication between computers or between man and a computer as well as of biological control and regulation mechanisms.

Two users can appear in a computer network in regard to the computer: man, who is directly linked to the machine by means of a terminal, or another computer which cannot solve the tasks it is programed to do without the assistance of the first computer or which passes on to it the entire problem, or which is linked to it due to data transfer.

Both the computer/computer and the computer/man exchange of information is limited, just as with biological control and regulation mechanisms, to syntactic information, which results from the aforementioned conditions of computer application.

In our opinion, the man/computer communication (as the exchange of information between the computer and the human being) is often confused with the exchange of information between human beings, due to the fact that the significance of the differentiation between the semantic and the syntactic is overlooked. The result of computer or computer network processed data is syntactic information which man uses in the overall process of his semantic processing of information. Herein lies also the essential difference between the computer/computer exchange of information, which is the processing of syntactical information in the sense of transition, and the exchange of information between man and the computer. He assigns formalizable semantic information to the machine, mapping semantic to syntactic information structures, and thus creating the conditions for carrying out machine-operated processing of structure. (This is represented in Figure 1 and Figure 2).

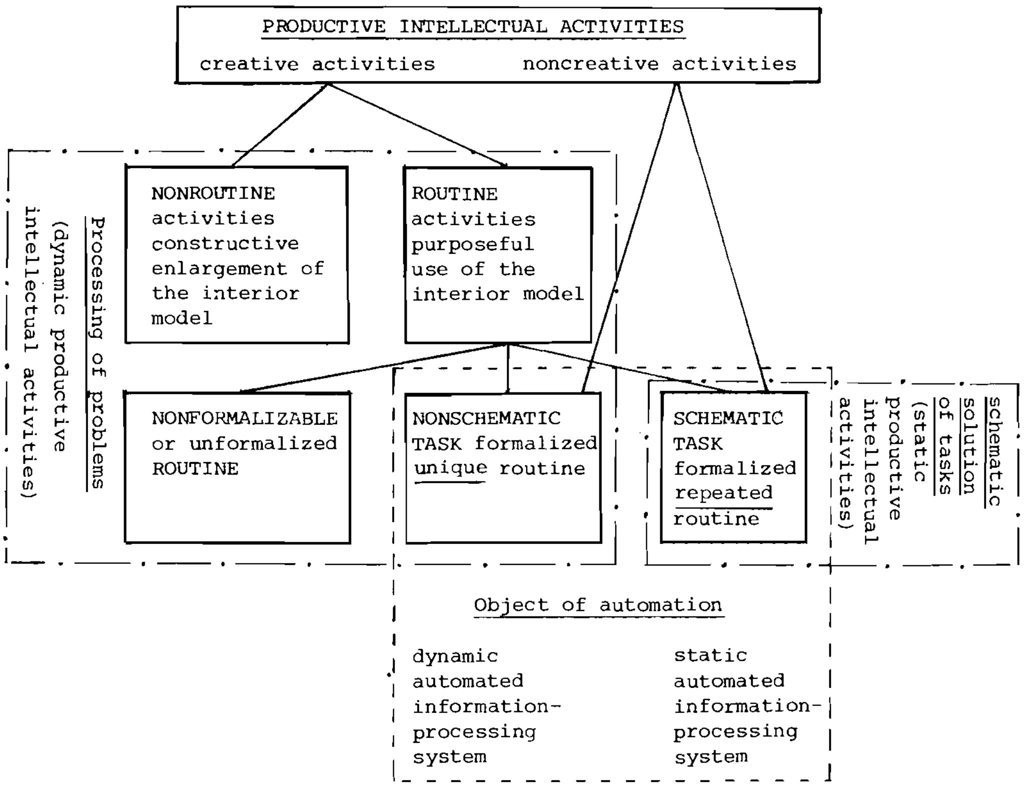

PRODUCTIVE INTELLECTUAL ACTIVITIES
creative activities
noncreative activities
NONROUTINE activities constructive enlargement of the interior model
ROUTINE activities purposeful use of the interior model
Processing of problems
(dynamic productive intellectual activities)
NONFORMALIZABLE or unformalized ROUTINE
NONSCHEMATIC TASK formalized unique routine
SCHEMATIC TASK formalized repeated routine
schematic solution of tasks
(static productive intellectual activities)
Object of automation
dynamic automated information-processing system
static automated information-processing system

Figure 1. Subdivision of productive intellectual activities according to the aspect of information.

