IFIP
TC–6

COMNET '77
BUDAPEST
OCTOBER 3–7. 1977

Working papers, Vol. I.

COMPUTER NETWORKS AND TELEPROCESSING SYMPOSIUM

NJSZT

FEDERATION OF TECHNICAL AND SCIENTIFIC SOCIETIES
JOHN v. NEUMANN SOCIETY for COMPUTING SCIENCES

IFIP
TC–6

COMNET '77
BUDAPEST
OCTOBER 3–7. 1977

Working papers, Vol. I.

COMPUTER NETWORKS AND TELEPROCESSING SYMPOSIUM

NJSZT

FEDERATION OF TECHNICAL AND SCIENTIFIC SOCIETIES
JOHN v. NEUMANN SOCIETY for COMPUTING SCIENCES

IFIP
TC–6

COMNET '77
BUDAPEST
1977. OKTÓBER 3–7.

SZÁMÍTÓGÉPHÁLÓZATOK TÁVFELDOLGOZÁS SZIMPOZIUM ELŐADÁSOK

I. kötet

NJSZT

MTESZ
Neumann János
Számítógéptudományi Társaság

Felelős szerkesztő: Szentiványi Tibor
Felelős kiadó: Károlyi Antalné
Eng.szám: 37851

SZÜV Nyomda 77,2384 – Felelős vezető: Mihályi Zoltán

COMNET '77

EUROPEAN SYMPOSIUM
ON DATA COMMUNICATION
OCTOBER 3–7, 1977
BUDAPEST

Organized by:

John v. Neumann Society for Computer Sciences
in cooperation with
The Hungarian Academy of Sciences, and the
Scientific Society for Measurement and Automation,
and the
Scientific Society for Telecommunication

Sponsored by:

International Federation for
Information Processing (IFIP)

Program Committee

Chairman: Szentiványi, T. (Hungary)

Bakonyi, P.	(Hungary)	Lukács, J.	(Hungary)
Bazewicz, M.	(Poland)	Meier, H. W.	(German D. Rep.)
Boyarchenkov, A.	(Soviet Union)	Németh, J.	(Hungary)
Butrimenko, A.	(Austria, IIASA)	Porižek, R.	(Czechoslov.)
Davies, D. W.	(United Kingdom)	Pouzin, L.	(France)
Gergely, Cs.	(Hungary)	Pužman, J.	(Czechoslov.)
Gvozdjak, I.	(Czechoslovakia)	Seidler, J.	(Poland)

Symposium Committee

Chairman: Gergely, Cs.

Bakos, T.	Kovács Gy.
Bogdány, J.	Mazgon, S.
Bohus, M.	Németh, P.
Csánky, L.	Sima, D.
Gulyás, J.	Uhereczky, L.
Kiefer, J.	Ujvári, Z.
Kiss, É. (Mrs.)	

Publication Committee:

Chairman: Tóth B.

Dobrovolny, T.
Ivanyos, L.
Károly, K. (Mrs.)
Vágner, Gy.

Companies supporting
the Symposium

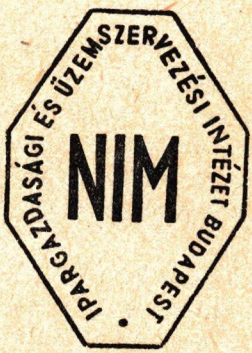
IRODAGÉPTECHNIKA V.

ÉGSZI

SZÁMKI

= SZÁMGÉP =

DATORG



SZÁMOK

ZÜV

VT TV VIDEOTON
rádió - televizió

PREFACE

COMNET '77 is the first international symposium in Central Europe dealing with teleprocessing and computer networks under sponsorship of IFIP TC–6. The John von Neumann Society for Computing Sciences organizes in its already traditional series of conferences Computertechniques taking place every year since 1968, the Symposium COMNET '77 dealing this time with an outstanding topic data communication.

The themes of the Symposium comprise the fields regarding design, implementation and use of networks realized by interconnecting computers. As the principles of the packet switched networks have been based on the experiences of multi-access time-sharing systems and of teleprocessing, and have been developed further from them, therefore also an overview about these fields is necessary to make the entire picture complete.

The aims of the Symposium are: presentation of results achieved in the field of teleprocessing till now, discussion of questions relating to design and development of national and/or international networks about to be realized.

The papers delivered by the members of the IFIP TC–6 (Data Communication) Committee give a unique feature to the Symposium. All the – more than fifty – lectures comprise results of work and experimentations relating to problems of packet switching, flow control, control languages of networks, protocols, development of networks and teleprocessing systems and use of several known networks.

In connection with the last item, among others, an expounding of CYCLADES, EIN, IIASA-net, TRANSPAC, ALOHANET and DATAPAC networks will take place.

Papers have been classed in the three following groups:

- *papers of tutorial character giving a detailed explanation of each subject,*
- *comprehensive, survey type contributions, expounding a given system, experiences gained from it,*
- *papers demonstrating results relating to some narrower field will be classed in a special category.*

The two volumes of the proceedings contain practically all papers to be delivered during the conference. In the case of several papers that arrived after the deadline for papers, only the abstracts have been published.

The sequence of papers seen in the proceedings has been determined only by typographical reasons. Their quick finding, however, is facilitated by an alphabetic list containing every author.

The IFIP TC–6 Committee assumed the sponsorship in the organization of the Symposium. We must not fail to acknowledge here the services of Mr. Pouzin, L. chairman of the Committee and Mr. D. W. Davies who have to be especially mentioned for sparing no pains in contributing to prepare COMNET '77.

The enormous work done by the international program and preparation committee cannot be appreciated without which the organization of COMNET '77 would have been impossible.

In the end, acknowledgements are due to the authors that have contributed to the edition of the proceedings in a high-level and convenient form. The publication of the proceedings is a result of the ardous work done by Messrs. L. Ivanyos, B. Tóth and Gy. Vágner in the course of editing.

September 1977

Tibor Szentiványi
Chairman, Section of Computer Techniques
John von Neumann Society

CONTENT

Papers marked with * appear in the second volume

FLOW CONTROL AND CONGESTION CONTROL

Donald W Davies

National Physical Laboratory of the United Kingdom

Abstract

Three schemes for flow control are compared. The effect of congestion on the transit delay and flow capacity of a network is discussed and control of congestion by means of entry permits (isarithmic control) is described. The possible lock-up mechanisms are described and the method to prevent them by buffer classes. It is postulated that congestion and lock-up are related. Rules are stated which have been found to prevent congestion. Multi-level networks need special measures.

1. FLOW CONTROL AND CONGESTION CONTROL

The concepts of flow control and congestion control must be distinguished.

Congestion in a network is a state produced by heavy traffic in which the various flows are interfering with each other. The result of congestion is that traffic flows which, by themselves, would be able to pass freely through the network are delayed or prevented by the general overload.

In order to give the concept of flow control a definite meaning we shall restrict it to a single "communication path" through the network. Nearly all communication concerns two communicating parties which may be user terminals, software entities within a computer or intelligent terminals. For the purpose of communication these entities are linked either conceptually or by actual hardware. Moving from "hard" to "soft" the link may consist of

(1) a circuit such as a leased line or a call in a circuit switched network.
(2) a virtual call or permanent virtual circuit.
(3) a "liaison" between transport stations in a datagram network.

The absence of flow control is illustrated by a typical connection from a computer to a teletype. The computer sends characters at a rate which it believes the teletype can absorb, allowing a small delay or inserting dummy characters to accommodate the carriage return. Suppose now that in the course of printing a table of figures the operator at the teletype wishes to halt the flow momentarily to adjust the paper. In some systems, any action at the teletype end which causes it to stop receiving characters means that the characters still being sent by the computer are lost.

On the other hand consider a system in which characters are held in buffers until the next buffer in the path is ready to receive them. Then if the teletype ceases to accept characters the buffers fill up successively back along the path (like cars in a road network) until the

sender is stopped. When the receiver releases the flow it will continue with no consequent error.

The essence of flow control is that the receiver of data has a means of shutting down the source. This is a kind of negative feedback, and it functions best if the 'throttling' acts quickly across the entire network.

The analogy with road traffic is useful when we consider congestion. Any shared resource may be subject to overload. Public communication networks are economic because the users share the lines and switches, which they do not all need at the same time. When the traffic goes beyond its capacity the network overloads and traffic is delayed or prevented. Congestion control cannot prevent this but it can ensure that under overload

(1) the available capacity shared equitably among the users.
(2) the available capacity is fully used.

It is difficult to make the first condition precise in data networks where traffic of so many different kinds is carried but practical networks demonstrate that no user need be entirely deprived of service and that, by careful design, low-rate, fast-response users can still operate when users requiring heavy flows have to be restricted.

Concerning the second condition, we must guard against the occupation of useful traffic capacity by the flow control measures themselves and, most importantly, against the kind of congestion which reduces the total useful capacity of the network - the kind of thing we are familiar with in snarled up road traffic.

2. FLOW CONTROL

In store and forward systems the data is carried in blocks which may be bytes, frames, packets or messages. The transfer of the blocks from one part of the system to another is governed by data-link protocols which can administer flow-control, error-control, and sequence-control. The location of these "data-links" in a system can be chosen in many different ways.

Figure 1 represents a path through a network between two communicating entities X and Y. It involves two "customer-network" interfaces X - a and Y - d and a number of links which are internal to the network, shown as a - b - c - d.

We can begin by thinking of each of these links as separately being provided with flow control, a situation which is found in the virtual circuits of the French Transpac network. Now if the flow of traffic from X - Y is halted at its destination at least one buffer will fill at d with a data unit unable to be sent to Y, resulting in the flow control mechanism c - d halting the data unit at c and so forth until buffers are filled along the whole path. The internal design of the network determines how many buffers are filled and, speaking from experience, the designer may find by an experiment such as we have described (in which the flow at Y is stopped) that he has much more data held up in the network than he expected. This is not a good feature in data networks. If the receiver of the data at Y has halted it to make a change to the process being carried out at X the change may come much later in the stream than he expected. To avoid this, in almost all communication situations between intelligent devices an additional protocol is introduced between the ends, and "end-to-end" protocol which then limits the number of units (generally packets) in transit.

In many virtual circuit networks the step-by-step flow control within the network is replaced by end-to-end control, shown in the second scheme of figure 1. The links a - b - c - d are replaced by a protocol operating a - d. This may allow the packets to travel with adaptive routing. The protocol restores the sequence, corrects errors and applies flow control so that the number of packets in flight through the network is just about right to "fill the pipe". If only one packet was allowed to leave the source until an acknowledgement had been received from the destination it would greatly reduce the flow. If the number of packets allowed in flight (the "window" in typical protocols) is increased, beyond a certain limit the flow does not increase. This is colloquially known as "filling the pipe". The operation of a "window mechanism" from end-to-end of the network gives a much better negative feedback than having many link protocols in tandem. The aim should be

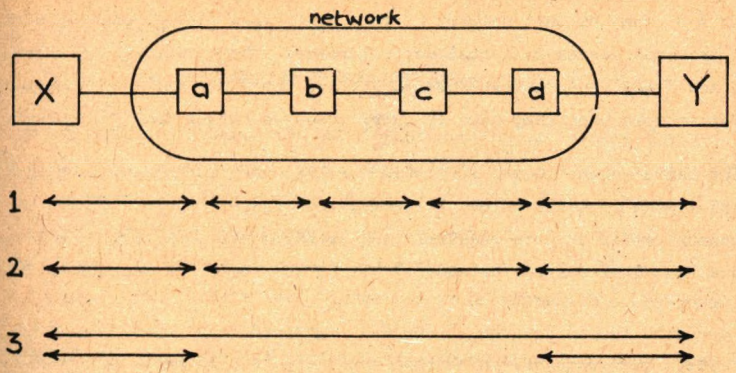


FIGURE 1 Schemes for flow control

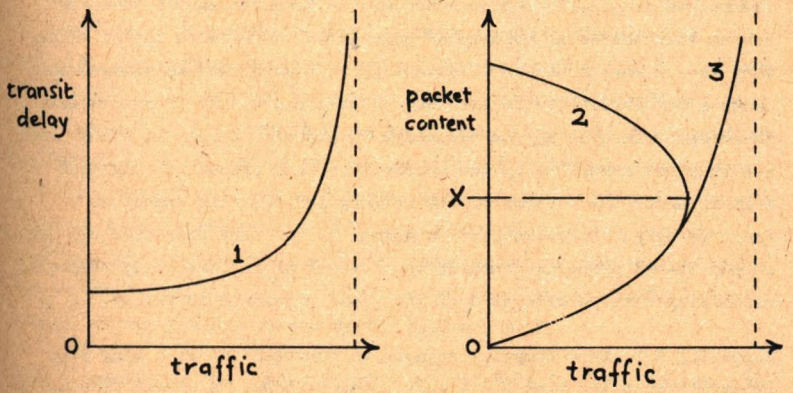


FIGURE 2 Network saturation characteristics

to have just sufficient packets in flight to achieve the required flow.

In a virtual circuit network of this kind, flow control is completed by a protocol across each customer interface. The communicating entities X and Y may nevertheless have to provide their own end-to-end protocol, for example when they need to carry messages larger than the packet.

The third scheme showed in figure 1 relies much more heavily on this end-to-end protocol which, it seems, the majority of intelligent communicating devices require. Acknowledgements pass back from Y to X and the "link control procedure" limits the number of packets in transit.

Because each part of a network is handling traffic for many different flows the entry of packets into the network has to be controlled and this is the function of the additional flow protocol shown over the link X - a. The corresponding link at the destination end is necessary when Y is a source of data but its function is less clear for Y as destination. It ensures that the inability of Y to receive packets results in a queue of undeliverable packets at d and the number of these should be limited by the end-to-end protocol.

The second and third schemes overcome the disadvantage of the first scheme that the total hold-up of packets in the network may be large. They have a corresponding disadvantage that when flow is stopped more packets may arrive at the exit node d than can easily be accommodated. In extreme cases it may be necessary to discard packets at d and these are then retransmitted by the appropriate link procedure. In case 2 this is under the control of the network and for this reason it is favoured by public communication authorities. They fear that the use of an end-to-end procedure outside the network as in case 3 may result in discarding many packets at d if the users procedure is faulty.

Supposing X to be a computer engaged in conversation with many other terminals then the link X - a carries many flows. A single flow control procedure over this link would not be sufficient because if any of the flows were stopped it would stop all the others. For this reason in a customer interface like X25 a flow-control procedure is set-up for each

of the "logical channels" which is operating. In the X25 interface there is also a "data link procedure" operating over the link as a whole for the purpose of error control.

3. THE EFFECT OF NETWORK CONGESTION

Our understanding of congestion and its prevention is largely derived from simulation experiments. Operating networks do not lend themselves to the experiments with different procedures and priority schemes that we need to carry out. The behaviour of networks has been found to depend on details of the procedures operating over each link and at each switching point therefore a large number of experiments has been needed to gain some understanding. The model employed in our work was a detailed one, down to the individual demands on the operating system of the nodal computers. Models at this degree of detail are beyond the reach of analytic queuing theory methods.

Figure 2 shows the result obtained for the mean transit time of a packet as a function of network traffic, for uncongested conditions only. It is typical of systems with many queues in tandem that their behaviour is like that of a single queuing system.

The right-hand part of figure 2 shows a different method of exploring the behaviour of the complete model network in which the total of packets in the system is plotted against the level of traffic. In both cases, the traffic applied was not uniform. The assumed traffic matrix was multiplied by a factor in order to produce a variable total traffic. In order to plot the second curve, the total of packets in the system was held constant by allowing a single packet to enter whenever one was delivered to its destination. The source-destination pair for an entering packet was determined by the same random process that was used for the other simulations but the volume of traffic (the traffic matrix multiplier) was now determined by the system itself.

Curve 2 represents the behaviour of a system which is badly designed for congestion control. As the number of packets in the system increases the flow eventually decreases. Though users demands reduce when the delay increases, the effect of the congestion shown in curve 2 would

probably be to fill the network and stop flow entirely.

This behaviour is typical of road traffic. If "vehicles per mile" is plotted against "vehicles per hour" a similar characteristic is obtained. The same flow along a road can be carried at one point by freely flowing, fast traffic and at another point by a slowly moving queue. The aim of good network design is to obtain the shape of curve 3, but even with this characteristic there is not much to be gained by pushing far into saturation with increasing delays at no great gain in throughput. A good network design will reject incoming traffic so that delay and hold-up is limited.

When the congestion of curve 2 was first observed it was thought to be due to the filling of available buffer space in switching nodes. It would have been understandable that, with full queues nearly everywhere, the opportunity for packets to move would be restricted. When the total number of packets in the system was examined it was found that the average queue occupancy was not large. The real causes of the congestion will be discussed later.

4. ISARITHMIC FLOW CONTROL

The congestion characteristic of curve 2 suggested that congestion could be controlled by holding the total content of the network below the critical value shown as X in the figure. This would prevent traffic from entering if the packets as yet undelivered exceeded a certain figure. It would function in relation to the whole network rather in the way that flow control functions for a single path. Control over the total number of packets in the system was given the name "isarithmic" from the greek words meaning "constant number".

Even if isarithmic control could be achieved there might be doubts about the effectiveness, since it only controls the total number of packets and not their distribution. But simulation experiments showed that, with high probability, congestion was indeed prevented if the total number of packets was held within limits. This might be expected for a well-connected network but a network consisting of two parts with a "bottle-neck" between might be different.

The method proposed to control the number of packets was to introduce into the network a fixed number of "permits" and to allow the packet to enter the network only if it could acquire a permit. It was necessary to decide how to redistribute the permits from packets which had arrived at their destination. A number of different schemes were tried but the simple one in which any free permit was sent at random to a neighbouring node was found to be as good as any.

If the flow in the network is reasonably balanced – the flow originating at each node roughly equals that destined for the node – the redistribution of permits would be minimal. In practice some redistribution would be needed because a concentration of computer bureau would tend to be the source of more packets than it was the destination.

Permits in transit between nodes do not need to be carried in individual packets. By equipping a data packet with a small field in its header which specifies that it carries a number of free permits, most permit traffic can be accommodated and when no data packets are flowing there are usually administrative packets which would serve, or as a last resort special packets could be sent to carry the permit flow.

A refinement of the permit redistribution scheme proved to be useful. This was to allow a certain number of permits to come to rest at each node and to be available immediately for source traffic at that node. This "permit queue" has a finite length and any packets arriving beyond that length would become migrant permits until they are either found a data packet which needed them or a permit queue with space in which they could rest.

To tune this "permit queue" mechanism, simulations were made with varying queue sizes. The number of permits used in the model was five for each of the eighteen nodes in the network. Figure 3 shows how the admission delay varied with the maximum size of permit queue. Permit queues should be of sufficient size to accommodate half of the total content of permits in the network.

The results of simulating isarithmic flow control can be summarized as follows. The method is satisfactory. Redistribution of permits presents no problem. For any flow up to approximately 80% of saturation the time spent waiting for a permit is very small.

There is a practical difficulty which might inhibit the use of isarithmic congestion control. A maximum number of packets in the system is set by loading a number of permits when the network is set-up. Any malfunction which destroyed or created permits could not be stopped quickly enough to prevent the collapse or overload of the network. There were proposals to provide an automatic control of the total number of permits in the system but these were not pursued because, at the time, the need for this isarithmic system was doubted.

The reason for beginning the investigation of isarithmic control was the belief that a characteristic such as curve 2 of figure 2 was an inherent property of packet-switched networks. It later became clear that characteristics like curve 3 could be achieved by attention to the details of network design and for these networks, isarithmic control was unnecessary.