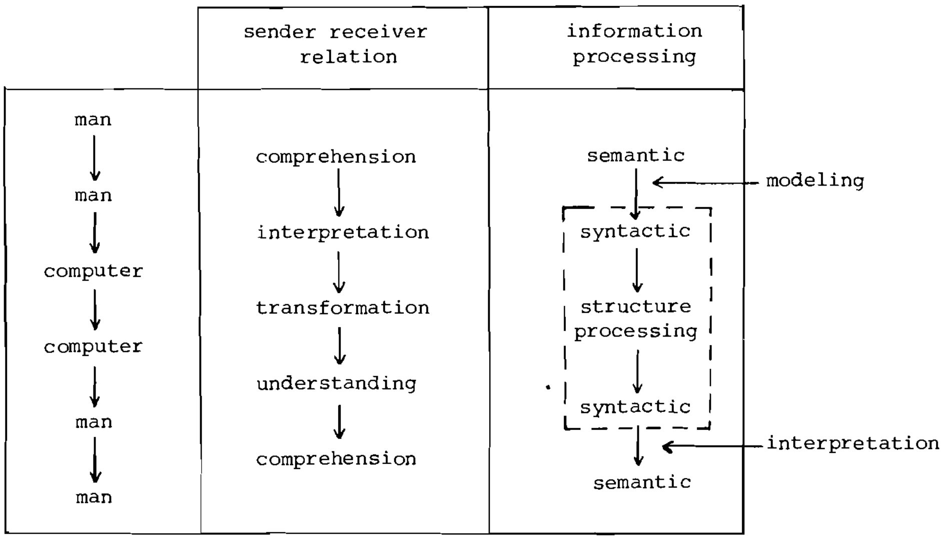


Figure 2.

Therefore, efficiency of electronic data-processing systems depends not only on the technology that can be employed, but rather on the degree of scientific insight into the informational processes and the obtained organizational solutions. In this respect, the point is not only to detach formalizable information (data) and operations from the total amount of human activities, but, above all, to reintegrate automated information-processing operations into the complex process of human work in a useful manner.

Man/computer communication is to be considered from two interrelated aspects. On the one hand, it is a mere instrument to rationalize human information-processing operations. On the other hand, it is more and more becoming an integral part of the intellectual process of work performed by man. As a mere instrument, computer application is limited to supplying and using models. These models are representing the continually repeated tasks that can, therefore, be put into a scheme. These tasks can temporarily be separated from the problem processing and solving process of man engaged in creative activities from his semantic information processing. But we have also to consider the possiblility of solving, with the help of the computer, the tasks which are not continuously repeated, which are relatively unique and occur unforeseeably in human problem processing and solving (nonschematic tasks) which are worth being automated.

Then the dynamical transitions between emerging new tasks and their solution by computer must be mastered. Computer application always means to link syntactic and semantic information-processing operations.

Semantic information processing is the combination of meanings of information to form new meanings. It is typical of man's intellectual information processing to be concentrated upon the contents. The structural processes which are underlying the meaning of words and sentences are carried out by man unconsciously. By making man continually aware of the structural processes, his creative thinking would be interrupted and troubled. Hence, semantic information processing is *immediate* processing of semantic statement contents.

Syntactic information processing is a transformation of the structure of information carriers. On the basis of unique rules between information carriers and their meanings, new meanings are ascribed to them. The contents of semantic statements are processed by the mediation of structural transformations.

The aim of modeling the semantic information processing operations is mapping the semantic coupling processes of operations with syntactic representations of semantic information. When modeling semantic information-processing operations, immediate processing of meaning is replaced by indirect processing of meaning. In the ideal case, functional identity is achieved between processing semantic statement contents and structural processing. Models satisfying these requirements are considered as a formalization of information-processing operations.

When modeling information processing operations, that is, representing information processing operations by syntactic information-processing operations, as a rule, *one* essential aspect is defined. We speak then of a one-aspect definition of the essence of things.

Semantic information processing is characterized by complexly viewing the essence of things from many directions and by considering the unity of the essential and nonessential features when combining meanings. We speak of a many-aspect definition of the essence of things then. By modeling, the operational scope is limited in which semantic information-processing operations are taking place, and functional principles of semantic operations are replaced by principles of transferring syntactic structures.

Isomorphic mapping of semantic information-processing operations on syntactic ones is, therefore, impossible. Still, modeling of semantic information-processing operations should be tried, where it is reasonable. Scientific cognition of man proceeds in this manner and thereby successfully solves practical and theoretical problems. Exact (mathematical) scientific explanations of facts can be represented in formalizable science languages. By the tension between semantic and syntactic information processing, the possibilities and limits of mathematical modeling are shown.