For any link within a network in which the total number of packets is controlled, for example by a window mechanism, there is a kind of local isarithmic congestion control. Whether this will be effective depends on how many such protocols are in operation at one time and whether the number of packets thus admitted in the maximum case exceeds the level at which congestion begins. For the type of network which gave congestion similar to curve 2 this number was small, typified by the five packets per node used in the isarithmic study. For a virtual call system of the order of one thousand terminals per node the number of calls in operation at the peak hour would probably cause congestion irrespective of any window mechanisms. The key to prevention of congestion lies elsewhere.

5. STORE AND FORWARD LOCK-UP

The type of congestion represented in figure 2 occurred at much lower packet contents than sizes of queues in the networks led us to expect.

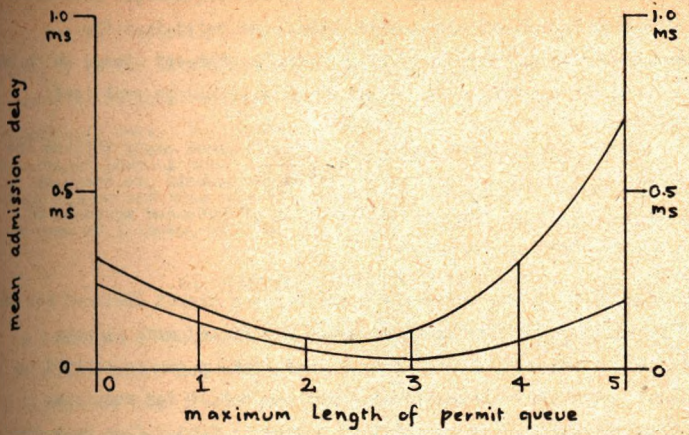


FIGURE 3 Admission delay in isarithmic control

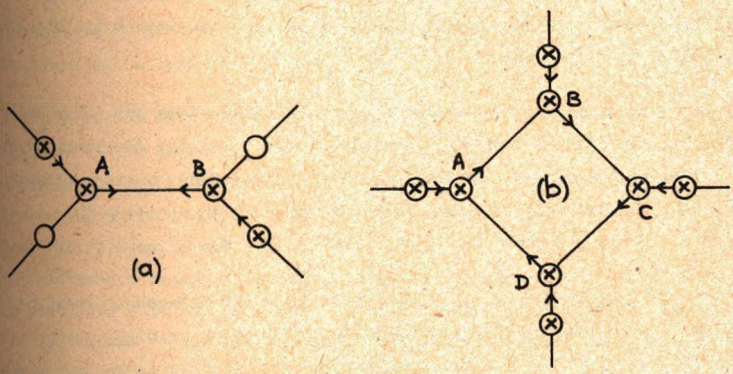


FIGURE 4 Lock-up situations

We therefore stopped our simulation runs in congested conditions and examined the state of all the queues. This indicated that the congested network was very close to certain "locked-up" states which had been observed in the ARPA network and which will now be explained.

Figure 4a shows the simplest case. In these figures empty buffers are shown by empty circles and buffers occupied by packets are shown by circles with crosses in them. The direction of intended movement of each packet is shown by an arrow.

In figure 4a each of the two nodes A and B has a packet destined for the other. If all the buffers in these nodes are filled with packets destined to the other, none can move. This lock-up can be avoided by reserving in each of these nodes at least one buffer for receiving a packet from the other. We expect this packet to find a free exit from the pair.

In the case of figure 4b, a cycle A B C D exists in which each node has packets in its buffers destined for the next one in the cycle. Packets coming into the cycle cannot find any space and packets in the cycle cannot move. It is more difficult in this case to provide buffer allocations to prevent lock-up. A mesh network has within its structure cycles of various sizes.

Figure 5 illustrates that congestion of this type does not occur when there are no cycles. The bottom nodes A and B are the destinations of all packets, which move steady down the diagram from the top. Assume that all buffers are full initially but some of those above the confluence are destined for A and some for B. Now if packets are taken from both the destinations, one by one, there is always a possible movement of packets to buffers lower in the diagram which will free one of the sources. Any "fair" method of accepting packets where they converge will result in giving some part of the capacity to each source. The flow to A, if stopped, may stop the flow to B but this is a different phenomenon, and it clears if A starts to accept packets.

The "post mortem" investigation of congestion showed that congested states were similar to the locked-up configurations we have described.

In some cases the "near lock-up" could remain for a long time such as a second before dispersing. The simple lock-up of figure 4a had been avoided by buffer allocation and that of figure 4b was not very likely in practice, so total lock-up of any part of the network was comparatively rare. We formed the hypothesis that the existence of these "rare" lock-up conditions showed itself by a succession of near lock-up states and this was the cause of congestion. Any scheme which would prevent lock-up, should, on this hypothesis prevent congestion.

6. SHOULD PACKETS BE DISCARDED

It is sometimes proposed that a network should attempt to clear a lock-up or reduce local congestion by discarding packets. This is based on the assumption that a user protocol (or a network end-to-end protocol) will detect the loss and retransmit. But the effectiveness in removing lock-ups and reducing congestion is doubtful.

In a datagram network the absence of flow control within the network can cause a problem if packets cease to be accepted at their destination. Discarding may be necessary. Such an event is due to a failure of the user's protocol. If the protocol is correctly formulated but the processors fail, at most a few packets will arrive at a closed destination and a timeout in the link protocol should deal with these.

If several communicating devices are multiplexed to form a single subscriber to the network, the failure of one of those to receive packets should not be allowed to block the others. The multiplexer (private branch exchange or local network) must therefore accept all packets and discard those it cannot deliver.

An incorrect or wrongly operating protocol will be very rare. Schemes have been proposed to shut off the source if a large proportion of its packets are undeliverable, but the concensus of opinion is that such facilities are unnecessary.

Public network designers have adopted the virtual circuit method and one of the reasons stated for this choice has been the problem of the undeliverable datagram. They are particularly concerned not to charge

their subscribers for packets that were discarded by the network – for whatever reason. In the author's opinion this is making too much of an unimportant problem.

The virtual circuit presents the same problem in a different way. Call request packets are datagrams and can be undeliverable. Public network authorities intended not to charge for call set up, but X25 now allows the call request to carry up to 16 bytes of user data and the call clear packet to carry one byte. These packets must be charged for or users will obtain free communication capacity by refused calls. When a virtual circuit shares its buffers with other circuits there are possibilities of congestion, as for datagrams. Then if the user's protocol fails some virtual circuit networks can accept and charge for packets until many internal buffers are full and then discard them when the call is cleared – little different from the datagram network. In the public circuit-switched network nobody is concerned that the users may lose data and pay for service when their own protocols fail.

The conclusion we reach is that a public network could, without difficulty, operate both datagram and virtual circuit facilities in a way that users would accept. Discarding packets is a little used safety mechanism, not a congestion control policy and it penalises only those users who operate a faulty protocol. For a large part of the short transaction traffic of a future network, datagrams would prove simpler and more economic to operate than virtual calls.

7. LOCK-UP PREVENTION BY BUFFER CLASSES

The avoidance of lock-up of the simplest kind, shown in figure 4a, was achieved by introducing two classes of buffer in the adjacent nodes. Packets coming into this system from the outside could enter buffers of class 1 of which there was a restricted number. Packets coming from the other node of the pair have a higher priority and can use buffers of classes 1 and 2. Only one buffer in each node need be of the reserved class, i.e. class 2.

A similar scheme could be applied to the problem of lock-up in figure 4b. If it was thought that packets might visit each one of the four nodes in

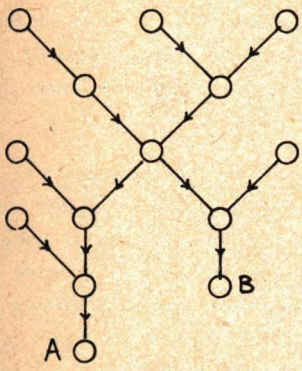


FIGURE 5 Non-Lock-up situation

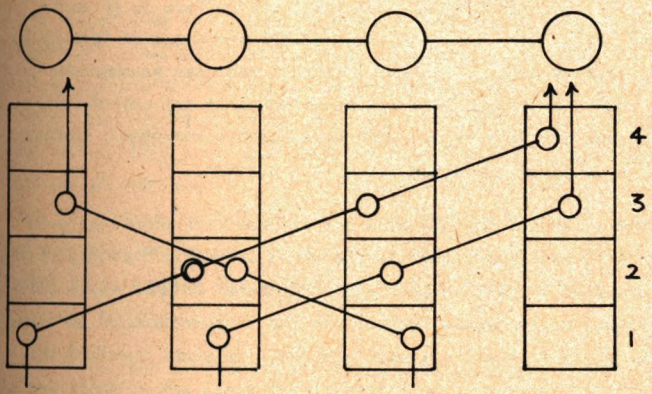


FIGURE 6 Buffer classes

the ring, then four buffer classes would be introduced. Packets coming from the outside use only buffer class 1. When they pass to the next node in the ring they use classes 1 or 2. Passing to the third node in the ring they could use classes 1, 2 or 3. Finally, in reaching the fourth node on their journey they would have access to any of the buffer classes in these nodes.

Figure 5 suggests the extension of this principle to a network with any topology. Lock-up is impossible in a network with the form shown there because progress is always down the page and no cycles can be found. Access to buffer classes is controlled by the number of nodes a packet has visited on its journey. The progression through buffers of different classes then follows a pattern like that in figure 5. Figure 6 shows how this works out for four nodes connected in a line. The packets are shown entering and leaving different points and sometimes sharing buffer classes but the resultant "overlaid network" is similar to that of figure 5.

Note that packets of class N are not confined to buffers of class N. They may use buffers of class N or any inferior classes such as N-1, N-2 etc. The availability of the high level buffers is to avoid conflict. It is also true that the logical avoidance of lock-up is achieved even if each buffer class in each node can accommodate only one packet. Buffers of any class can be attached to any output queue. The number of buffers required in each node, with a reasonable routing scheme, is measured by "diameter" of the network.

The method is easily administered but becomes less practical with networks of increasing size. In the realm of public networks it would not be practical at all because each additional network connected to the international scheme would necessitate the introduction of an extra range of buffer classes in all switching nodes. We might question the possibility of lock-up on an international scale but we shall see later that the possibility of such lock-up in a hierarchical network cannot be ignored.

8. PRAGMATIC RULES FOR LOCK-UP PREVENTION

Because the strict avoidance of lock-up by buffer allocation is limited in practice to small networks we should look for simplified schemes that work nearly as well. Early in our simulation work we arrived at two rules which, in all our experiments, avoided congestion under conditions of overload.

Traffic which is about to leave the network should be given priority in getting through to its destination because it will leave space for more traffic to flow. When a link was blocked by congestion, packets waiting for transmission across that link were examined to see if their destination was the other end of the link. If so, they were accepted and delivered to their destination without occupying any of the buffer space allocated to queues for output from that node. This special priority over the final link to a destination helps to clear traffic from the network. There is an associated rule for routing such packets. Adaptive routing is suspended and they must go directly to their destination. This avoided any possibility of packets circling or "ping-ponging" in nodes adjacent to their final destination.

Figure 7 illustrates the second rule which helped to avoid congestion. The traffic in a switching node can be divided into three kinds. The traffic with the highest priority is the destination traffic already discussed. The second priority is given to transit traffic which is passing through this node on its way to a destination. Since the network has already accepted this traffic it makes sense to help it on its way. The third and last priority goes to traffic which originates at the node and has not yet begun its journey. By giving last priority to source traffic, when the network approaches overload it will restrict the incoming traffic.

Figure 7 shows that the second and third priority traffic (source and transit traffic) competes for buffer space in the queues for transfer to other nodes. These priorities are treated like the buffer classes described earlier. The amount of space in each queue accessible to source traffic is restricted but transit traffic can use any part of the

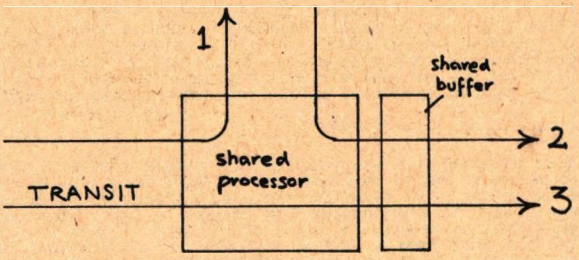


FIGURE 7 Priorities of traffic at switch

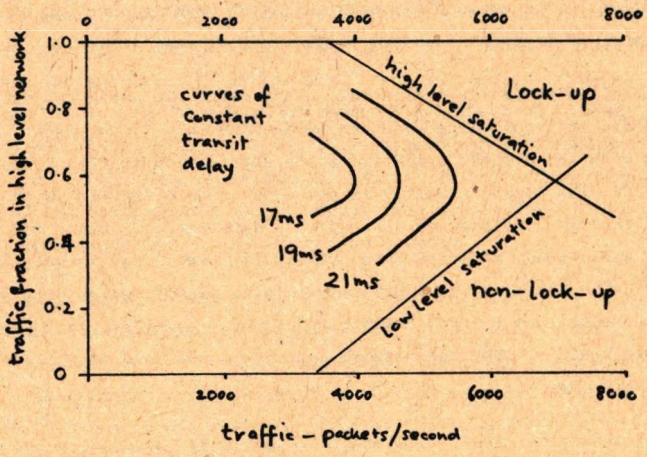


FIGURE 8 Saturation of two-level network

queues. As before, it is only necessary to reserve space for one packet in each queue solely for transit traffic.

The high priority accorded to destination traffic becomes important when the main limitation at the node is processing power rather than capacity of output links. In this case, the operating system should give priority to processes concerned with destination traffic and then next to transit and source traffic in that order. Our own simulations did not have such limitation of processing power.

With the aid of these priority rules, congestion was no longer a problem in the network we studied and it was concluded that the isarithmic scheme, though successful, would not in practice be required. This conclusion was to some extent reversed in the later study of hierarchically connected networks.

9. HIERARCHICAL NETWORKS

It is conjected that a network of one hundred nodes or more would be more efficient if designed as a number of smaller networks connected together by a "super network" of higher speed trunks.

One reason for hierarchical construction is the size of routing tables. In a hierarchical network, packets destined to a distant low level network are transferred to the high level network and are sent through to the appropriate exit. Only the routing table for the "super nodes" need be stored in the high level network. The eventual route to the destination is then determined by local routing tables in one of the lower level networks.

Another reason for hierarchical construction is that great economies in transmission come from using lines of higher transmission capacity. The hierarchical structure concentrates the traffic on to main trunks and provides this economy.

Our simulation studies employed a 50 node model containing five lower level networks each of nine nodes and nine interconnected "super nodes". The routing tables were somewhat arbitrary, based on minimum number of

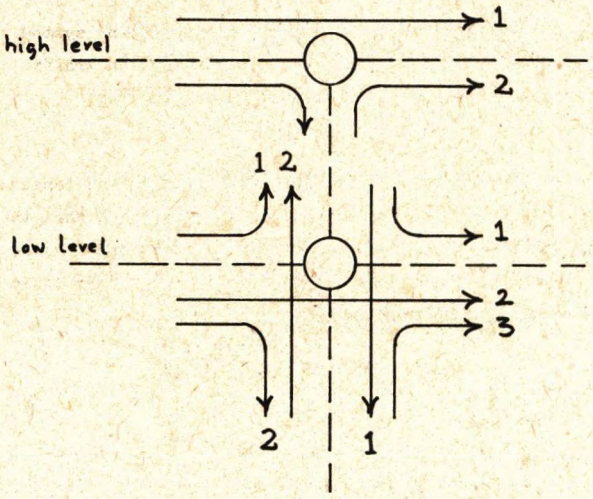


FIGURE 9 Buffer priorities in two-level network

8. PRAGMATIC RULES FOR LOCK-UP PREVENTION

Because the strict avoidance of lock-up by buffer allocation is limited in practice to small networks we should look for simplified schemes that work nearly as well. Early in our simulation work we arrived at two rules which, in all our experiments, avoided congestion under conditions of overload.

Traffic which is about to leave the network should be given priority in getting through to its destination because it will leave space for more traffic to flow. When a link was blocked by congestion, packets waiting for transmission across that link were examined to see if their destination was the other end of the link. If so, they were accepted and delivered to their destination without occupying any of the buffer space allocated to queues for output from that node. This special priority over the final link to a destination helps to clear traffic from the network. There is an associated rule for routing such packets. Adaptive routing is suspended and they must go directly to their destination. This avoided any possibility of packets circling or "ping-ponging" in nodes adjacent to their final destination.

Figure 7 illustrates the second rule which helped to avoid congestion. The traffic in a switching node can be divided into three kinds. The traffic with the highest priority is the destination traffic already discussed. The second priority is given to transit traffic which is passing through this node on its way to a destination. Since the network has already accepted this traffic it makes sense to help it on its way. The third and last priority goes to traffic which originates at the node and has not yet begun its journey. By giving last priority to source traffic, when the network approaches overload it will restrict the incoming traffic.

Figure 7 shows that the second and third priority traffic (source and transit traffic) competes for buffer space in the queues for transfer to other nodes. These priorities are treated like the buffer classes described earlier. The amount of space in each queue accessible to source traffic is restricted but transit traffic can use any part of the

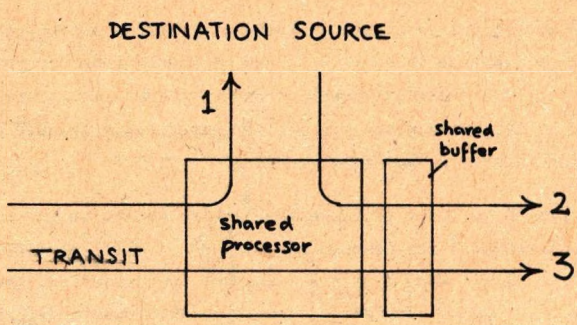


FIGURE 7 Priorities of traffic at switch

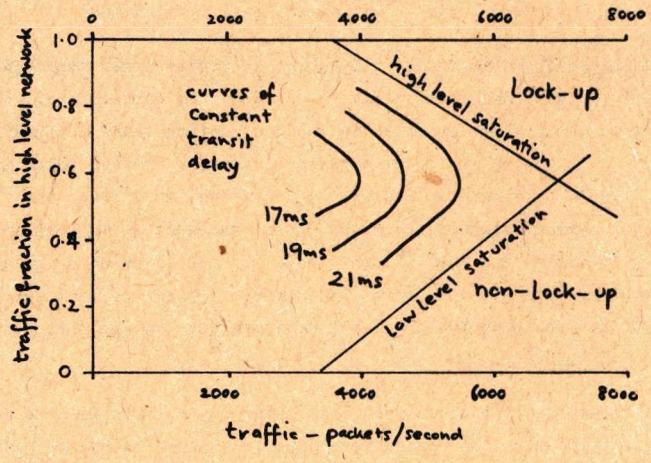


FIGURE 8 Saturation of two-level network

NETWORK INTERCONNECTION AND PROTOCOL CONVERSION

Michel GIEN

IRIA - B.P. 105
78150 Rocquencourt, France

Abstract

Now that computer networks start proliferating among private as well as public organizations, the question arises of connecting these together, so that users of one network can get to resources and services of the others. The development of standards, when they are designed to take into account "internetworking", will help to solve the problem. Unfortunately, networks often grow on their own and "gateway" functions are to be necessary to take care of interface conversions between two adjacent networks. The level at which gateway functions should operate, when connecting networks with a layered architecture, is analysed here. The internetwork addressing problem and its implications on routing is touched and a simple case example of a network interconnection between EIN and CYCLADES networks is presented.

1. INTRODUCTION

Since the past few years, computer networks have been seen emerging from a wide variety of bodies and organizations. Starting with research networks to experiment concepts and new technologies, computer networks are now being built, on a much wider scale, by private or public, national or international organizations, as every day tools for their operation (private networks), or to provide new services to their customers (public data networks).

The wish to interconnect computer networks, already expressed with the research networks, will inevitably increase with the proliferation of newcomers. Unfortunately, networks are built to provide similar services but usually accessed through interfaces which are most of the time different, and indeed incompatible. They use technologies optimized for their own environment, e.g. a private network can be built according to an overall architecture whereas a public network design will be optimized mainly for solving data transmission problems only. Most of the time, in any case, very little thought is put on internetworking problems.

Some of these problems will be analysed here with a tentative classification. Solutions will claim for the introduction into network architectures of standard levels of services, where a "bridge" between networks can be established.

2. THE NETWORK INTERCONNECTION PROBLEM

2.1. Global internetworking

Networks are often built independently from each other. Each local net has its own internal characteristics, its own addressing conventions, routing algorithms, subscriber interfaces, packet formats, internal services and tariff policies.

Subscribers or hosts on these networks come from different manufacturers. They use various and generally incompatible operating systems. They have different conventions for handling their terminals, files, other devices and also processes that one would like to distribute over one, and even several, computer networks. They now have their own internal network architecture for handling remote terminals, or terminal concentrators.

Trying to insure global inter-working between all these elements of computer networks e.g. give a mutual access between any host and any terminal, does not appear to be a trivial task, even when they are connected to the same network. Computer networks, and even big systems, made of the assembly of homogeneous processors, terminals and other components, can be looked at, as completely distinct worlds that one wants to make interwork.

"Gateway", in this context, is the name usually given to the processors, pieces of software, hardware black-boxes, etc..., which are placed between two networks to act as a bridge or interface convertor, in order to allow interworking between their elements.

One can easily imagine how many of these conversion functions would be needed and how complex one gateway would be, to allow global interworking between two and even several computer networks, made of such independent elements (see Fig. 1).

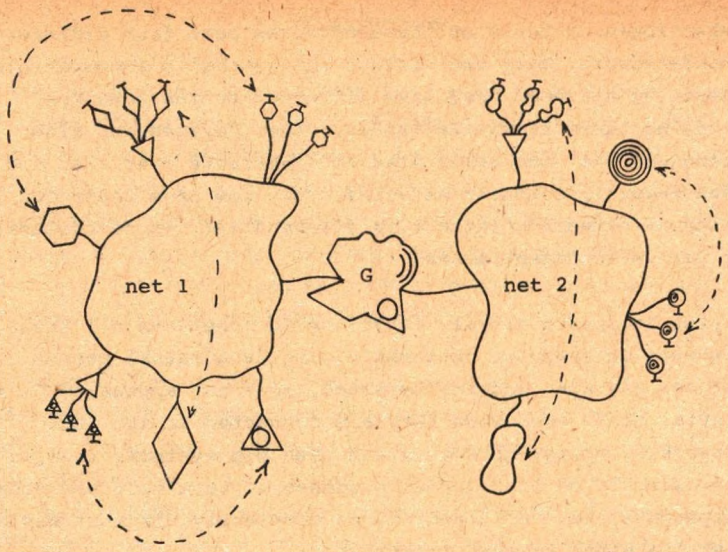


Fig. 1 - Global interworking

One can also see that the interworking problem may exist between incompatible elements of the same network as well.

2.2. Interworking between standard elements

We have seen that, if one wants to preserve completely any local network (and even systems and terminals) characteristics and sometime exotic features, conversion functions are needed for any pair of systems one wants to make interwork.

One alternative to that deadlock situation is to define common standards. These standards can be used on each network, most of the time on top, or in parallel with the local conventions, to allow interworking between previously incompatible elements.

Standards can be defined where levels of services are common, between the systems or networks involved, thus reducing considerably conversion problems. Such standard conventions to be used for cooperation throughout the networks, must make minimum assumptions on internal network architecture and conventions of each system. They only apply to cooperation between components of a "supernetwork" made of the concatenation of individual networks. Of course, when such conventions are widely established and widely admitted, it may be sensible to use them also for internal cooperation within one network (or system), but it is not, in a any case, a must.

The gateway between two networks appears now as the component, placed between them, which interface each of them, in order to allow communication between standard elements. Fig. 2 illustrates such a task, for host to terminal applications using standard conventions.

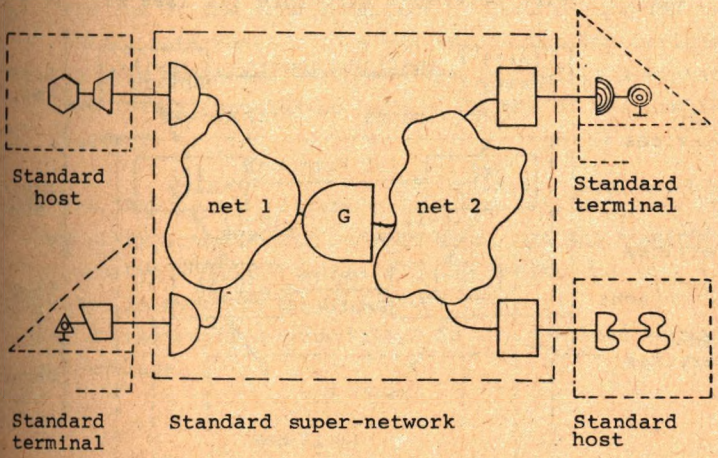


Fig. 2 - Interworking of standard elements

In other words, gateways can appear as nodes of a "super-network" made of standard elements (e.g. standard hosts and standard terminals). These standard elements are connected to the super-nodes by means of individual local networks, which are also used to connect super nodes to each other.

3. NETWORK ARCHITECTURE

3.1. Layered architecture

Networks can be seen as made of entities which are associated in order to perform given services. Possible associations of these entities may be very complex but they can usually be decomposed into elementary categories. The types of such structures usually referred to, are the "onion skin" or layered architecture, the "hop by hop" or cascade assembly and the end to end control.

1°) A layered architecture can be formally represented by the following reference structure [1] (see Fig. 3) :

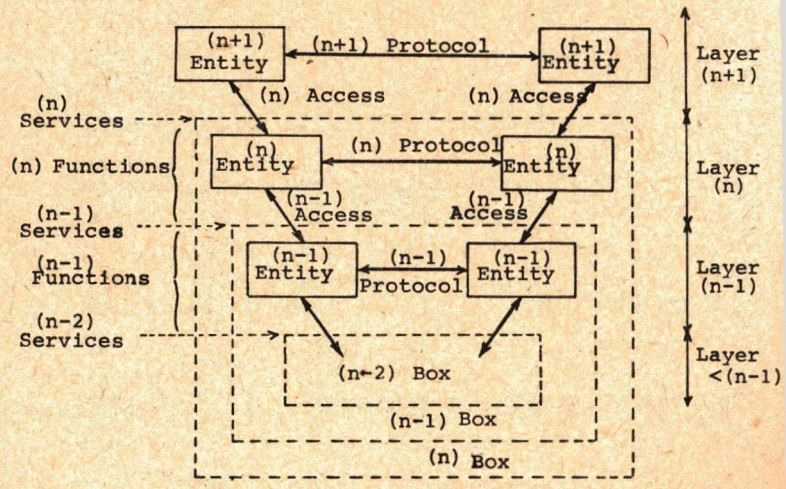


Fig. 3 - Layered architecture

- Layer (n) of the structure makes use of (n-1) Services provided by the lower layers through the (n-1) Access (see note).
- The structure of these lower layers is not known by layer (n) that considers only the services provided by an (n-1) Box.
- Layer (n) is made of (n) Entities that cooperate according to an (n) Protocol.
- The (n) entities perform (n) Functions using the (n-1) services to provide (n) services to the layer (n+1).
- The specifications of a layer of the architecture must in some way refer to the set of services provided by lower layers. This is done by making use of Access-Functions. The set of access functions to a service can be viewed simply as a means to describe the logical structure of the network. It does not necessarily imply the existence of the corresponding Interface in any implementation of a piece of the network.

Note : The use of the word interface is here restricted to actual interfaces in the context of implementations. In this model the word "access" is used to designate the logical boundaries between layers of the architecture.

2°) In the "hop by hop" assembly (see Fig. 4), entities of a common layer are placed side by side, and the cooperation between two adjacent entities (n i) entity and (n i + 1) entity is ruled by an (n i, i+1) Protocol.
Such a construction constitutes an (n) Box that provides (n) Services.

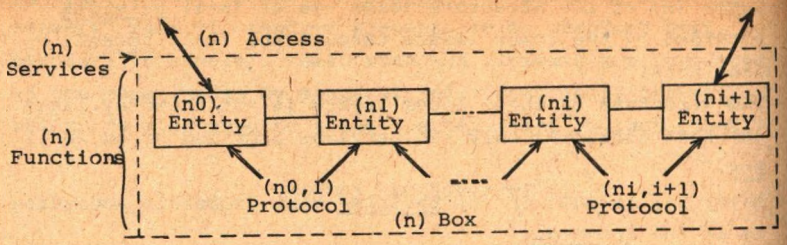


Fig. 4 - Hop by hop assembly

Most of the time, inside one network design, entities and protocols along the same chain, are identical (e.g. the assembly of successive packet switching nodes with a node-node protocol to form a data path between two destinations).

3°) The end to end control usually applies to the control of a cascade (or more complex) assembly done by two entities at the far ends, cooperating directly through an end to end protocol (see Fig. 5). It allows to introduce into a layered architecture, services provided through a more complex assembly of entities of the same layer.

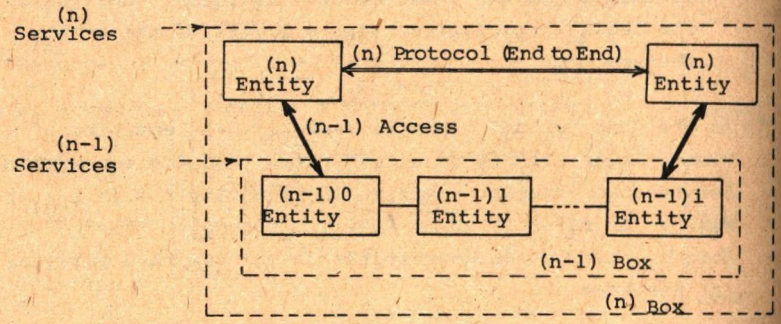


Fig. 5 - End to End control

When interconnecting networks, by means of gateways, end to end control will allow to bring again the "super network" into the layered architecture model (see Section 4).

3.2. Visibility of the structure

The logical structure of the layered architecture model ensures maximum flexibility since it allows to replace any box by another one, provided it offers the same services. That is to say that the Visibility one layer has of the underlying boxes remains unchanged. Adopting such a model with independent layers, for internetwork standards design ultimately favours a modular architecture. This means that as long as the innermost layers remain functionally the same, any change in their implementation or in their internal conventions do not impact the outermost layers.

Thus, it is possible to define a protocol for (n) entities independently from physical implementation of (n-1) box. This flexibility can be extended to the implementation of each piece of the network provided the interfaces between layers actually appear. This may be too constraining or too costly for some network implementation. In that case, it might be a sensible choice to implement several entities of consecutive layers with a different structure (e.g. as a single piece without identifiable interfaces), the loss in flexibility (if one layer changes, the whole piece has to be changed) being balanced by a lower cost in implementation and/or operation. The only must is that the outside appearance of the set of entities complies with the model, i.e. the visibility of the protocols remains unchanged.

3.3. Functional lay-out

The functional layers, that are usually present in most computer networks, and which are candidate for providing standard levels of services, are the following :

- The data transmission facilities which constitute the lower layers of the architecture. Data Transmission Services are provided by Data Transmission Boxes. CIGALE [2] is such a box providing probably the most adequate data transmission service for distributed informatics (i.e. Datagram service [3],[4]). Other data transmission boxes/services exist (e.g. Virtual Call service, Leased Lines ...) and it is most desirable that their variations be not visible from application programs that constitute a huge investment and must therefore be kept independent from changes in the data transmission part of the network. This is achieved by :

- Transport services through which higher layer entities communicate. These transport services are provided by a Transport Box embedding (but masking) a possible variety of Data Transmission Boxes. The Transport Layer built on top of the Data Transmission Boxes is made of Transport entities called Transport Stations located in Data Processing Systems. These Transport Stations cooperate according to a Transport Protocol [5] making use of data transmission services to provide Transport Services [6], to the higher layers. These transport services can be seen as an extension, over the network of inter-process communication facilities.

- Among the higher layers protocols making use of the transport services, one of the most commonly used is the Virtual Terminal Protocol [7],[8], which masks the incompatibilities of real devices to present to the applications a standard way to drive terminals.

4. NETWORK INTERCONNECTION AND PROTOCOL CONVERSION MODEL

The aim of interconnecting computer networks is to allow "standard users" accessing different networks to communicate. These "users" must be placed at the same functional layer and therefore must have the same view of the super-network. This common view can be obtained through end to end standard conventions at the super network level (see Fig. 6).

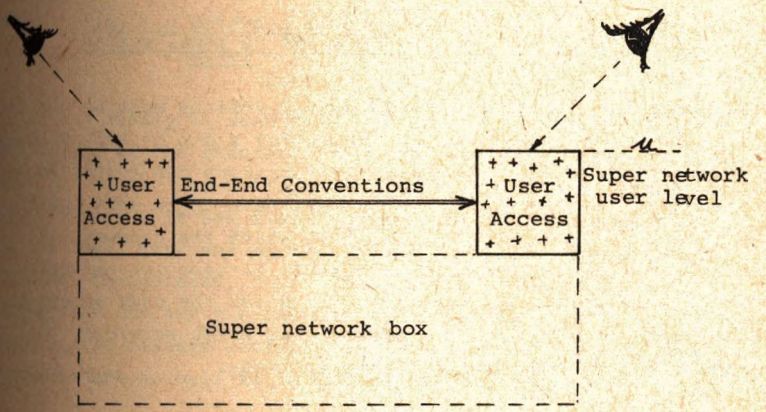


Fig. 6 - Super network user access

The super network box is made by the interconnection of local network through what we called gateways.
Gateways are the super-nodes of the super network and therefore act as focal points, between each individual net, which any end to end conversations has to go through (see Fig. 7).

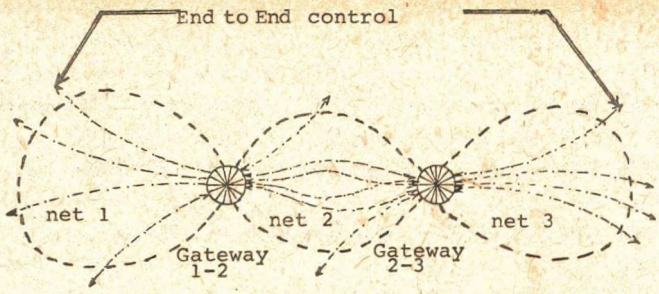


Fig. 7 - Gateways as focal points

Being at focal points, the gateway functions must be placed at a level of service which can suffer a hop by hop assembly. This will determine what kind of functions can therefore be placed at this level, and what will have to be left end to end. Let us call "g" this level at which the gateway conversion functions must be placed, between two networks. This level will also be the one on which the user access to the super network will be built, i.e. which services, the end to end standard conventions will use.

Looking now at each local net, some up (or low) grading of their respective level of service may have to be made to bring them to the gateway standard level "g". This is the task of the access entities to the local nets, in the hosts as well as in the gateways. The gateway can now be seen as performing a bridging function between two networks at the same levels of service. The bridging consists in bringing each local net level of service to a standard "g" level, common all over the super network, and at which the communication is done. In addition, the gateway can perform if necessary some format conversions between the actual interfaces to each network (see Fig. 8).

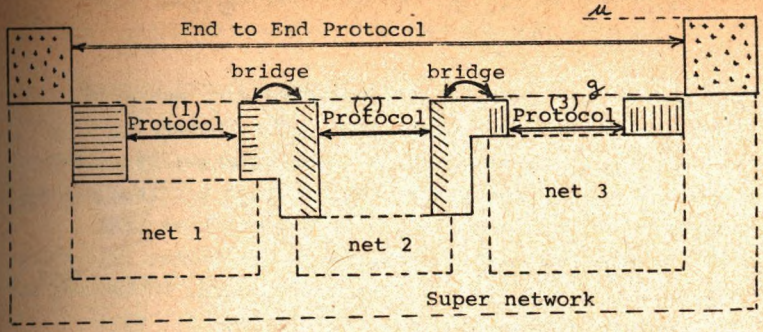


Fig. 8 - Gateways in the network interconnection model

As it can easily be seen from the above discussion, the definition of the standard "g" level as well as the standard user "μ" level, has an important impact on the supernetwork architecture ; it has also consequences on the access boxes to the local nets and on the end to end protocols that will be used. The gateway, cannot be considered as an ad hoc conversion box any one can build between two networks. On the contrary it belongs actively into each individual network architecture as well as in the overall design.

The network interconnection problems can now be looked at, as the definition of these "g" level of services, with the constraints mentionned, and their implementation within each individual net as well as in the gateways. The "g" layer, built on a hop by hop assembly, will have to be controlled by an end to end protocol, insuring again a layered architecture to the super network and providing a new level of service "μ" to the super network users.

5. INTERCONNECTION AT THE DATA TRANSMISSION LEVEL (ADDRESSING AND ROUTING PROBLEMS)

Each network has usually its own addressing scheme to allow its users to designate their correspondents. When networks are interconnected, a global addressing scheme is needed to name uniquely any subscriber of any interconnected network. Such standard addressing schemes have been proposed together with a standard packet format [9], using a hierarchical structure : a network address followed by the local address within the network (see Fig. 9). This addressing structure allows to preserve existing local conventions for addressing and creates a global name space by concatenation of local networks name spaces.

There is, in principle, no difficulty to standardize such a global addressing scheme, which should not change the way subscribers are named within one particular network, and allows to address users of other networks. But addressing conventions cannot be isolated from routing techniques [10]. Being able to name subscribers of other networks means also that information (i.e. packets) can be routed to these addresses.

Some networks, like CIGALE [11], have been designed to take into account a hierarchical address structure. For such networks, a simple concatenation of several of them, just by linking nodes together, makes a super network, as far as addressing and routing is concerned (e.g. CIGALE + CIGALE = CIGALE). In this case, any node can act as a gateway for this function. This is usually not the case, even with identical networks, if they handle a single level address space (i.e. at least, previously identical local addresses will have to be changed, and routing algorithms accordingly).