Communication between man and a computer, at any rate, raises the problems of linking syntactic and semantic information-processing operations. It is possible to use a computer for rationalizing human thinking on the condition that the mapping of semantic information-processing operations on structures capable of being processed by machines has been mastered and the results of it can be included into the system of human information processing. When considering the complexity, variability, and dynamics of human semantic information processing, three different basic situations of computer use are to be distinguished:

a) Semantic information-processing operations have been completely mapped on syntactic ones. They are replaced by machine-processing functions. Integration is reduced to receiving the results of the syntactic information-processing operations assigned to the machine. We speak of solving schematic data-processing tasks with ensuring evaluation of the resulting data.

b) There are formalized and hence uniquely defined submodels for syntactic information-processing operations that can be included into human information-processing operations by variable definition of different parameters and various concatenation. Subtasks that can be presented as a scheme are solved then, the resulting data being used for continuing or completing the semantic information-processing operations.

c) Semantic information-processing operations lead toward formulating tasks that can be mapped in principle on syntactic information-processing operations. There exists no ready-made scheme for them, however. Therefore, we speak, in this case, of tasks that can not be represented as a scheme. Human and machine information processing can be linked here if there is a rational method--a worked-out technology of modeling. Design and programing are to be considered as task and program generating.

This situation corresponds to processing and solving problems leading to formulating, with unforeseeable data, processing tasks and thus to modeling as an integral part of the problem-solving process.

There are different situations in man/computer communication:

a) the existence of simple and complex overall models,

b) the existence of determined submodels in variable information-processing operations,

c) the production of submodels according to the aims of the complex information-processing operations.

From these situations result different characteristic points of intersection in man/computer communication (see Figure 3). By point of intersection we mean the way in which information processing typical of man and of the computer coincide.

There are:

a) static points of intersection between semantic and syntactic information processing. The static character results from working on defined models and the simple supply of results.

b) Variable points of intersection between semantic and syntactic information processing. The variability consists in working on defined submodels and in the possibility of correcting obtained results by changing the parameter determination and concatenation of models.

c) Dynamic points of intersection between semantic and syntactic information processing. The dynamic results from the decisions of man to produce submodels according to the problem processing; the models exist here as a potential, as possibilities, and every decision determines the intersection between semantic and syntactic information processing.

In case a), we speak of selective man/computer communication, as determined models are chosen from a stock.

In case b), we speak of selective man/computer communication with variable data file input and variable concatenation of programs. Determined submodels are chosen from a stock.

In case c), we speak of generating man/computer communication covering both generating input data files and program generating. Data file generation is to be regarded as a combination of data or a stock of data, and program generating as an interplay of the operating system and special programing systems to produce workable evaluation programs.

In all three cases, at least in cases b) and c), communication between man and computer can take place as a dialogue, that is, as some interplay between human and programed decisions. In this context, selective dialogue can be distinguished from generating dialogue between man and computer (see Figure 4).

In most cases, only simple communication and direct dialogue are distinguished from each other. On the contrary, we think a distinction between three basic forms of man/computer communication necessary. The question of dialogue design is a minor aspect subordinated to this tripartition.

The various possibilities of designing the dialogue depend on the type of intersections. In case a), with only simple parameters being handled, the possibilities of a dialogue are limited.

Type of Intersection / criteria for the user	Statical	Flexible	Dynamical
Knowledge of the models	Use of the model as a black box	Use of the model as a parameter controlled black box system	Knowledge of the elements of the system for generating tasks and programs
Concatenation of models	None	Selective concatenation choice of submodels and parameters and mastering the possibilities for concatenating submodels	Generating concatenation Ability for problem-related concatenation of the elements of the system for generating tasks and programs
Data handling (input and output)	Simple data handling supplying or calling in data files/activation of prefabricated output functions	Complex data handling controlled supply of data files for concatenated submodels/concatenation of prefabricated output functions	Generating of data files generating and using data statements to produce combined input data files/ generating and controlling output functions
Production of models	None	None	Problem-oriented generating of tasks and programs
Communication languages	Simple communication language	Extended communication language	Programing control and macro-languages, language elements for generating tasks
Type of communication	Simple communication	Extended communication	Indirect (transmitted) communication

Figure 3.

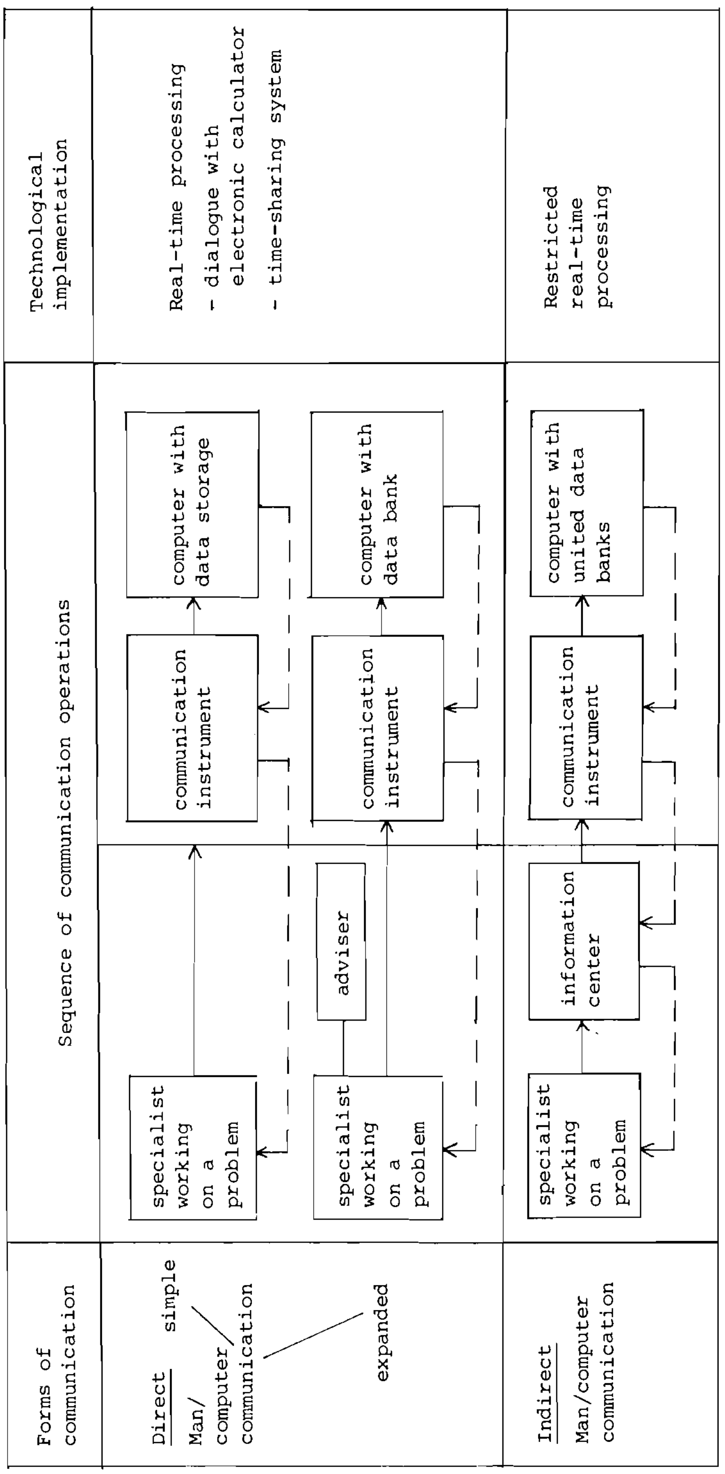

Forms of communication
Sequence of communication operations
Technological implementation
Direct Man/computer communication
simple
expanded
specialist working on a problem
communication instrument
computer with data storage
adviser
computer with data bank
Real-time processing
- dialogue with electronic calculator
- time-sharing system
Indirect Man/computer communication
information center
computer with united data banks
Restricted real-time processing

Figure 4. Man/computer communication in (on-line) dialogue operation.

The requirements to the user are differing in the three mentioned cases of man/computer communication.

In case a), the user must know the logic of a model and be able to use it with variable parameters. He must be capable of putting in data and calling off data from a data storage device.

In this case, as in most cases after a short initial period, the user is able to carry out communication with a computer completely independently. We call this simple man/computer communication.

In case b), the user must know the logic and performance of the submodels and be capable of putting in varying data, as in case a), and, in addition, of selecting and concatenating submodels.

The requirements to the user are so extensive here that some aid by a problem analyst may become necessary. In this case, communication may be carried out either directly between the user and the computer, or with the aid of a problem analyst. We therefore speak of extended man/computer communication.

In case c), the user is required to know the logic and performance of the system in generating some basic models. He must have a good mastery of all basic forms of data handling and of controling program generating. We think, in this case, a division of labor is useful between the man who works on a problem and a problem analyst or programer. Then we speak of indirect man/computer communication.

For man/computer communication, a language is needed that can be interpreted both in semantic and in syntactic terms. That means, it must stimulate human thinking and trigger off computer processing functions.

The character of the necessary communication language depends on the goals of communication between man and computer.

In case a), it is sufficient to design a simple communication language. Its elements are related to solving the following problems: data input or call off, elementary data handling, activation of a complex program, and choice of lists with results.

A few examples of this are languages for generating lists, such as RPG or LISP; query languages for management systems, such as CIS or POLIS [1]; query languages for retrieval systems, such as AIDOS or FAKIR [18]; and simple dialogue languages with elements adapted to the user's habits of language.

In case b), it is necessary to develop an extended communication language to formulate about all the program control of concatenated submodels and problems of complicated data manipulation.

For instance, the simulation of an ecological system calls for a combination of many specific functions that can be called off in terms of a special simulation language.[1] Designing machines or buildings necessitates another logic and another complex of functions and operations. To call them off and combine them, special programing languages have been designed.[2] There are also special programing languages for planning economic systems. It is typical of programing languages at this level that, in most cases, they have such denominations for functions and subprograms that are familiar to an expert in the respective field.