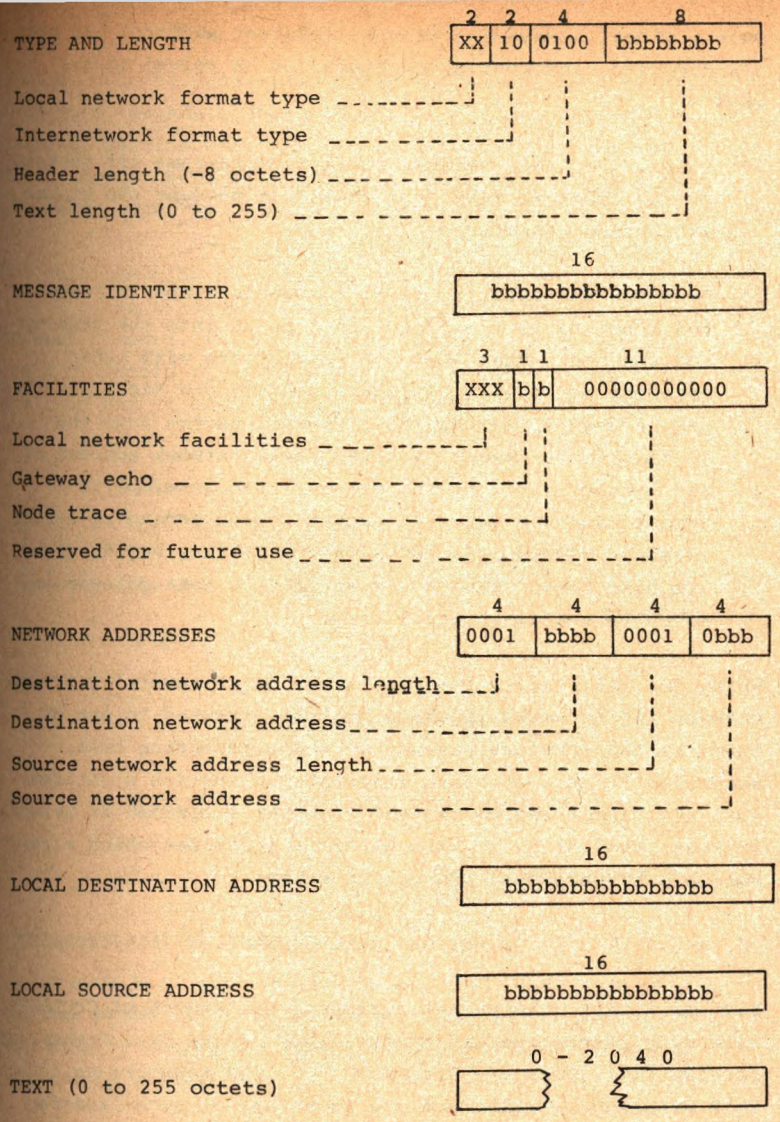


Fig. 9 - Proposed standard D-Format

Once a global addressing scheme (internet address) has been standardized, the routing problems can be solved in two ways :

1°) Change the local network routing mechanisms (if necessary) in order to make them take into account the global addressing scheme.

2°) Preserve the existing local conventions and introduce the handling of the global name space into the gateways and the hosts that wish to communicate with other networks hosts (i.e. internet hosts). One technique that can be used is to wrap the internet packet into the local network header with the local network address of the gateway [12]. At the gateway, the envelope is removed and the internet packet routed according to the internet address ; a new envelope is then built, with the next local network address of the next gateway and so on to the final local host.

When, for reliability reasons, there exist several gateways between two networks, packets must be directed to one of them, by the previous gateway or the originating host, according to some standard internetwork routing algorithm. Such algorithms can be derived from the extension at network address level of those used within a local net which already handle a hierarchical addressing scheme.

6. A SIMPLE CASE OF NETWORK INTERCONNECTION (EIN/CYCLADES)

EIN and CIGALE (the packet switching subnetwork of CYCLADES) have been interconnected at the level of the Data Transmission service [12]. Both are Datagram (DG) type networks using the IFIP proposed D-Format and one advantage of such Datagram networks is that the service obtained by their concatenation is still a Datagram service (which may not be true with

Virtual Call services). In other words, they can be assembled on a cascade without loosing their properties. The level of service being the same on both sides, it was not necessary to upgrade it at one side or use only a subset at the other side, to come to a common standard level.

The main function of the gateway in this case, is to perform format conversions between the respective access primitives to each network. The end to end functions of the upper layer protocols (i.e. Transport layer and Virtual Terminal) are kept end to end [13] (see Fig. 10).

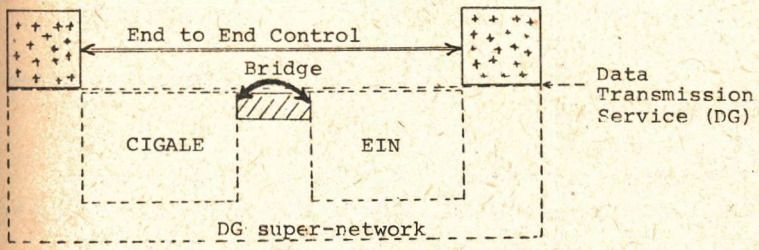


Fig. 10 - FIN/CIGALE interconnection

The gateway has been implemented on a CII-HB Mitra 15 minicomputer which can also act as a CIGALE node (i.e. it has been made by adding the gateway functions on top of the CIGALE node software).

Packets are taken from one network and submitted straight away to the other one, after some adjustments in the packet header facilities field have been made. This interconnection, very close to a simple concatenation of both DG networks, experiments the CATENET concept introduced in [14].

BIBLIOGRAPHY

[1] AFNOR - Basic architecture for end to end protocols, Contribution to ISO, ISO/TC97/SC6 N 1369, (Jan. 77), 8p.

[2] POUZIN L. - CIGALE, the packet switching machine of the Cyclades computer network, IFIP Congress, Stockholm, (Aug. 74), 155-159.

[3] BRITISH POST-OFFICE - Proposals for a Datagram Service, Contribution to CCITT, CCITT-COM.VII-nr 24.

[4] ANSI - Generic requirements for Datagram Service, Contribution to ISO, ISO/TC97/SC6 N 1403, (Feb. 77), 8p.

[5] CERF V.G./Mc KENZIE A./SCANTLEBURY R./ZIMMERMANN H. - Proposal for an International End-to-End Protocol, ACM Sigcomm. Comput. Comm. Review, vol.6, nr 1, ISO/TC97/SC6 N 1282, INWG note ≠≠ 96, (July 75), 29 p.

[6] AFNOR - Definition of the transport layer in the End to End protocols architecture, ISO/TC97/SC6 N 1368, (Jan. 77), 5 p.

[7] EURONET - Data entry virtual terminal protocol for EURONET, VTP-D/1, (Jan. 77), 30 p.

[8] EIN - Proposal for a scroll mode virtual terminal, edited by SCHICKER P. and ZIMMERMANN H., EIN/CCG/77/02, INWG note ≠≠ 62, (Jan. 77), 45 p.

[9] INWG - Basic message format for internetwork communication, INWG note ## 83, also Contribution from IFIP to ISO/TC97/SC6 N 1281 and CCITT-COM.VII-nr D69, (May 75), 7 p.

[10] SUNSHINE C.A. - Interconnection of computer networks, Computer networks 1, (1977), North-Holland pub., 175-195.

[11] POUZIN L. - Revised CIGALE header, réseau Cyclades MIT 571, (Apr. 74), 2 p.

[12] GIEN M./LAWS J./SCANTLEBURY R. - Interconnection of packet switching networks : Theory and Practice, EUROCOMP, Brunel Univ., (Sept. 75), 241-260.

[13] DEPARIS M./DUENKI A./GIEN M./LAWS J./LE MOLI G./ WEAVING K. - The implementation of an End to End Protocol by EIN centres : a survey and comparison, ICCC 76, Toronto, (Aug. 76), 351-359.

[14] POUZIN L. - A proposal for interconnecting packet switching networks, EUROCOMP, Brunel Univ. (May 74), 1023-1036.

NETWORK ARCHITECTURES AND COMPONENTS

Louis POUZIN

I - INTRODUCTION

There are various brands of computer networks. The present paper is only concerned with the type of general purpose heterogeneous computer network of which Arpanet is a typical prototype. In this field experience is scarce, since no network has yet progressed beyond an initial experimental stage, while most are still in a planning or construction phase. Consequently, applications and user needs can only be thought of in prospective terms. Organizational and sociological issues hiding underneath the concept of network may be of such magnitude that many years will elapse before we can record any significant breakthrough.

In the present stage major efforts bear on technical issues, with the objective of developing and mastering the network technology. We have to recognize that proposed structures and techniques have yet to be validated from experience, and that various other approaches might as well be considered. It would be nice if we had practical evidence that present views are justified by economics and user acceptance. But assessing networks on today's applications is not necessarily a sounder standpoint.

II - COMPUTER NETWORK CHARACTERISTICS

A - Heterogeneous installations

Networks of computers may be homogeneous. But this is the exception rather than the rule. Some organizations, like IBM, Control Data, University Computing Co., etc... have set up computer networks, to provide specific services, using the same brand of machines and operating systems. Sooner or later a new generation comes by and must be inserted. Replacing a whole network with new systems in a short span of time is quite unrealistic. Thus, homogeneous networks are bound to turn heterogeneous, or die out.

Most computer networks cannot afford to be homogeneous in the first place. They have to start out with existing installations, which happen to be a mixture of every computer system on the market. But this is a reasonable constraint, as it forces to design network mechanisms for the general case, which makes them more suitable to accomodate future extensions. (Fig. 1)

As a result of this heterogeneity, there is no common interface, or procedure to interchange data. Actually, this is not a direct implication of having heterogeneous computer systems. Indeed, there might exist some generally agreed methods to govern data exchange between various computers. But this is not so. Every manufacturer is still entrenched in its closed world of idiosynchrasies, and must devote a great deal of effort only to have its own machines to communicate.

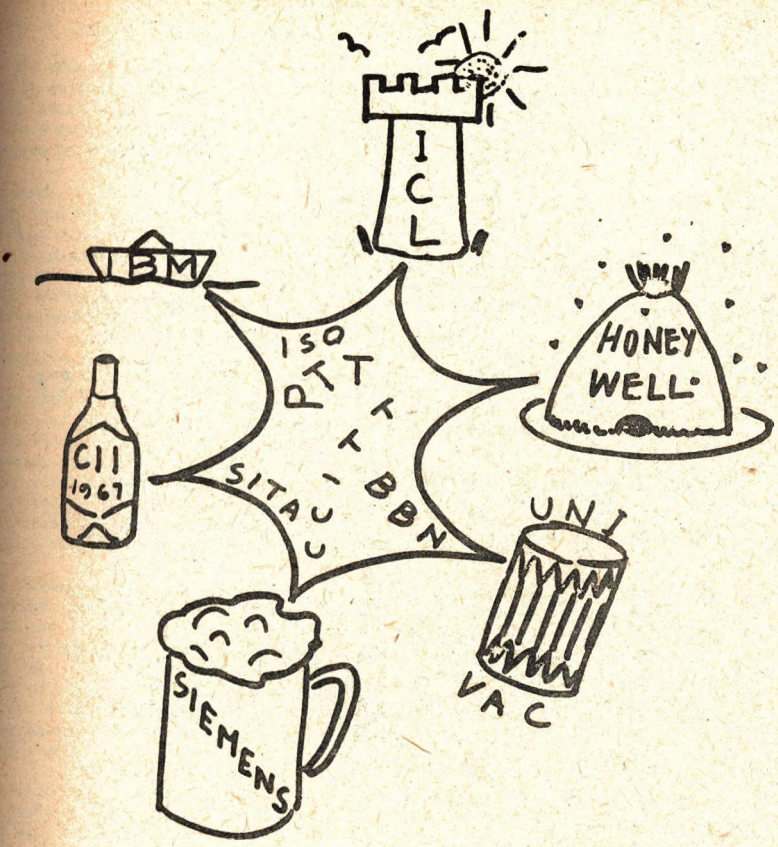
Installations are not just heterogeneous in terms of computers, but also in terms of management, activities, interests, habits, and there is no magic receipe to turn them into a strong unified body. Being part of a network should not mean losing autonomy and decision freedom. They want to retain all their capabilities to run their own business as they please. Local peculiarities are to be expected as a matter of course, and operation of the whole network should be made independent of local installation behavior.

As in any association, federation, or business, some members tend to cluster in sub-groups, or clubs, based on some common ties not shared by others. In a computer network, clubs may be made of users of the same system, service bureaus, bankers, installations of a corporation. Presumably, most of the traffic generated by members of a club will be exchanged with other members. Conventions about information interchange may well be tailored to some particular dominant type of transactions peculiar to the club. (Fig. 2)

On the other hand, members of a club will want occasionally to have some exchange with members of another club. They may even belong simultaneously to different clubs. Although a club sounds initially as a closed group, it should not be that closed after all. Some excursions should be possible toward foreign installations, just in case. In fact, installations belong to what we may call semi-closed groups.

B - Data communications system

The challenge in designing a communications system for a computer network is that only qualitative aspects may be guessed. No figures are available, since computer networks do not exist. Or at least they have not yet emerged out of the initial building up stage. What we may assume is that there will be computers and terminals, and some barely predictable amount of data traffic between them. Presumably, the amount of traffic will be related to the capability of those devices to generate or accept data. In other words, computer-to-computer traffic, high speed, and low speed terminals will likely fall within distinct categories of traffic patterns.

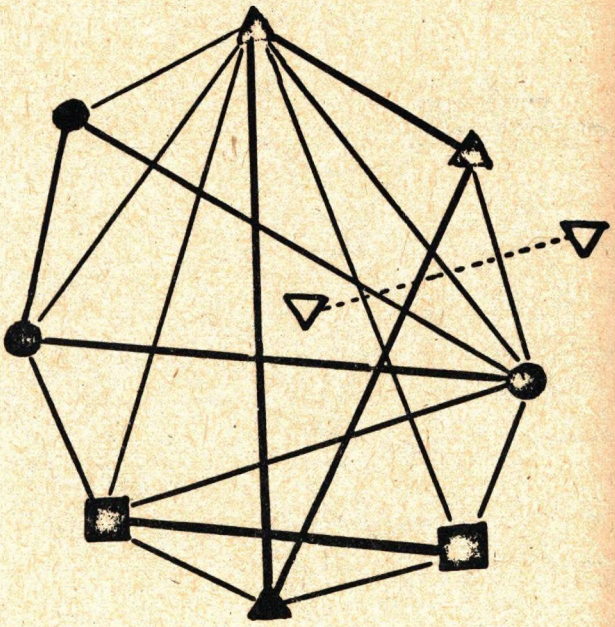


Heterogenius

computer

network

Figure 1.



Semi- closed groups

Figure 2.

Therefore, the communications system will have to meet simultaneously a mixture of requirements pertaining to different classes of data interchange. (Fig. 3) This comes in addition to carrying dominant flows, depending on the geographical topology of installation clubs.

Since the communications system should cater for an unlimited number of computer makes, be suitable for semi-private traffic, and work in an environment of loosely related installations, it cannot depend on any of them. In a context of mutually suspicious parties, the communications system will be deemed at fault, unless it has proven very reliable.

Furthermore, it should carry, within some limits, a somewhat unpredictable traffic, ranging from low speed conversational messages, up to bursty high speed process-to-process interplay. New installations may be added, others can be discontinued. Since the number and type of applications using the network is a priori evolutionary, there cannot be any favorite characteristics tailored to some specific traffic. Although some requirements may turn out to be contradictory, the communications system has to walk a thin line, and comply efficiently with a diversity of traffic patterns. This is called flexibility, in a subjective sense, as opposed to rigidity.

Carrying data between heterogeneous computers or terminals, including all sorts of private or semi-public conventions, would end up in a Penelope web type of task, if the communications system had to get involved in data codes, formats, procedures, etc... Adding constantly new features, fixing old ones, would not make for a reliable system, and installations would resent being tied with not quite satisfactory network options. A better approach is to make the communications system data transparent, in order to accomodate any type of code or convention that installations might want to use. (Fig. 4)

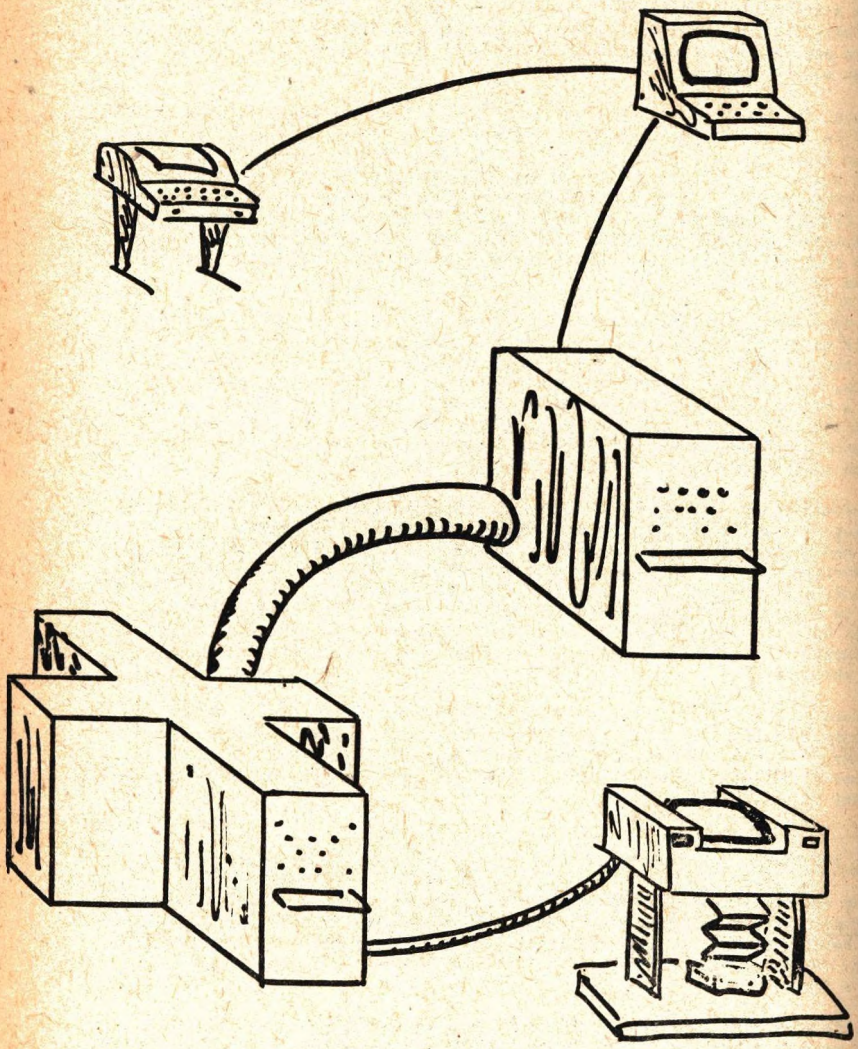
Once a network starts operating, it becomes very frustrating for users to disrupt its working. Particularly, if they have built a substantial investment in software and organizational structures. A most appreciated virtue of a communications systems is to provide for abiding service, so that long term planning be carried out, counting upon well defined network characteristics. Communications are akin to a public service. Changing components of the communications system should not have any visible impact, other than improvement, on the way installations exchange data. Requiring all users to adapt their programs overnight would verge on sheer upheaval.

The previous considerations converge towards opting for a communications systems as independent as possible from installations or terminals which it is to serve. On this way, both users and communications system can protect their investment, and be free to introduce whatever gadget they like, as long as they keep the same conventions to talk to each other.