Insofar as these languages contain elements for dialogue, the user can start a conversation with the computer, a special dialogue technology (terminals and displays) being given.

An example for a user-oriented dialogue system based upon the APL language and designed for the process of planning is provided by CPL (conversational planning language).

In case c), the design of a language for communication with a computer is identical with designing rational programing, macro, and control languages to handle all elements of automated information-processing systems. Only by means of such languages, mostly universal programing languages, can the potential of a system be used in full.

In cases a) and b), special communication languages are useful for people unfamiliar with computers. In case c), however, communication with a computer will be the affair of specialist programers. By means of an ASSEMBLER or of general programing languages, such as FORTRAN, PL/1, APL, or others well suited for formulating various algorithms, the potential of the computer can be well utilized.

---

[1]Here, we think, for instance, of the language system CML, presented at Hasenberg, and underlying the simulation language HPMOD [3].

[2]For instance, in the GDR, we have the languages:

WISMAR, Wirkflächenverfahren-Sprache für maschinenbauliche Rationalisierung (effective surface language for rationalizing machine tools [12].

PROBAT, PROFUND, for the dialogue of civil engineering designers by means of a single-purpose computer.

MAUS, mehrdimensionale Messdaten-Auswertung von mechanischen Werkstücken (multidimensional data evaluation of mechanical parts).

For people working on problems, we must aim at designing language elements for formulating tasks rationally allowing a very simple transition to machine readable formulation of a task (that is programing). These language elements are a link between the formulation of tasks in semantic terms and the intended syntactic information-processing operations.

The development of a special communication language ought to be limited to the application cases a) and b). The special communication language serves to design an immediate dialogue between the user and the computer, and at the same time, always means real-time processing.

Application case c) requires very detailed special knowledge in the field of computer technology and of modeling processes. Above all, there occur nonschematic tasks, and the necessity of immediate action and decision is not typical of the creative problem-solving process. Therefore, it is not necessary to build direct man/computer systems with real time processing. To give aid to the process of working on problems and solving them, especially for management staff of higher levels, indirect man/computer communication is by far a more advantageous solution. In indirect communication, a special process of work is switched in between the syntactic and the semantic information-processing operations. This process of work comprises the fact process to be modeled until the formulation of a machine-intelligible data processing task, including cooperation of the user with the modeler (problem analyst) on the semantic level.

So, the function of mediating is carried out by a special organizational component of man/computer communication that is often called an information center [15].

The distinction between direct and indirect communication, as outlined here, once more emphasizes the necessity of regarding man/computer communication in a very large context. By this, we cannot understand the immediate conversation of man with the computer only.

The necessity of indirect communication and of setting up an information center results, above all, from the necessity of reducing to a minimum some basic difficulties of information processing automation, as well as of the possibility of mastering large quantities of data, and of secrecy protection.

In indirect communication, the user has direct contact not with the computer but with the problem analyst in the information center.

The exchange of information between the user and the information center is carried out on the semantic level. The user sends problem-oriented assignments to the information center and receives the results of the syntactic processing in a semantically processed manner. In the information center, the user, on the one hand, is confronted with an aim of stating, in more precise terms, the process of working upon the problem and of formulating the assignment in greater detail, and, on the other hand, the aim of formulating the final assignment in a language that is intelligible to the computer. The information center has direct contact with the computer which can have either an on-line or off-line character.

Hence, the following functions are fulfilled by the information center:

a) to advise the user when formulating and processing problems by submitting a problem-oriented set of data that are relevant to the problem;

b) to convert the user's wishes, which have been made more precise, into evaluation programs;

c) to develop analysis programs with an aim of making visible the potential existing in the data and program store, and of making an independent contribution in support of the user's problem processing;

d) to organize the available data, its updating and storage;

e) to maintain and further develop the programing system.

The qualifications of the information center staff must include special knowledge for communication with the computer and allow them to aid the users as specialist consultants, to a certain extent.

In this way, it is possible to use to the maximum all the available potential both in the semantics controled process of conceiving the solution of a problem, and when using the data and program files of a data bank.

In the literature, in most cases, indirect communication by means of an information center is considered as a temporary solution caused by the state of technology. Our explanations, on the contrary, make clear that an effective linking of semantic and syntactical information-processing operations calls upon an information center as an essential element of man/computer communication. Above all, this helps to overcome basic difficulties of data processing that we may call paradoxes.

As the tasks to be solved by computer cannot be reduced to merely using ready-made models, the sole consideration of direct communication that is typical of cases a) and b) is insufficient. By this, we would impose limits on computer application that are unjustified by the facts. It is the generating integration of machine-operated information processing into the complex information processing of man. This trend will obtain a decisive role in future. The setting up of automated information-processing systems will cover all spheres of social life.

To sum up, man/computer communication can be characterized as semantical directing of syntactic information-processing operations. It comprises the development of simple and complex models to work upon schematic tasks, including a technology for producing problem-related submodels for nonschematic tasks, hardware and software and organizational components being united to form efficiently overall.

Indirect man/computer communications extend the possibilities of applying electronic data processing to solve tasks that can be programed, and makes the possibilities of direct man/ computer communication more complete.

The ever larger application of electronic data processing allows recognition of two basic directions which must be considered in theoretical reflection and in practical systems design. On the one hand, the development of hardware and software in the next years will make possible an increasing user friendliness of electronic data processing, and thus a design of ever more efficient direct man/computer communication systems, but, on the other hand, the indirect forms of man/computer communication aquire a constantly growing importance for the dynamic integration of the potential of the data-processing technology into complex forms of problem processing and solving operations in management, planning, and scientific research, as well as in diagnosis, therapy, and in development. From this, there results a more and more intimate interaction between problem analysis adapted to the needs of the computer, the designing and programing of information-processing operations, and the creation of useful complex information systems.

In this way, the organization of information processing is becoming a link between computer science and the rational organization of processes of work on a social scale. For this reason, informatics as a new science of integrating machine-effected information processing operations into the whole systems of human activities is to be considered in a larger sense than is suggested by the term computer science.

The problem of man/computer communication as linking semantic and syntactical information-processing operations turns out to be a central problem in the development of the theoretical foundations of informatics.

A number of technical problems and theoretical problems of organization are involved in linking up computers to computer networks [13]. One of the technical questions is the optimal layout of the network according to the given technological facilities, the performance demanded from the net and according to economic conditions [4]. Other technical questions are those of network-control, line-control, of adaptation in heterogeneous computers and the communication system, which may include so-called communication counters, for instance in the case of the net ARPA: IMP (interface message processor) [16]. As computer networks are supposed to contribute to a better use of capacities of the individual computers and of overall resources [11], it is necessary to investigate the forms of cooperation between computer network and the user under theoretical aspects of organization. Systems planning is involved when trying to satisfy all of the individual user's demands and, at the same time, to make best use of the resources of the computer network. The user's demands can very well contradict each other and thereby make an "ideal" solution for the computer network under economic aspects difficult. The individual user, when processing a problem, should be confronted with the computer network as little as possible. It should be the aim to be able to place, at the user's disposal, the means of a complete computing center independantly of him having, in the direct or indirect dialog, a simple terminal or a larger computer.

At present, batch processing is dominating among the ways of running of computers. Starting from networks already present, for instance ARPA, TSS Network, OCTOPUS, etc., [19], we will have to consider for the future a mainly interactive way of running (direct dialogue). The running of a computer network will, therefore, consist in a mixture of simple, extended, and indirect (mediated) communication. Questions concerning cooperation of the different ways of running [14] and concerning cooperation with the user will arise. Batch-processing jobs will enter into the net not only via the operator or the information center but, according to the technical facilities, also via batch processing or more simple terminals. The technical integration, also of the programing, of the network determines whether distribution of resources is accomplished manually or to a large extent automatically. In heterogeneous networks, the incompatibility of job control languages causes difficulties in adaptation among the computers, which cannot be eliminated even by a standardized network language (that is, one valid to all computers). The forms of communications between the computer network and the user are the same as those between the user and an individual computer.[3] Direct communication between the network and the user is essential particularly with dispositional systems with short-time response, for instance in the case of travel-booking systems (space reservation system) such as SITA (Societé-Internationale Télécommunication Aéronoutique). Direct communication also takes place when controlling technological processes. If the user is working with a data bank, which has been designed for the solution of extremely different tasks (schematic as well as

[3]This is true if we consider the computer networks essentially from the aspect of information systems, and preferably not from the aspect of the new qualities, which could arise from the multifoldedness of the resources, for instance when the networks are used for immediate communication among people, as at a conference.

nonschematic) such as universities or hospitals may be confronted with the choice of a computer with the most appropriate resource for solving problems, a particular task cannot be accomplished immediately or automatically. The putting over of the tasks to the appropriate computers as well as the advising of the users of how to use the system effectively makes an information center absolutely essential in the case of indirect communication. Neither will such a center be of only a provisional short-time help until better technical means, also of programing, are available (contrary to other opinions). The establishment of a center of information is absolutely necessary when a problem-related task and programing are called for. One can, more or less, ascertain thereby if the technical basis consists of a single computer or a network. We are even of the opinion that the computer network will create additional tasks, based on the degree to which the information center is equipped. The reason for the installation of an information center evolves, in our opinion, from the solution of nonschematic tasks related to the following elementary situations of data processing which, possibly, one could describe as paradoxical:

a) the handling of a problem requires the mastering of tremendous amounts of data, although it is impossible for an individual user to master such amounts of data knowledge. We proceed from the viewpoint that the user should not acquire extensive knowledge and capability for computer or computer network use. It must be noted that the degree of complication related to the technology involved in mastering vast amounts of data considerably increases with the amount of data and the number of computers storing only part of the total amount of data.