C - Basic structure

As already exemplified in Arpanet, the resulting structure of a computer network is two-level. A first level is the communications network, in charge of carrying messages between computers. A second level is the set of computers, concentrators, terminals, which use the communications system as a black box. (See e.g. H. Zimmermann's paper)

Ideally, the communications system should be so transparent that installations would not even notice it is there. In other words, installations are not concerned primarily with exchanging messages with some communications medium. They would like to send and receive data as if they were in direct connection with other installations. But we have seen previously that installations are heterogeneous,



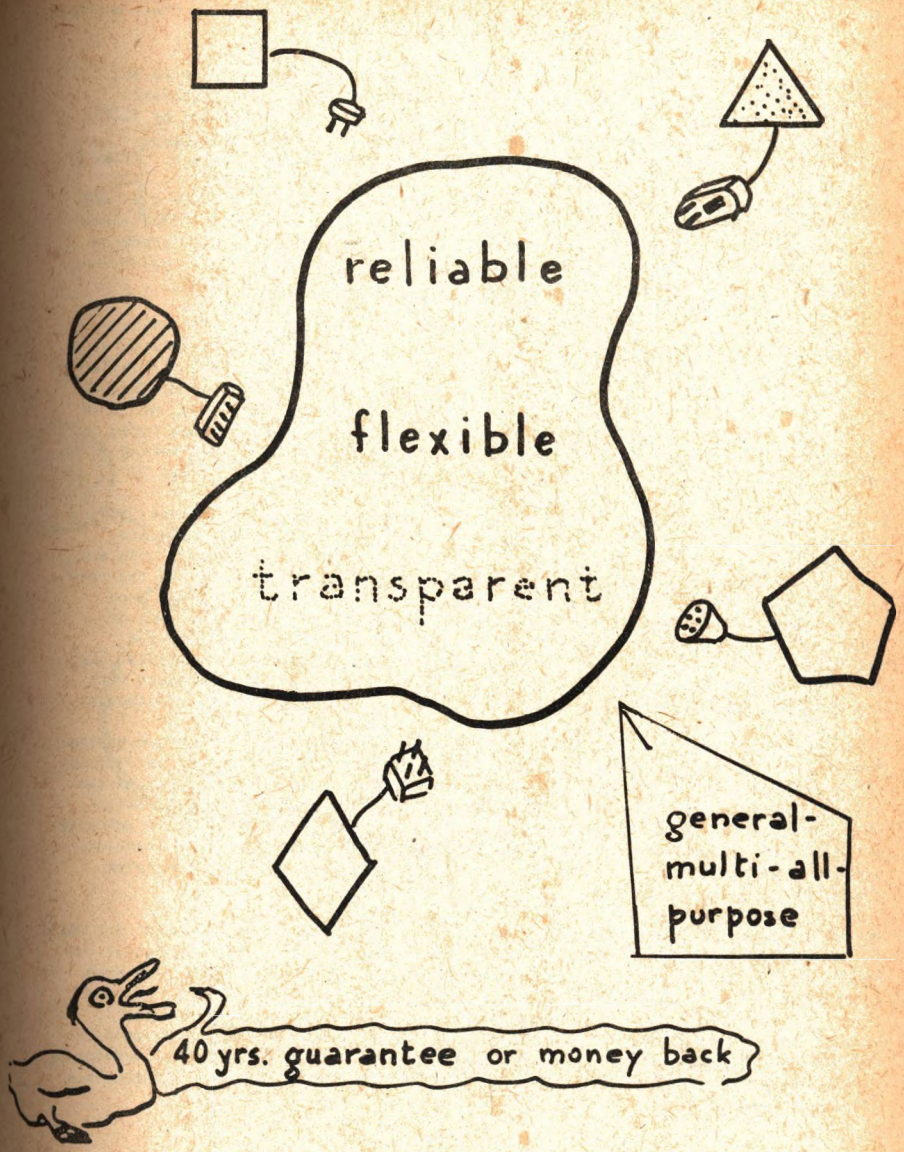

reliable
flexible
transparent
general-
multi-all-
purpose
40 yrs. guarantee or money back

Figure 4.

and do not have any common convention to exchange data. A solution would be to force every one of them to adapt to a network standard. This might be reasonable if such a standard were originating from a distinguished body, like ISO, or CCITT, or the U.S. Navy. Otherwise, one can expect most computer manufacturers being up in arms, as there is nothing they loathe so much as users putting on their machines some exotic piece of hardware or operating system.

A more attractive approach consists in putting some interface adapter in the communications system, so that no modification be required from individual installations. This is acceptable, since data speeds are standardized by CCITT, and transmission procedures can be limited to those of a half dozen major manufacturers, while others can be talked into adapting their software. (Fig. 5)

Nevertheless, procedures may change with a new system release, and new ones are put forth over the years. Thus, it is typically the kind of function that should be made modular and optional, so that the rest of the communications system be kept well insulated from procedure updating. Before long it will be located in a pluggable micro-program. (Some machines already have this feature).

We have seen above that installations may occasionally interfere with members of different clubs, and that some private conventions, as opposed to manufacturer's, might be used in the context of specific applications. Thus, a single installation may wish to use a few different options, according to its type of work. This is quite feasible if it has several links with the communications system, or if it can direct messages toward appropriate adapters according to a message type, (software link). Logically this results in the communications system seeing several virtual installations mapped onto the same physical one. (Fig. 6)

A bare communications service may be unsufficient for some installations. They may wish to have more sophisticated options, like long term storage of data, information retrieval, traffic back-up in case of installation outage, code and format conversion, and what have you. There is obviously no limit as to the type of services that may be offered, grafted on information transfer. It is an area where experience and an imaginative marketing approach can result in many an interesting business. At first sight, this seems in contradiction with the desirable characteristics of a communications system : long term stability, high reliability, while said services would smack more of a commercial product with its all too familiar lack of testing, specification changes, quick obsolescence.

These opposite objectives may be reconciled through an appropriate system design. Anything not directly relevant to data communications purposes should be left out of the communications system. On the other hand, desirable add-on functions can be implemented by specific installations considered as somehow embedded within the communications system, as seen from external users, while they would actually be logically independent, as any other external installation. For all practical purpose, their junction with the communications system would be no different whatsoever. (Fig. 7)

On the previous chart, we have represented some "internal" installations. But there is no intrisic technical difference between internal and external, as far as the communications system is concerned. The distinction is intended to stress such considerations as guarantee of service, tariff structure, maintenance, and the like. But there is a continuum ranging from a public service, to a non-profit agency, to a service bureau, to an organization offering services as by-products. The decision as to which services should be offered from within or from outside is mainly commercial or political. One can even switch status, without interference in communications techniques.

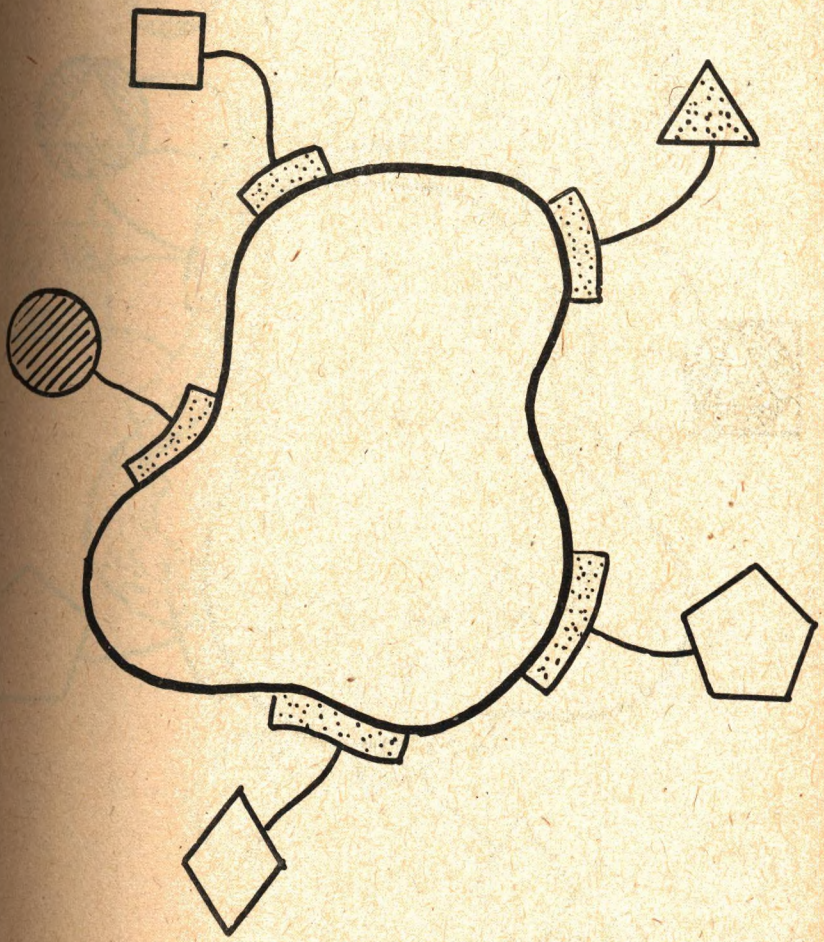


Figure 5.

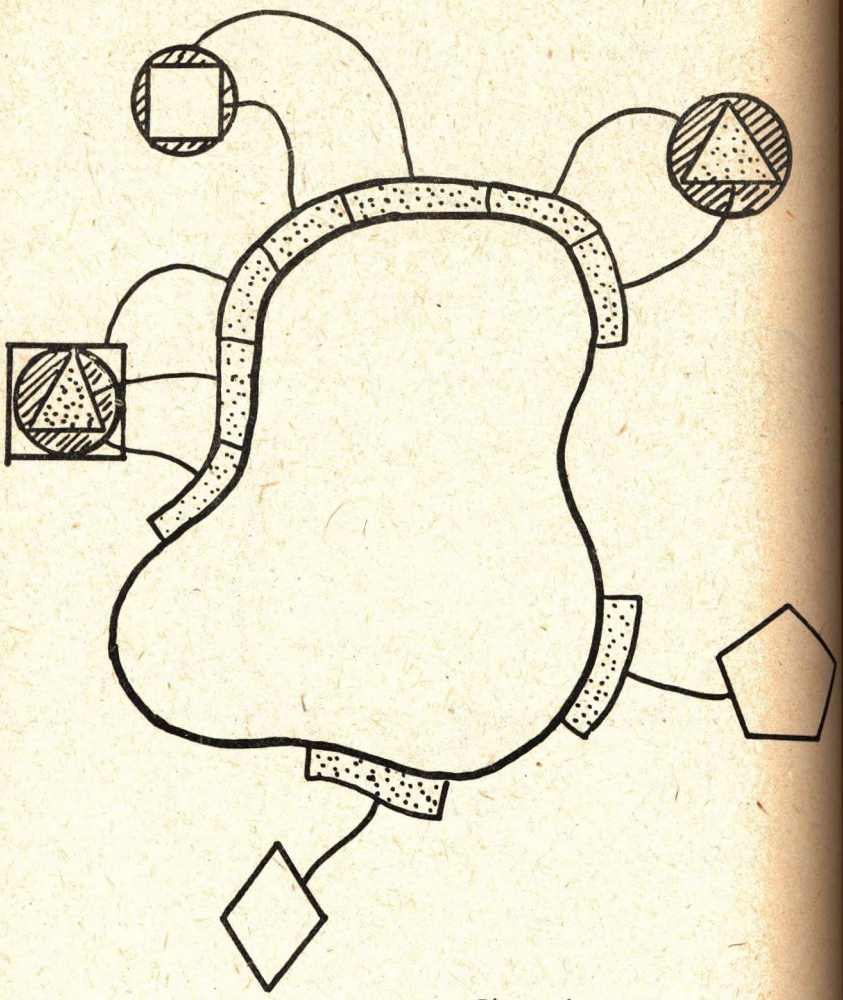


Figure 6.

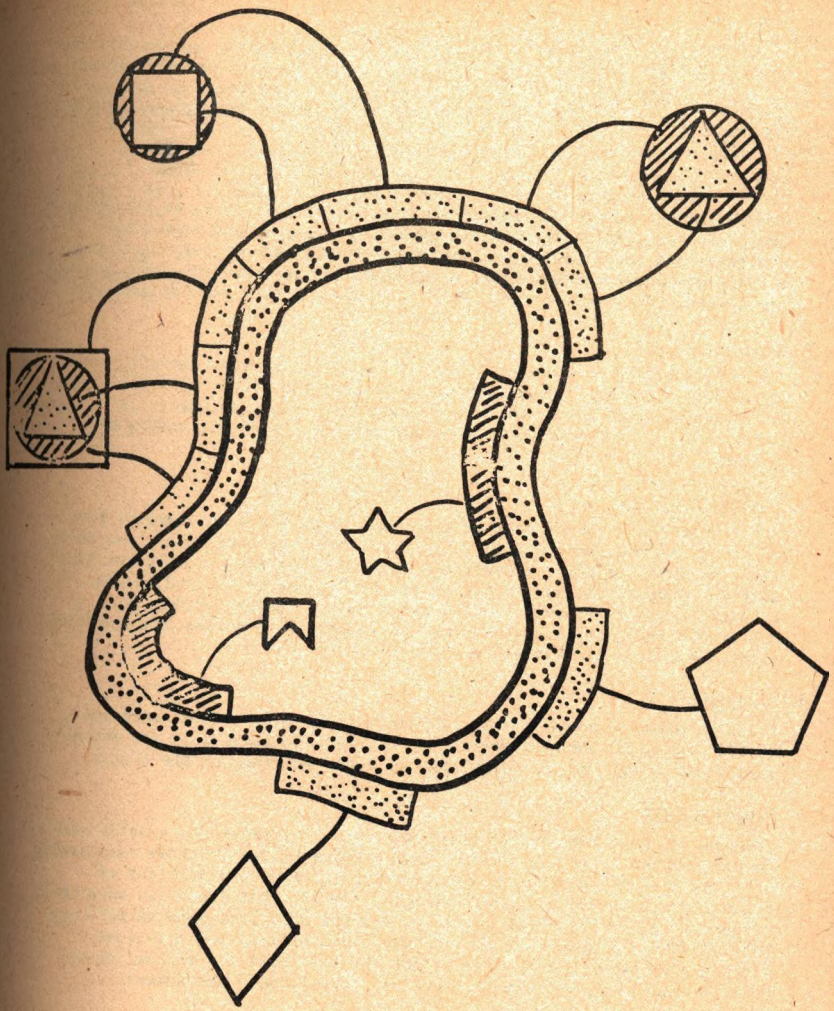


Figure 7.

D - Network Authority

It certainly could be read as a miracle if every component of a communications system would belong to a different organization, and if the service were satisfactory with no one responsible for it. Myriads of operational problems require a well defined responsibility vested in an organization we may call the Network Authority. It is not necessarily the owner of every piece of equipment, but it is assigned the mission of seeing to it that the whole system works properly. The extent of this mission may vary : typically it would cover the following areas :

1. Maintenance :

   Namely error tracing and repairing. This is a tricky task, as it involves equipment from various suppliers, including lines leased from common carriers. And it is positively undesirable to bring the communications system to a complete halt, just to find out which component needs fixing up. Interventions must be done on a live system, with appropriate care

2. Accounting :

   Exchanging data costs money. Again, it would not be very satisfactory to rely upon individual installations to declare the amount of traffic they have been generating. On the other hand, the Network Authority is in a good position to record traffic figures suitable for later accounting and billing.

3. Traffic analysis :

   Communications networks are far from being completely understood. From mathematical theories to practical cases, numerous studies are presently being pursued. In order to validate or feed in models, traffic patterns must be recorded and sorted out. Furthermore, due to the individual freedom of installations, traffic patterns may change rapidly. A continuous analysis is necessary to anticipate in due time configuration changes and tune the system to its actual requirements.

4. Development :

   It is an obvious extension of the responsibility for maintenance and traffic analysis. Developpement is probably best assured by the organization which has the best knowledge on inner workings of the system.

5. Clearinghouse :

   Independent from the cost of communications, installations provide services to one another, which is liable to some cross-billing. Rather than having each one to send invoices to every other, it is more convenient to consolidate all accounting in a single location, and apportion each one's balance sheet to its total activity. All the more that some tariffs may be tied with the amount of traffic exchanged between installations. Although the Network Authority is not the only place in a position to perform this task, putting it there is probably more convenient.

6. Network control center :

   Most tasks entrusted to the Network Authority cannot be accomplished without computerized aid, such as running diagnostics tests, monitoring traffic, cross-billing, etc... For that purpose a computer installation is needed, and it can be linked to the communications system on the same basis as any other installation of the network. Owing to its obvious privileges, this installation can be considered as "internal", in the sense used previously.

E - Information centers

The diversity and the geographical distribution of resources in a computer network create the need for various information repositories. This belongs to network sociology, rather than technology. Indeed, like in any association conducting business, some media are needed to facilitate an encounter between supply and demand. This can take the form of ads, news, manuals, indexes, mail service, broadcast, etc... Probably at some point in the future the video-technology will allow live meetings to take place over networks, and be canned in play-back files. An example of such a center is developing in Arpanet at Stanford Research Lab. Some of these services might be provided by the Network Authority, but other specialized installations may do it as well. Ideally, it should be thought of as set of coordinated interactions between installations, and again some dominant patterns are likely to match the partitionning into clubs. (Fig. 8)

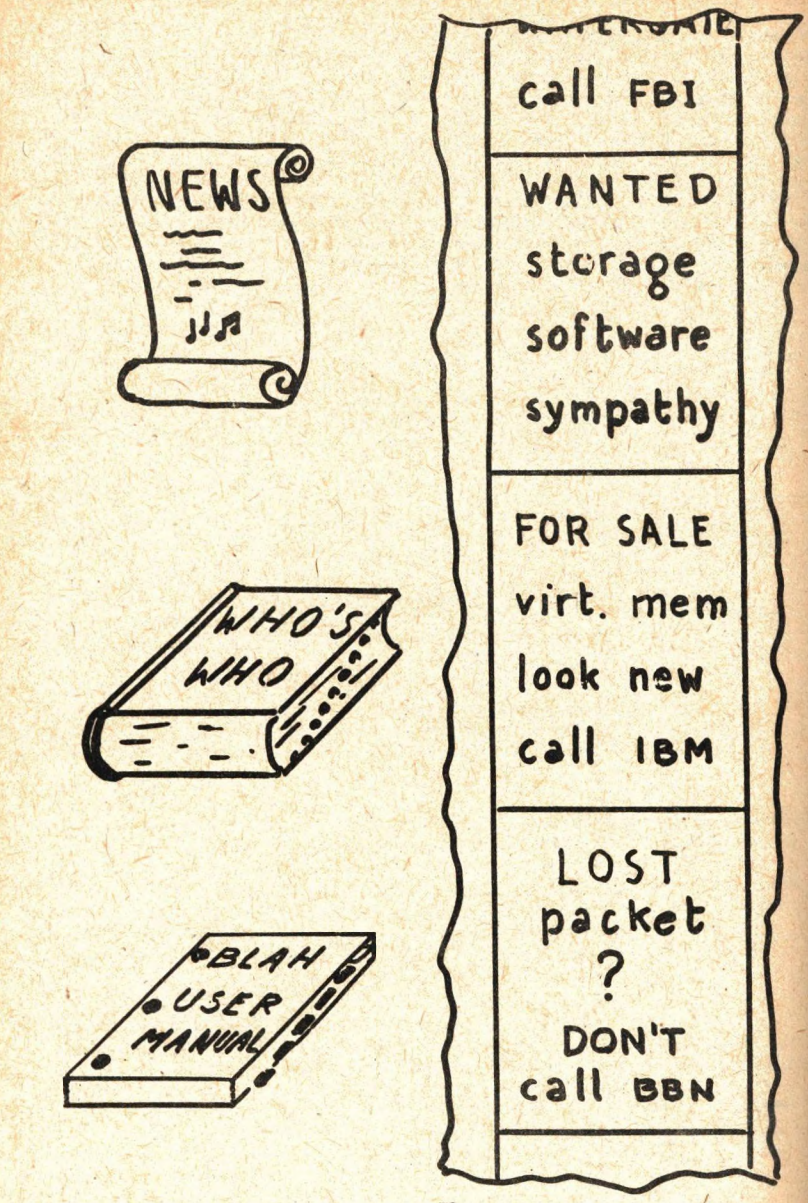

NEWS
WHO'S WHO
BLAH
USER
MANUAL
call FBI
WANTED
storage
software
sympathy
FOR SALE
virt. mem
look new
call IBM
LOST
packet
?
DON'T
call BBN

Figure 8.