   Thus far, the situation in the use of computers is that the operator has had to acquire extensive knowledge about computers or the computer network. We have concluded that the human being is capable of dealing with a maximum of ten thousand up to a hundred thousand pieces of data depending on the average degree of differentiation. It is practically impossible for the individual user to deal with several million bits of data. Thus, we submit that it is impossible for the individual user to have an overview of vast amounts of data, which most probably are stored in several different computers.

   In this case, the information center has the task of advising the user in selecting the necessary data and program functions needed for the solution of the problem.

b) The information center makes sure that, despite the unavoidable changes in available data (updating expansion, filing the data, etc.), the processing system remains stable and topical for the user. These changes are made by the staff of the information center. The user must exercise a control, however, to determine if the system is capable of providing the information necessary to solve the problem to be dealt with.

The information center also informs the user about changes in the link-up possibilities of the computer network. The stability of this system can also be secured through the establishment of priorities for tasks that are applicable within the network.

c) An effective use of the given amount of data requires usage by all members of the sphere of application, although the conditions for order, security, and protection of secrecy limit the dissimination of detailed knowledge about the type, structure, and operation of the system. There are two cases that must be differentiated hereby: First, the data stored in the computer assigned to this sphere must be secured. Second, the data bank of the entire computer network must be secured against unauthorized usage.

In the first case, such security can be implemented through program and organizational means taken by the information center of the area of application.

In the second case, the securing of data occurs through corresponding safety measures included in the program, and this is the reason that the other information centers of the computer network do not receive information related to these data.

The language elements necessary for the computer network control to activate the tasks are also added by the information center.

The establishment of a main information center could become necessary based on the degree to which the network is equipped, for instance, if the distribution of resources (i.e., computer still time which can be made available for general usage) is determined centrally. Among these decisions are included the stipulation of limitations and priorities [4].

In the same way in which computer networks of the future will prove to be the technical basis for the use of computers, the actual accomplishment of man/computer communication will determine the effectivity of the organizational systems of their use. From this view, investigations of man/computer communication are essential components in addition to the technical and programing technical, computer network conceptions.

Acknowledgement

The authors wish to thank Dr. Trützner and Dr. Tschirschwitz for discussions on these problems.

References

[1] Ahrens, F., and H. Walter, *Datenbanksysteme*, Berlin, De Gruyter, 1971.

[2] Chupin, J.C., Control Concepts of a Logical Network Machine for Data Banks, *Proc. IFIP Congress*, Stockholm, Rosenfeld, J.L. (ed.), Information Processing 1974, Amsterdam, North Holland, 1974.

[3] Deboeck, J.J., User Oriented Modelling for Health Planning and Programming, *Systems Aspects of Health Planning*, Amsterdam, North-Holland, 1975.

[4] Eckert, D., and W. Schönauer, Ein universeller Netzknoten für den Verbund von Hochschulrechenzentren, *Elektronische Rechenanlagen*, 16, 6 (1974).

[5] Fischer, E., BASIC - eine Dialogsprache, *IBM-Nachrichten*, 214, 2 (1973).

[6] Fuchs-Kittowski, K., et al., Mensch und Automatisierung, *Proceedings of the XVth World Congress of Philosophy*, Sofia, Bulgaria, 1973.

[7] Fuchs-Kittowski, K., et al., Zum Gegenstand der Automatisierung körperlicher und geistiger menschlicher Tätigkeiten, *Messen/Steuern/Regeln mit "Automatisierungspraxis"*, 17, 8 (1974).

[8] Differenzierung von Informationen und Konsequenzen für die Gestaltung von Informationssystemen, *Messen/Steuern/Regeln mit Automatisierungspraxis*, 1 (1975).

[9] Fuchs-Kittowski, et al., Informatik und Automatisierung I, (in preparation).

[10] Gluschkow, M.V., Dialogsystem in der Plannung, *Rechentechnik/Datenverarbeitung*, 2, 1974.

[11] Grützner, R., Aufbau von Rechnernetzen und ihre Betriebssysteme, *Rechentechnik/Datenverarbeitung*, (1974).

[12] Herrig, D., and H. Müller, Einfache Sprachen für Menschen und Mittel beim Konstruieren, Manuskriptdruck, 1975.

[13] Jotzloff, R., Über Theorie und Technik von Rechnerverbundsystemen, *Angewandte Informatik*, 9 (1973).