A - Data switching machine

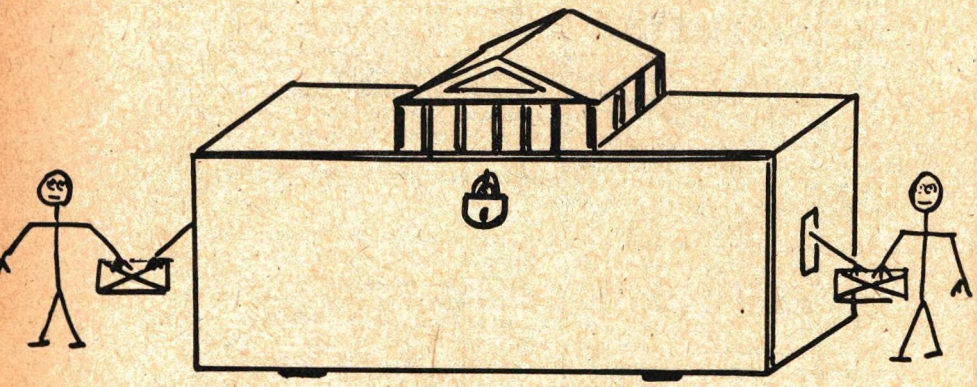
Various techniques may be considered when it comes to set up some communications system between computers. What they have in common is that they must comply with some government regulated monopoly, and they require existing communications facilities Developing a specific one is perhaps not to be ruled out, but the initial cost would be unacceptable, unless intense pressure is put on, like in a period of war.

It is not the intent of this paper to discuss communications techniques. Let us remark only that computer input-output is message oriented, in digital form, and that existing communications facilities have been engineered for something else than data communications, (voice, telegrams, television). More recently a technique called "packet switching" has been proposed and experimented, (see Dr. Davies'paper). It seems to make the best out of existing technologies, and it can be supported by telephone, radio, or satellite transmission. Observing the trend in networks shows that transmission systems evolve towards a message store and forward technique with short messages, and short transit time, in order to meet the requirements of conversational traffic. With mini-computers and the availability of high speed lines, it looks as if the day of packet switching has come, at least for a decade, till new transmission systems are widely offered. In the following, assumptions made about communications systems will be restricted to packet switching.

As we have seen before, we want a communications system independent and insulated from computer installations. Consequently, a packet switching system has to be a computer network in its own right, but a highly specialized one. What it has to do is to switch short messages (packets) on the best possible way, i.e. fast, efficient, reliable. Of course, each of these adjectives might take pages of discussion. Since every other additional service is to be implemented in some internal or external installation, the communications network should do no more than packet switching. Externally it looks like a black box to which messages are handed for delivery to a specified destination. (Fig. 9)

The internal structure of a communications network could be looked at conventionally as reflecting its physical structure, i.e. a set of computers exchanging messages, to carry either data or control information. There is nothing invalid in such a model. But some concepts, in addition to represent a physical structure, have also the power to expand and generalize the model in a way that brings new light, sets guidelines, and eventually helps re-thinking the structure itself. Considered as a black box, a communications network can be viewed as an abstract machine, which as usual has an instruction code, processors, I - O, and internal states. Passing a message over to a communications processor (node) is tantamount to inputting an instruction (op. code + operand). Executing the instruction is in effect transferring the message into another component of the machine, and output it. In a short-hand form this could read :

```
input A  . . . . . input message
B := R(A)  . . . . . evaluate destination
(B) := (A)  . . . . . transfer message to destination
output B  . . . . . output message
```



Data switching

Figure 9.

Some instructions may spawn several outputs, if they fail (e.g. R(A) undefined), or if some options are exercised, like tracing the transfer path. Internal states are mainly used for internal control of the machine (monitoring, maintenance) or as a result of failure diagnosis. Other instructions may act upon internal states, which can be thought of as switches, special registers, or panel lights, in a conventional computer.

Describing a communications network as an abstract machine is a way to draw a line between its functional specifications, as visible from an external user, and its physical implementation. Of course, it would be non-nense for a designer not to have precise ideas about the mechanisms underneath. But as far as possible, inner workings should be kept invisible at the functional level, because computer installations on one hand, and the communications network on the other hand should conserve maximum autonomy.

Opening the black box discloses a new dimension, (Fig. 10). Our abstract machine is actually composed of a set of components, say processors, which must be put to work in a coordinated fashion to get a message through. In other words, executing an instruction of the global machine is a multi-processor action. Then it might seem that well known process coordination techniques would apply nicely. Actually they don't. Although we are dealing with an abstract model, we are nevertheless in a real world. There is no common store, no indivisible operation, no well-mannered processes. Components are geographically scattered, and they may fail. In this machine there is a communications problem between its own components. We are facing a more general case of multi-processor, i.e. a distributed machine.

Rather than attempting to devise a complex model of a distributed machine, an easier and more powerful approach consists in breaking it down into simpler components along its geographical distribution. Each node is again a local machine, whether it be uni or multi-processor. Its abstract model is identical to the global one, with an instruction code, I-O, etc... Messages are instructions as previously. Executing an instruction of the global machine appears now as a sequence of instructions on local machines. Each one performs a step of the global instruction, and outputs a result towards a neighbor machine. The process propagates up to a point where output is directed to the external world, and there it stops. (Fig. 11)

Actually this process can be other than a sequence of local machine instructions. Indeed, components may fail, particularly the communications lines, and a local instruction may be repeated a number of times, with different results, because internal states may change. Thus, outputs may be sent to various neighbors, and the set of local instructions becomes a tree, with loops and branches executing in parallel. In the end there may be one, none, or several results. It is clear that some error control scheme is needed to make our machine reliable.

This global instruction moving step by step through successive local machines is reminiscent of the pipe-line structure of high speed processors, or time-slotted multi-processors (CDC 6000 peripheral processors). The similarity is not accidental. With very fast and cheap micro-components, future machines will contain hundreds or thousands of processors. They will be micro-networks.

As seen previously, the local machine model is the same as the global machine. Our structure is recursive. But only one level of recursion is not much after all. Let us carry this approach one step further, by applying the distribution concept

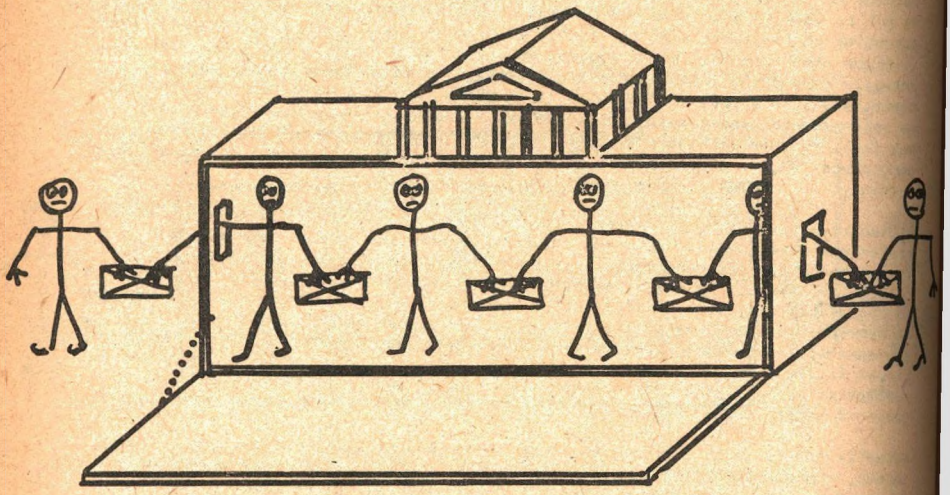
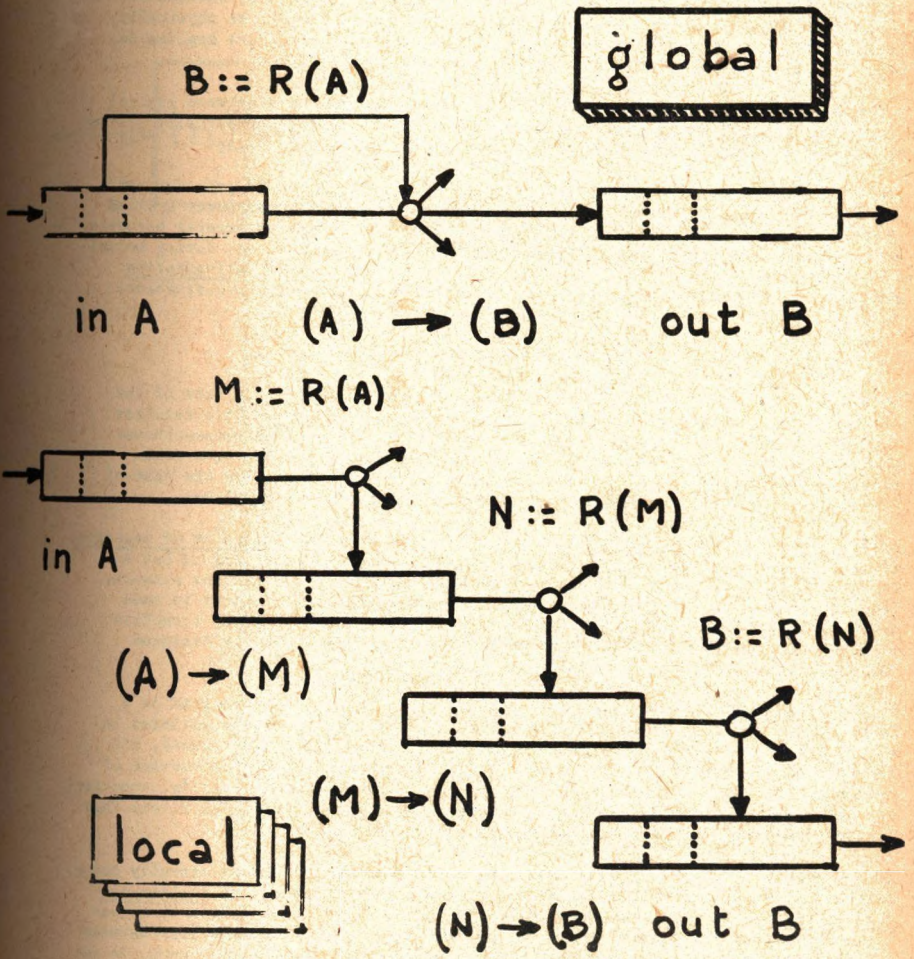


Figure 10.

INSTRUCTIONS



77
Figure 11.

to the instruction code. In conventional machines, the instruction set is closed, due to the number of bits reserved for the op. code. In our network machine we can consider that each instruction type is executed by a logical component, located somewhere in the global machine. An op. code is then simply the address of the appropriate instruction processor. Executing a global instruction in a first phase propagates it toward its specific processor, where it is executed with possible further propagation. By putting logical network components within the destination name space, the instruction set can be as large as desirable without practical limits. In other words, the instruction set is open-ended. This is a nice feature for an experimental system. The network machine is now defined in terms of logical (software) components, instead of local (hardware) machines. Practically these components have to be physically located within some or all nodes. Consequently, the same structure applies as well to a local machine (node). (Fig. 12)

A consequence of this regular structure is that communications between logical components (software machines) is the same whether they be geographically distant or located within the same node. But there is a difference in delay.

Another consequence is that several networks of that type can be put together and make up a single network. This is the recursive structure property applied outwards. Indeed, every network can be modeled as a node, hence several networks are a network. Partitionning and coalescence of networks may occur during normal operation as a result of line failures. Therefore this sort of self-adapting structure is most desirable, as it saves on down-time and service disturbance.

B - Hosts

So far we have used the term "installation" to mean vaguely a computer of the set making up a heterogeneous general-purpose computer network. This requires further attention, as we know that computers can reveal various metamorphoses. Since the Arpanet literature introduced the term "host", it has been used widely, although not always in the very same sense as Arpanet. For the sake of consistency, we use it here also, in some loose sense, as we shall see.

As seen from the communications network, a host is a source and a sink of messages. It is able to input and output messages of a predefined format, and it has an address, i.e. it is known within the name space of the communications network. This is enough, but one can notice that not the slightest assumption is made about the logical or physical nature of a host. This means that the structure of a computer network is totally independent of that of a properly designed communications network.

At the host level, discounting some special cases, no one is interested in a computer per se, because it is not a usable logical entity. Usable entities in a computer are resources : files, processors, peripherals, user accounts, etc ... In order to get some productive work, activities are run to which a sub-set of resources are assigned, exclusively or in some shared mode. Customarily, computers are used in a multiplexed fashion, allowing several activities in parallel. Thus a host is merely a container of resources and activities.

In a computer network environment, useful host-host communications actually occur between activities, in order to access remote resources, or set up distributed activities. Due to the heterogeneity of the hosts, there is no common or compatible definition for a local activity. On the other hand, communications between hosts cannot take place unless some mutual agreement has been agreed upon. To get around this dilemma, we introduce a new kind of network-wide entity called a subscriber. They are purely abstract things, bearing a name known by every host, which activities can plug into when they want to start some host-host communications.

Data switching machine

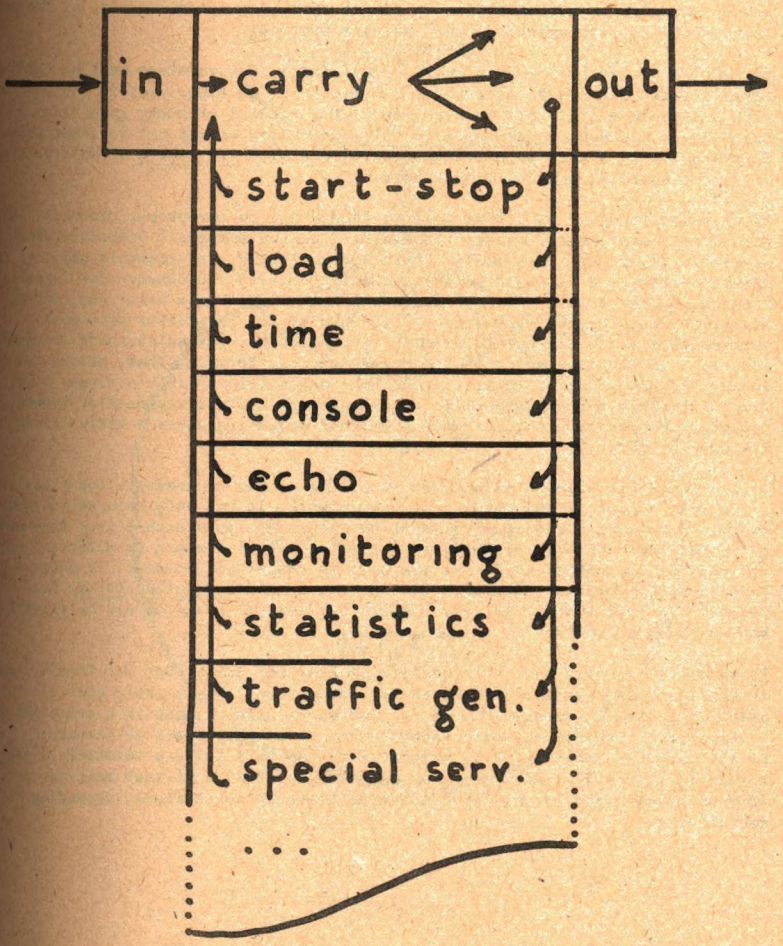


Figure 12.

In many ways subscribers are like phone sets. (Fig. 13)

The number of subscribers per host should be large enough to accomodate the needs of simultaneous network activities. But if we want to avoid intricacies and inconvenience associated with a dynamic assignment, subscriber names should be assigned permanently to real world entities such as users, terminals, services, etc ...

Usually, all messages exchanged between a host and the communications network are multiplexed onto the same transmission lines, typically two for reliability. Thus a logical component is necessary to collect and deliver messages fro and to the proper subscribers. It is typically a host operating system module, of the access method species, implementing "the" network protocol.

However, a unique protocol is quite constraining. New versions must be tried out occasionally, user clubs want their own protocol, an experiment is contemplated with a host in a different network, and for a handful of other good reasons, a host should accomodate various protocols, which may or may not be compatible. This results in a more sophisticated structure with two levels of multiplexing, one for protocols, one for subscribers.

If messages received by a host must be fanned out to the proper protocol, this means that some general convention has been agreed among all protocols about the selection algorithm. This would not be too realistic, as protocols may just happen independently, and it would be in contradiction with the concern for installation freedom. Consequently, we use a different scheme. Within a host there can be several virtual hosts, each one with its own set of consistent protocols. Incompatible protocols are in as many virtual hosts, for which there are different addresses in the name space of the communications network. Thus, messages are dispatched according to their virtual host destination within a single host. (Fig. 14). This technique applies as well when a host is composed of several computers hooked together, or when a host is operating under a virtualizing system like CP-67.

So far there has been no assumption about the number of transmission lines used to connect a host to the communications network. It has only been mentionned that two would be appropriate for reliability. It is not said either that they should go to the same node. For reliability it is better to connect to different nodes, but this is immaterial as far as the host is concerned. However, it matters to have several lines, particularly if the host is a distributed system, i.e. a network. In this case it is very likely that several paths should be established between both networks.

To recap, it appears that the concepts of host and subscriber are flexible enough to elude any precise definition, except being names. Practically they can be associated with a variety of objects. Let us say that a host is a container and a supplier of resources, while a subscriber is usually a set of resources intended for an activity. E.g. a subscriber can be a file, a terminal, a user, a sequential process, a sub-system, an operating-system, a virtual machine, etc... Some entities might as well be considered hosts or subscribers, depending on convenience.

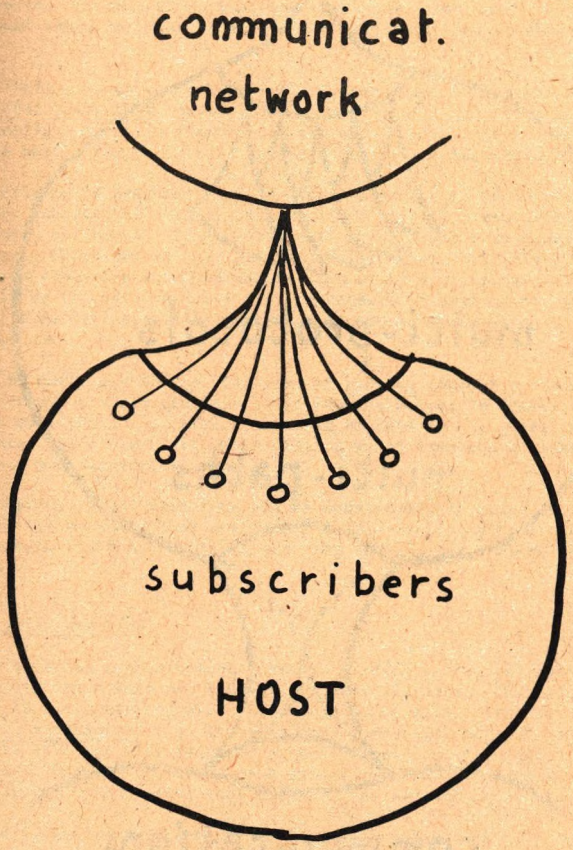


Figure 13.

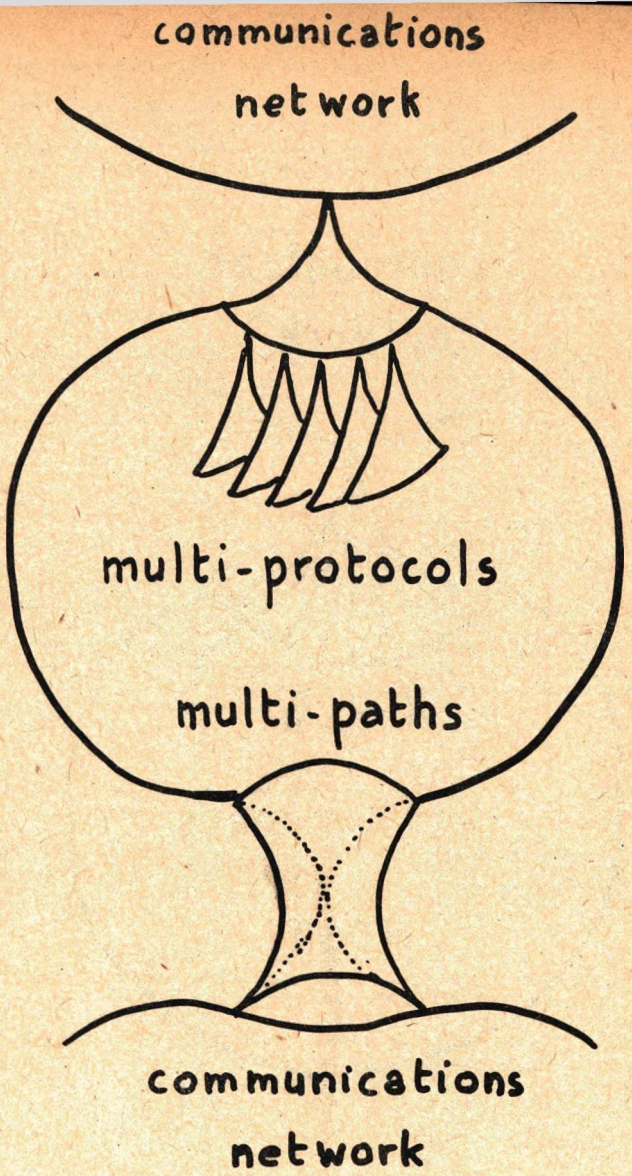

communications
network
multi-protocols
multi-paths
communications
network

Figure 14.

Using distant resources, communicating with distant processes require naming them. But since hosts are heterogeneous, there is no consistent scheme to name distant resources. Furthermore existing names may conflict. A first attempt to circumvent the problem is to define a new name space global to all hosts. This may be the case for subscriber names. Therafter, every host organizes on its own way a mapping of its local names onto the part of the global name space allocated to it. But this means that one must know beforehand how a distant host maps its local names in order to access its resources. Again, this is heterogeneous.

A second technique is to use a hierarchical name space. Names are 2-component, one is a global name (host, subscriber, or any other), the other is the local name designating a specific resource in the host naming scheme. This method is usually more costly for computers, but more convenient for human handling.

There is a dilemma in networks when it comes to locate an entity from its name. Due to delays inherent in communications, it is costly to search for a name in several or even all hosts. Consequently one tends to make up names with a host name component, which serves as a geographical pointer. But this scheme is not well suited to resources which belong to virtual or distributed hosts. There is more flexibility if resource names are independent of host names. But to reduce search time it is desirable to reinstate a geographical locator, which we may call a region name.

Regions can be defined with the objective of cutting through a minimum traffic, so that region and traffic clusters would roughly match. In addition, within the confines of a particular region there is no need to use global names, since the region name can be implicit. Both effects would compound to make most names shorter, hence more convenient.

Other more sophisticated variations may be considered, in order to simplify naming across the network. They are inspired from dialing schemes in the telephone system, which are quite interesting indeed. See a phone directory for more details.

D - Communicating entities

In a message oriented communications network sending individual messages is a basic capability which does not require any initial set-up. This can be put to an advantage in a computer network when cooperating activities exchange short self-identifying messages, e.g. transactions, samplings, control information. In this case, the only entities required at host level for activities to communicate are subscribers. Once an activity has a subscriber name assigned to it, it can just send messages to any other subscriber in the network under the same protocol.

But more traditionally, interchanges between activities tend to be modeled after sequential I-O. It is a transposition at network level of programming systems designed for sequential devices. In effect most inter-process communication schemes are data stream oriented, stemming from an I-O-minded approach. Thus it is convenient to have mechanisms at host level geared to data stream handling. For that purpose new kinds of entities are required, to provide something like data pipes between activities. There are a number of possible variations in the functional design of such tools. Also the terminology is not stabilized. In the following we designate by pipe such an entity capable of channeling a data stream between two distant activities, and ports the names used at each end by host activities. Here are some typical pipe properties which may be encountered.

. Uni - or bi-directional :

Since a data stream always requires some feedback for error and flow control, it is most convenient to have bi-directional pipes.

. Half - or full-duplex :

On bi-directional pipes full-duplex mode is much more simple and efficient, as it does away with the contention problem that plagues half-duplex procedures.

. Sequential :

This is the property of delivering messages in the same order as they are put into. For sequential data streams, sequencing is mandatory. But a pipe may be used by a sub-system which contains its own scheme for tagging and controlling the message flow, in which case sequencing is not necessary.

. Error control :

It is the ability to insure that every message sent has been received only once. It can be a by-product of sequencing. It is not questioned that error control is needed at some point when data travel from host to host. But where to put it is an issue far from settled. There is more elaboration on that in the following.

. Flow control :

It is a mechanism whereby the receiver can choke up the sender to prevent overflooding. Same comments as for error control.

. Parameters :

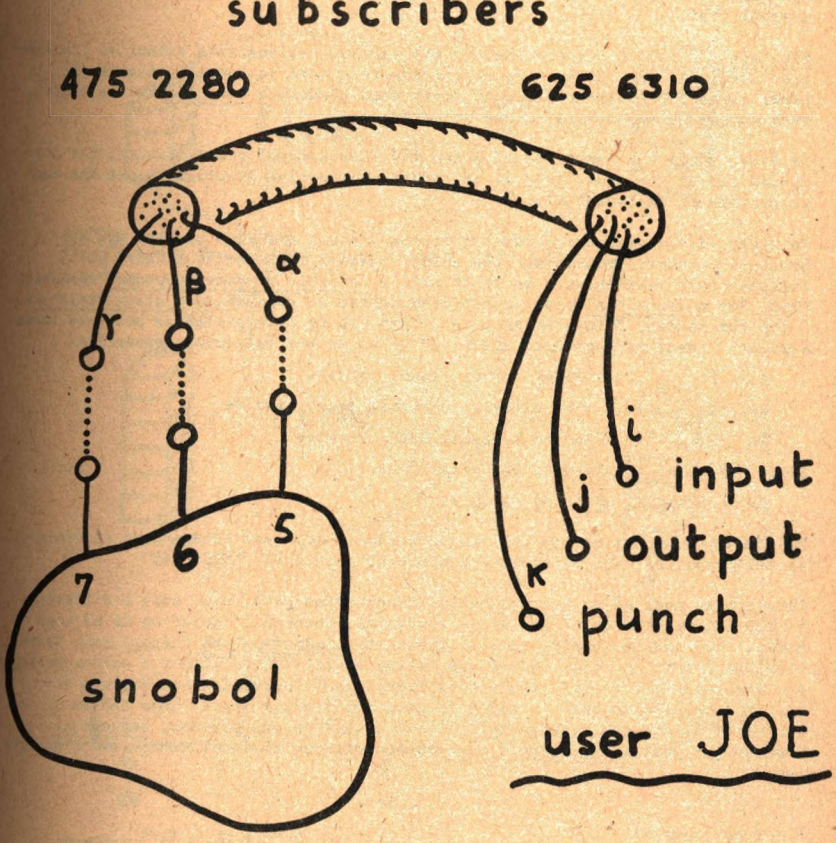
To regulate the data flow and tune it to the amount of resources available to both sender and receiver, it may be appropriate to associate parameters such as : message length (mini. + maxi.), traffic rate, time-outs, etc... These parameters can be negotiated between both parties as part of the pipe initial set up.

It should be emphasized that pipes are only a construction implemented at host level. In some cases they are extended end to end through the communications network. The implication is a strong coupling between the design of host protocols and communications network. This may be justified for specific applications (e.g. Tymnet, a service bureau time sharing network), but in the general case, this is in contradiction with our principles of insulating the two domains, due to the undesirable constraints attached.

Data pipes are to be set up (opened) and destroyed (closed) at the request of host activities. As one can figure out easily, it would be a tricky problem if both activities had to agree on a common pipe name, because there is no referee to break a tie up. It is more convenient to name a pipe from both ends with local names as seen from each activity. These are port names. Each activity refers to a port when sending a message. But this is its own local name. In order to deliver messages from the appropriate pipe at the other end, they must arrive with the destination port name. Hence, port names of both ends must be exchanged between hosts during initial set up.

Pipes and ports are particularly useful when host activities are programmed with I - O streams left unassigned to any specific device, and can be referred to by labels, say by port names. Using a command language, or some form of declarative statements allows a user to bind network and program port names at execution time. Depending on his requirements, a user is able to use either local or distant files and I - O devices, just by assignment commands. (Fig. 15)

PIPES



PORT names

Figure 15.

E - Error and flow control

These functions are often associated whilst they may appear of distinct nature. We shall see why. Error control means detecting the loss, or alteration, or undue repetition of messages. But detecting errors only would not be of much avail without recovery. Thus a satisfactory error control mechanism should include a procedure to restart transmission in a proper way. Any recovery procedure assumes that the sender has kept copies of a certain number of past messages. Since they cannot be stored indefinitely. part of error control consists in a procedure whereby the receiver releases back-up copies of received messages stored in the sender's space. All that boils down to a producer consumer process. The sender produces copies of messages, that are gradually consumed by the receiver, once it has exercised its error detecting and recovery capabilities.

Flow control is a procedure whereby the receiver allocates a potential transmission credit to the sender ; no matter the form used to specify this credit. In other words the receiver produces empty cans gradually consumed by the sender, once successful transmission has occurred.

In other words the semantics of error and flow control are different; but the syntax can be the same. Thus it is common engineering practice to use the same mechanisms to implement both.

To recap, discounting some idiosynchrasies, error and flow control are two producer-consumer procedures. The actual message transfer is a third one. Together they make-up what we might call a reliable producer-consumer communication. For a sequential pipe, the procedure can be reduced to 6 state variables, 3 for the sender, 3 for the receiver. They can be interpreted as pointers in an infinite linear message name space. If we adopt the following conventions :

Fs , Fr . . . . Flow control, send, receive

Ts , Tr . . . . Transmission control, send, receive

Es , Er . . . . Error control, send, receive

these variables should meet the conditions :

$$Es \leqslant Er \leqslant Tr \leqslant Ts \leqslant Fs \leqslant Fr$$

This is a generalization of the well known producer-consumer scheme used in conventional computers, and often termed circular pointers. (Fig. 16)

This scheme applies to non sequential transfer as well, with only a difference in the interpretation of the state variables. Instead of pointing to message names, they can be understood as pointing to windows in the message name space, with the condition that windows must slide sequentially. Within a window messages are managed individually. The window width can be variable, and depends on resources available both in terms of name and storage space. This can be a parameter negotiated at pipe initial set up. As we can see now, sequential transfer is just a degenerated case of non sequential, with window width exactly one.

F - Event control

Controlling message transfer is in effect controlling specific events to which we attach the meaning : message allocated, message sent, message checked, etc ... But any other meaning would be acceptable as well, as far as control mechanisms are concerned. We have an example in the control of non sequential transfer, where the semantics "message window" have been substituted to "message".

Error & Flow control

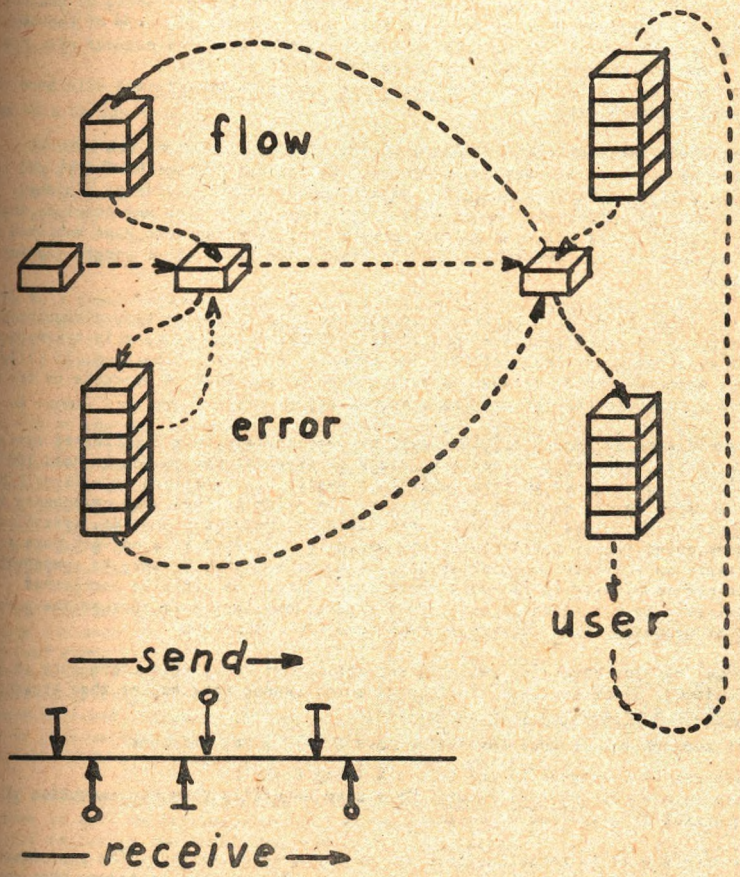


Figure 16.

Event transfer is a keystone in distributed systems, since there arise constantly some points in time when several distant components must all agree about the fact that something did or did not happen correctly. E.g. transferring a file, executing a job, updating a data base, etc... Both alternatives are equally acceptable : either it did happen and all activities can go on, or it did not and they all have to go to a recovery procedure. The unacceptable situation is some activities deciding that it did, others deciding that it did not, or don't know. This we may call a distributed event ; it has to be either true or false.

Since there cannot exist an ideal observer, event setting requires propagation, hence delay. Furthermore, communications components are unreliable. Consequently, the following situation can never be ruled out : "After all sorts of control, "an activity decides that an event is true, and updates its copy of the event. "Assuming that control is totally reliable, another activity decides also that "the event is true, but before updating the event, it fails".
Then the event is "trulse". A unique event cannot be ascertained with more than a certain probability.

If an event is seen false by some activity, due to failure in the control mechanism e.g., a recovery procedure will be entered by that activity, and the others will be solicited again for an action that they considered finished. If the event was supposed to occur only once, these activities may conclude that it was "trulse" after all, and go to recovery. But they may also have vanished under the assumption that everything was all right, and there is no longer any way to recover.

On the other hand, if a series of repeatable events is expected, confusion may develop as to which instance is actually occurring, unless care is taken to make each event uniquely identifiable. For example, A opens a pipe to B. It fails for A but succeeds for B, because an acknowledgment got lost. A proceeds to try again, and B interprets that as opening a second pipe.

It would seem as though nothing could be ever certain in a distributed system. In a strict sense this is correct, since no physical system can be held 100 % reliable. But we used to term reliable a system whose failure rate is well below a tolerance level, whatever that may be. The point is that some components in a computer network are known to be unreliable by computer standards, specifically transmission lines. Subjectively one would be satisfied if a network were as reliable as a single computer system, or maybe somewhat better, to compensate for a larger complexity in recovery procedures. Since the pitfalls mentionned previously do not normally occur in a single computer, some appropriate procedures appear necessary in a network to keep them from happening.

A usual way for obtaining reliable service out of unreliable components is redundancy. Since a single distributed event cannot rate better than trulse, we introduce a precedence rule as follows :

" if event E (n) is true then all events E (i), $i < n$, are true".

which can be condensed in the form : $E(n) \Rightarrow E(n-1)$.
These we may call chained events. They stem naturally from the execution of sequential or iterative processes.

The onus now is to establish that at least one event is true. The precedence rule allows to do just that, as we shall see. But we have to assume that components are reliable enough, so that :

. events cannot be "undone", once true they do not change.
. there cannot be "ghost" event reports.

Let E (i) be a distributed event in a series of chained events, with local components $\{E_j(i)\}$ in the domains of a distributed activity $A = \{A_j\}$.

... procedure executed by each local activity :

```
procedure sync (i)
    E_j (i) := true ; report E_j (i)
    wait until all E_k (i), k ≠ j, reported true
```

and the structure of each A_j activity would look like :

```
...
step (n)
sync (n)
step (n + 1)
sync (n + 1)
step (n + 2)
sync (n + 2)
...
```

From this we can infer :

- if sync (n) is carried out successfully, then E (n) is true, since every A_j has reported so.
- if sync (n + 1) is blocking for some A_j, then no A_k can progress beyond sync (n + 2), since there will be no E_j (n + 2) report. Consequently, E (n) = true, E (n + 1) = E (n + 2) = trulse, E (i > n + 2) = false, as seen from A_j.

Using this synchronization scheme, any A_j can determine safely the state of the distributed activity, with a range of uncertainty restricted to 2 steps. In case of failure, it is thus possible to guarantee a safe recovery from a well defined point of execution.

Let us remark that uncertainty is not a new phenomenon peculiar to networks. Some way or another it has to be dealt with in any system capable of truly parallel activities, for which there is no ideal observer. Those persons familiar with multi-processors have already encountered such situations where it is impossible to determine the state of some variables, without bringing the system into another state in which they can be observed safely. This is not just an effect of the component unreliability, but a basic ingredient of asynchronous systems. There exist "fuzzy" states which must be carefully segregated and analyzed, instead of being fought out with makeshift logic.

With our chained event scheme, a distributed activity can progress safely. But we have not seen how it can terminate safely. We have to agree that there must be a last event past which activities can terminate. Since an event state cannot be ascertained by prior events, the last two events may be left trulse, with some activities getting blocked, and going to recovery. If some other activities have already vanished, recovery will not succeed, and hung up activities will have to give up on some way, lest being deadlocked. On the other hand if we make it acceptable that sync (last) may fail, is it worth to execute it at all ?

Let us assume that the last event is only reported without sync, then no A_j activity is able to determine whether E (last) is true, i.e. whether A really finished correctly. An ideal observer would be necessary to collect all reports and investigate missing ones. In effect it would perform a centralized sync operation, which would be no different from any of those performed by each A_j. In particular a report may be missing from some A_j that finished correctly and is gone. It appears that for this scheme to work we need a controlling environment

embedding A, to which E (last) would be reported. Actually this makes E (last) belong to the controlling environment rather than to the distributed activity A. Thereafter this environment could use whatever ad hoc technique suitable to the problem at hand, like keeping track of E_i (last) reports in local A_i environments for as long as necessary. This may work after all, but we did not actually solve the problem, we only threw it in someone else's hand. Practically the controlling environment is nothing but a distributed activity, for which E (last) can be just one event in a chained set, and thus ascertained by subsequent occurrences. In the end we have to assume a global network environment which will never terminate its execution. For all practical purposes, this may be satisfactory, but it still leaves a mixed feeling that a distributed activity be unable to insure its normal termination.