[14] König, D., and J. Schwarz, Sprachvergleich der FORTRAN-compiler, *Online*, 11 (1973).

[15] Koreimann, D.S., *Systemanalyse*, Berlin, De Gruyter, 1972.

[16] McQuillan, Improvements in the Design and Performance of the ARPA Network, *AFIPS Conference Proceedings*, 1972.

[17] Mertens, P., and H. Kress, Mensch - Maschine - Kommunikation als Hilfe für Entscheidungsvorbereitung und Plannung, *Zeitschrift für Betriebswirtschaftliche Forschung*, 1 (1970).

[18] Petkova-Schick, I., Anfragesprache für ein automatisches Fakten-Recherchesystem, Forschungsberichte, 1972.

[19] Rustin, R., *Computer Network*, Englewood Cliffs, New Jersey, Prentice-Hall Inc., 1970.

[20] Seifert, P., Rechnerkopplung - Möglichkeiten und Grenzen, *Rechentechnik/Datenverarbeitung*, 7 (1969).

APPENDIX: LIST OF PARTICIPANTS

CHAIRMAN: Dr. A. Butrimenko
Project Leader
Computer Science Project
IIASA

AUSTRIA

Dr. G. Haring, Institut für Angew. Mathematik und Informationsverarbeitung, Technische Hochschule, Graz.
Dr. H. Kopetz, VOEST Alpine, Linz.
Dr. R. Oberparleter, VOEST Alpine, Linz.
Dipl.Ing. F. Oismüller, Bundesversuchs- u. Forschungsanstalt, Vienna.
Dipl.Ing. Dr. A. Sethy, Bundesversuchs- u. Forschungsanstalt, Vienna.
Prof. Dr. R. Eier, Institut für Datenverarbeitung, Vienna.
Dr. H. Bodenseher, Computer Centre, Technische Hochschule, Vienna.
Dr. P. Hofbauer, Academy of Sciences, Vienna.

BELGIUM

Mr. Banh Tri An, University of Liège.
Prof. A.S. Danthine, University of Liège.

FRANCE

Mr. J. Le Bihan, Reseau CYCLADES, Rocquencourt.
Mr. G. Sergeant, Institut National Polytechnique de Grenoble.

FEDERAL REPUBLIC OF GERMANY

Dipl.Ing. H. Bender, Institut für Informatik III, Universität Karlsruhe.
Dr. K.D. Günther, Gesellschaft für Mathematik und Datenverarbeitung, Institut für Datenverarbeitung, Darmstadt.
Dr. P. Haas, Softwarelabor, München.
Dr. F. Hossfeld, Zentralinstitut für Angew. Mathematik der Kernforschungsanlage, Jülich.
Dipl.Phys. W. Lehmann-Bauerfeld, Hahn-Meitner Institut, Berlin.
Dipl.Phys. H.W. Strack-Zimmermann, Hahn-Meitner Institut, Berlin.
Dipl.Ing. F. Vogt, Hahn-Meitner Institut, Berlin.

GERMAN DEMOCRATIC REPUBLIC

Prof. K. Fuchs-Kittowski, Humboldt University, Berlin.

HUNGARY

Dr. P. Bakonyi, Computer and Automation Institute, Budapest.
Mr. A. Gyarfas, Computer and Automation Institute, Budapest.
Mr. T.I. Szentivanyi, Infelor Systems Engineering Institute, Budapest.

ITALY

Mr. L. Lazzori, CNUCE, Institute of CNR, Pisa.

NETHERLANDS

Dr. G. Megman, Rijswijk.

POLAND

Dr. I. Maronski, Institute for Organization, Management and Control Sciences, Warsaw.
Mr. A. Radziminski, Institute for Organization, Management and Control Sciences, Warsaw.

SWITZERLAND

Dr. P. Schicker, Rechenzentrum Eidg. Technische Hochschule, Zürich.

UNION OF SOVIET SOCIALIST REPUBLICS

Mr. Zadorozhny
Mr. Panjukov

UNITED KINGDOM

Mr. D.L.A. Barber, National Physical Laboratory, Teddington.
Dr. D.W. Davies, National Physical Laboratory, Teddington.
Prof. P.T. Kirstein, Department of Statistics and Computer Science, University College London.
Mr. N.H. Shelness, Edinburgh Regional Computing Centre, University of Edinburgh.

UNITED STATES

Dr. V.G. Cerf, Digital Systems Laboratory, Stanford University, California.
Mr. E.C. Hendricks, IBM Cambridge Scientific Center, Cambridge, Massachusetts.

<u>IIASA</u>, Computer Science Project

Dr. A. Butrimenko
Dr. J. Sexton
Mr. V. Dasko
Dr. Y. Masunaga
Dipl.Ing. U. Sichra
Mr. Klaus Gradischnig
Mr. W. Orchard-Hays