Going back to our synchronization scheme, we can take a different approach. Let E (n) be the last event in A marking off the end of the useful execution. We introduce two more final events and two more sync operations through which each A_i must step before normal termination. Should they all succeed, one may say that it were useless, because it did not add anything to what each A_i could tell after E (n). But this is different if some of them fail. The dilemma for an A_j goes like this : "I know I'm finished. I know you're finished. But do you know I am, so I can go home".

According to the synchronization logic, we may state that any A_i having successfully executed sync (n) may conclude that the activity A has been safely terminated. In case of failure on sync (n + 1) it should attempt recovery, because it may unblock some A_k latched on sync (n). If recovery does not succeed, or if sync (n + 2) fails, A_i can nevertheless go home safely. In other words, failures of E (n + 1) or E (n + 2) do not cause A to fail.

Failure to pass sync (n), for some A_i, is an unsafe termination. If recovery does not succeed, it may be that terminating A is impossible without external intervention. It would be the same thing for any failure prior to n. In this case, trouble reports must be directed to a controlling environment for further analysis and recovery. Thus one still has to resort to a controlling environment, as in any conventional system, but its role is limited to exceptional conditions, as it should be.

G - Time-outs

As we have seen abundantly, due to the potential failure of some components, a distributed activity cannot rely upon well behaved processes. This is also true for any large scale and complex system in which logical flaws cannot be completely eliminated. Since component activities are dependent on the correct propagation of various information, they must always be in a position to escape from deadlock if the expected information does not arrive. Regardless of the implementation (watchdog, loop, manual intervention), it boils down to some time delay beyond which an activity is forced to continue. What is appropriate to do is of course peculiar to each situation.

A time-out may be thought of a nil event which always occurs in lieu of an expected one. But it carries no reliable information, only doubt. Indeed, if delays had to be set so there would be no significant probability to observe what was expected, they would often be too long and deteriorate efficiency. On the other hand, if they are too short, exceptional conditions would be triggered too often and again increase overhead. Consequently, a first property of a time-out delay is to be tuned in relation to a particular environment. It does not mean that the expected event did not or will never occur.

Typically a time-out triggers some diagnostics or recovery procedure. But the timed-out event may have occurred already, or can be under way. Consequently, actions started in that context must be arranged so that they do no harm when they are redundant.

Usually time-outs are placed in several components cooperating asynchronously. This results in time dependencies requiring a thorough analysis. As in feedback systems, oscillations and vicious loops can kill normal behavior. Unfortunately this is an area where we lack a systematic approach. Constructions must be validated by hand, or computer simulation, and only simple ones lend themselves to practical analysis.

A very restricted example appears on Fig. 17, which represents a sender and a receiver automaton, with some suggested time dependencies. As an exercise readers will likely find that they would pick different ones. The reason is that one can make quite different assumptions about the operating environment. It may also happen that the environment is unstable. Conclusion : time-outs are a tangled issue.

H - Multi-level error control

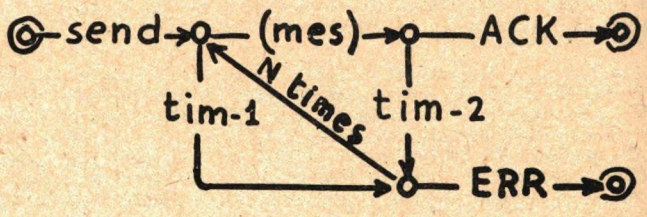
As seen previously, a distributed activity, as any other, needs a controlling environment to which it can report unrecoverable or doubtful situations. Furthermore, each component is necessarily restricted to a domain in which some types of errors cannot be detected, because it would require access to extraneous information not relevant to its activity. E.g. the communications network cannot tell whether a message is a correct command for an RJE service, (remote job entry). Therefore control must be distributed across several levels of environment, starting from inner level components (transmission lines), through logical components of the communications network, host protocols, user activities, network operating system.

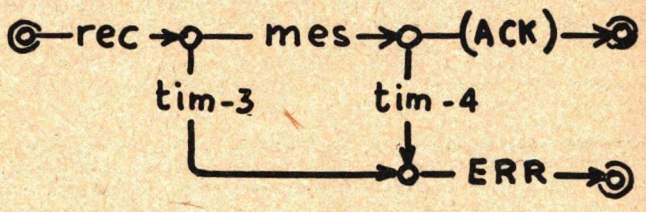
Since control means cost, or overhead, the objective is to make it cost-effective. In other words it should be implemented where it is most efficient, and it should not be redundant without serious justifications. Consequently, a logical approach consists in specializing components in some types of control which they can perform well. Errors that are detected but cannot be corrected with enough reliability should be reported to the outer layer, while inner layers should be trusted for whatever control they are in charge of.

When propagating information, a first type of control is to make sure that bit patterns are not corrupted. This is typically the sort of function which a hardware logic using cyclical redundancy checking (CRC) can do efficiently with high reliability. On the other hand, other methods are more expensive. Thus, bit control is to be entrusted to I-O adapters. But they usually do not correct errors, because it requires some programmed logic. This second level of control is devoted to the I-O adapter environment, i.e. a transmission procedure, which corrects errors by repeating corrupted messages.

At a further level, messages may be assumed correct as far as bit patterns are concerned. But due to the parallel and adaptive structure of the communications network, some messages may be lost or duplicated. Another level of control is thus required. It may be located within the communications network, but since it requires some form of end to end feedback, messages must be stored, identified, acknowledged, and the corresponding machinery appears far too involved if messages can flow out of the communications network to a host through several nodes, (this was one of our network characteristics).

TIME-OUT





$$tim\text{-}1 \ggg \Delta tr. \text{ (trans. del.)}$$

$$2\,\Delta tr \ll tim\text{-}2 < tim\text{-}4 + \Delta tr$$

$$tim\text{-}3 \ggg N.\,tim\text{-}2 + \Delta tr$$

92

Figure 17.

At another level, messages may have to follow some definite protocol implemented in a host. There message formats and contents are checked, and if not applicable, recovery is attempted. This is actually a redundant form of message control, which would not be necessary if host implementations and message transfer were reliable enough. Since communications network control cannot extend up to buffers in the protocol software, there appears a gap in message control which justifies somehow a duplication of function. But one might argue that message control should only be located at host level, because it is necessary anyway, and it encompasses all inner controls at a marginal cost. However, some message control along transmission lines may reduce error correction by hosts, but it does not have to be held reliable.

For some types of applications (data pipes) messages must be delivered in a correct sequence. Again, this implies message control. On the other hand sequential delivery to a user does not imply sequential delivery to a host. Reordering can be done by the receiver protocol, together with message control. This is further reason to put message control at host level.

More and more levels could be examined. They would take us into specific functions such as terminal control, file handling, etc ... Perhaps there is already sufficient matter to leave it up to reader investigation.

I - Fragmentation

What we call fragmentation in the context of computer networks is the breaking of inter-activity messages into smaller pieces, so that they fit into communications network packets. This is desirable because transit time is shortened and buffer requirements are smaller. On the other hand original messages must be recomposed before delivery to the destination activity. This is called reassembly.

It is clear that between a 2-char. control message, a 30-char. transaction, a 136-char. printer line, and a 50 Mchar. file, there cannot be any satisfactory "standard" message length. Even if packets can accomodate a few hundreds of characters, this will not cover 100 % of cases.

If fragmentation is performed in the communications network, all packets composing a message must be reassembled at the destination node. Control mechanisms are necessary to allocate storage and prevent deadlock between several messages. Furthermore, the maximum message length is still too small for bulk data transfer. Consequently, another level of fragmentation is required at host level.

Another approach is to do all fragmentation at host level and have the communications network carry just packets. Unquestionably this reduces significantly the complexity of message transfer, but it may increase the overhead associated with any host I-O. Some blocking scheme may allow I-O to take place for a bunch of packets, as if they were separated in time. Chained I-O channel control words can usually facilitate that shortcut. Another way is to make network I-O autonomous by using dedicated interrupts and software, or a front end processor having direct memory access. Indeed I-O overhead is by far an operating system syndrome.

J - Congestion

We use this term to mean a pathological situation when a communications network is swamped with messages which cannot find their way out. It can be a local congestion due to a broken link or a dead host, or a total congestion due to an excess of accepted messages. In many respects this can be compared to automobile traffic jams. This is presently a subject on which intensive research is pursued. More specialized papers are recommended to get a thorough treatment.

IV - OVERALL ARCHITECTURE

A - Abstract machines

Although we want to insulate the communications network from host software idiosynchrasies, the same design approach can be used successfully to define host components in terms of abstract machines, (see H. Zimmermann's paper). Instead of having a specialized homogeneous system, we have a general purpose heterogeneous one. Consequently, the emphasis is put on creating network standards out of software components, in a way that allows indefinite expansion and evolution. This is a layered structure, instead of a recursive one.

B - Host components

Starting from the communications network, we find a line transmission procedure, in charge of exchanging messages core to core with a host. Each line is controlled with a single procedure, but there may be various procedures for different lines. (Fig. 18)

The next level is a single communications interface which carries on a conversation with the data switching machine. It formats instructions (messages) handed over to a line procedure, and receives results (messages or status codes) output from the network. In case the network interface might change, all modifications are localized at this level of machine.

Above that we find protocol levels. It seems that flexibility suggests to have a set of basic protocols organized in a 2-layer fashion. The lower set implements basic conventions peculiar to a distinct computer network, or a private club. Since a host may belong to different networks, one needs several protocols. These protocols are oriented towards host exchanges. Basically they implement a multiplexing of host subscribers. (Fig. 5)

The upper basic layer is data communications oriented at user or activity level. It provides for various means of handling data fields expressed as user entities rather than network messages. Except for some incompatibilities, data transfer protocols can use several host exchanges, assuming they can get a counterpart in a destination host.

Going further up, we enter an area of specialized protocols oriented towards specific applications, services, or devices. Examples of this sort are file transfer, job handling, data base management, virtual terminal (an ideal standard terminal). They can be more or less intertwined, and use each other capabilities.

At any level above the communications interface, more protocols can be added, including those developed by users for private experiment. The practical limit is the confusion stemming from too many protocols poorly documented, etc ... There is clearly a need for a few well engineered protocols, satisfying most user requirements, and properly maintained by a responsible organization. These would be network standards.

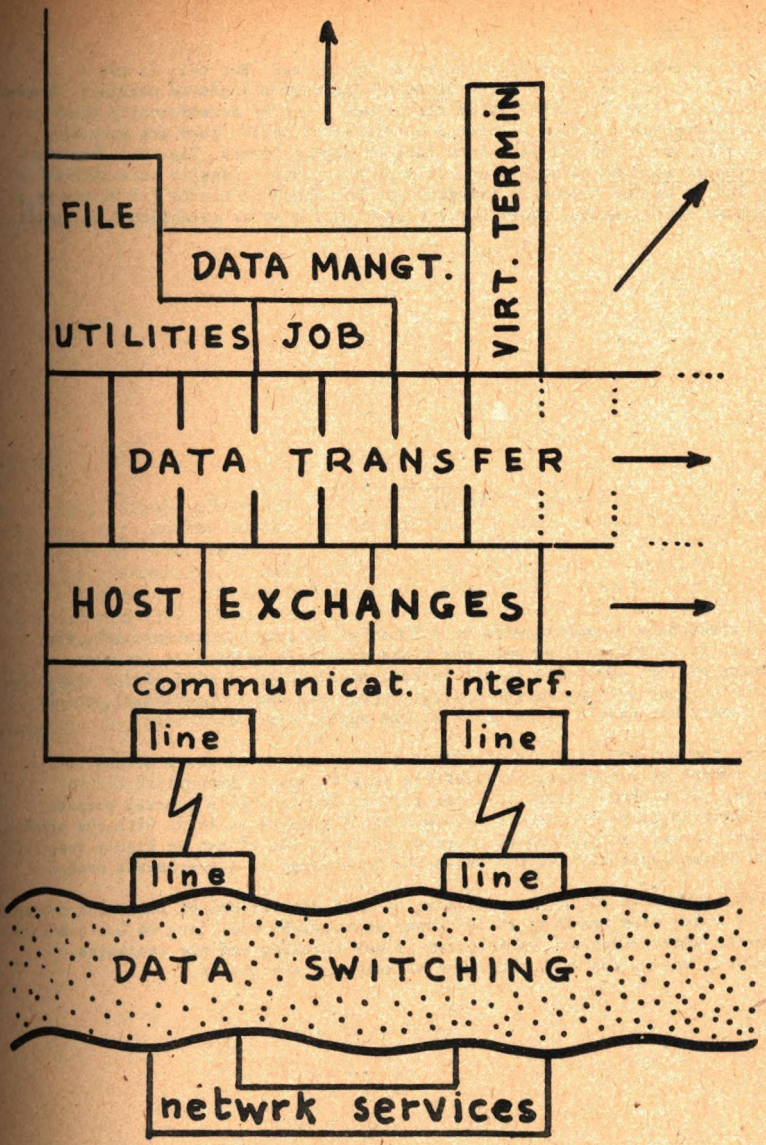

FILE
DATA MANGT.
VIRT. TERMIN
UTILITIES
JOB
DATA TRANSFER
HOST
EXCHANGES
communicat. interf.
line
line
line
line
DATA SWITCHING
netwrk services

Figure 18.

C - Distributed machines

As we have seen, machines proliferate at host level. But this is not a distributed machine, it is a hierarchy of distinct and nested machines. However the concept is still there. If we single out a set of functionnally identical (say isomorphic) components located on different hosts, they are mutually compatible and offer the same interface to higher levels. Together they are a distributed machine network-wide, at some host level. Ideally this machine is supposed to be functionnally homogeneous, but since it has to fit within heterogeneous environments, equivalence conventions must be established on exotic hosts.

V - FINALE

These notes have no ambition to be exhaustive or even consistent. Like the subject, it is distributed and open-ended discourse about what computer networks are, or could be, in our instant fuzzy state of mind. Since they nudge at conventional well established structures, they are controversial, and in this domain one should pay a blink at the author's personal bias.

Besides the obvious facility of giving a large number of users access to a large number of computers, considerable work is yet to accomplish before networks can be made useful on a casual way. The idea of a general purpose heterogeneous computer network may turn out a chimera at least with the present and next to come generation of computer systems. But it can be highly successful for selected applications, for which the technology is nevertheless needed.

Network structures are already forcing us into new visions of tools and concepts, some of which were reaching at religious statute. We will have to learn how to integrate uncertainty and parallelism in our thinking and our languages. Communications are the next challenge in computer structures.

AN INTRODUCTION TO DATA NETWORKS

Louis POUZIN
Directeur du Réseau CYCLADES
Institut de Recherche d'Informatique et d'Automatique (I.R.I.A.)
78150 - ROCQUENCOURT (France)

ABSTRACT

There are a variety of network types according to their structure or their use, e.g. packet switching, or computer network. Whether they are homogeneous or heterogeneous, their general structure is the same : processing, transmission, terminal distribution. Packet transmission is a new technology, with substantial advantages, but also with some adaptation problems. Communications between terminals and applications require conventions (protocols) which will be integrated in future computer systems. Existing systems may be adapted with converters (gateways). Networks are now technically accepted as a tool for resource sharing. They may have much more potential in the domain of human communication.

TYPES OF NETWORKS

Networks have been around for a decade. However, their characteristics were somewhat restricted, and usually tailored to a particular usage. Also, the term "network" being presently successful tends to be used to designate constructions that barely qualify for that. But this is not so important, as long as it is made clear what sort of network is considered. Here follow typical examples :

Network of terminals

A central computing facility (one or several computers) is linked to a score of distant terminals through a tree-structured set of communications lines. Multiplexers and concentrators may be used along the way, to improve communications service and reduce cost. Ex : any large computer installation.

Message switching

One or several computers used in a store and forward mode to transport messages point-to-point. It is a very specialized computer network. Ex : SITA network.

Packet switching

It is a particular variety of message switching, with very short transit delay. Ex : SITA high level network, or ARPA sub-network.

Computer network

Several interconnected computers carrying distinct tasks in a cooperative manner. This type of network can also be categorized according to some of its characteristics.

General purpose computer network

It is a category within the previous type. Tasks submitted to the network are not predefined. They can use resources such as : compilers, files, peripherals, storage, etc. Basically, the rationale for this kind of network is resource sharing. Ex : ARPANET, CYCLADES.

Special purpose computer network

Even though computers may be used individually for a variety of services, only specific applications are accessible to network users. Ex : information dissemination.

It is clear that the term "network" will gradually lose its glamour, as practically every computer installation will become more or less part of a network.

This term will be used to designate any kind of network composed of processing elements, transmission elements, and distribution elements, (Fig.1)

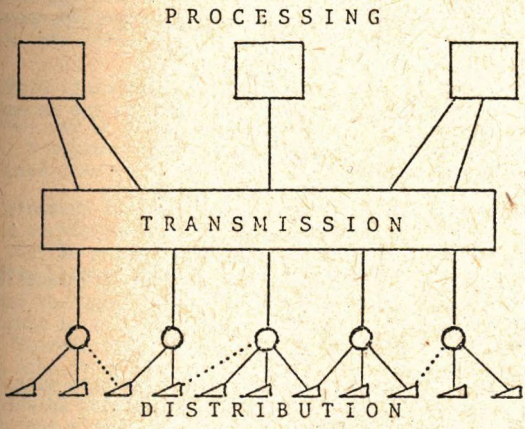


Figure 1 - INFORMATIQUE NETWORK

This ternary decomposition is typical in many industrial and commercial systems. In terms of informatique, processing elements are computer systems; transmission elements are communication lines, modems, or a specialized network; distribution elements are terminals and controllers.

It should be noted that this type of structure, where functional levels are carefully delineated, appeared quite recently. It may be observed in ARPANET, CYCLADES, and new products announced by computer manufacturers (DEC, IBM). Previous networks implement all these functions, but their structure is so confused that it is difficult to distinguish elements performing well defined basic functions.

An informatique network is homogeneous when all of its elements have been designed to be compatible. This is the case when the whole network comes from a single supplier, or when plug-to-plug compatible elements come from other vendors.

An informatique network is heterogeneous when it is built out of elements that have been designed independently. This is the case when computer systems come from different manufacturers. Heterogeneity may result from the association of equipments which were already installed independently. It may also be a policy of diversification in order to get the best products at the best price.

TRANSMISSION ELEMENTS

Telephone lines

They are the most commonly used transmission elements. Modems (interface adaptations) are used to convert digital signals to analog. A variety of speeds are available, from 200 bits/s to 19 200 bits/s.

Other specialized circuits include telegraphic lines (50 or 75 bits/s) and primary groups (48 K bits/s).

Configurations may be point-to-point, or multipoint. Dial-up circuits may be used in the first case, only leased ones in the second case.

Multiplexing

In order to save on transmission costs, several channels may be routed through a single transmission line. Various techniques are used. Recent ones are based on time division, and packetizing of bit streams. Then, configurations take the form of hierarchical levels, with one or several levels of traffic concentration.

Packet switching

It can be thought of specialized message switching with short transit delay, or a variety of time division multiplexing in which time slots are variable. Packet switching is becoming a basic tool for data transmission, as it is well adapted to digital techniques.

Satellites

They are commonly used for the transmission of analog signals on independent channels. New techniques are developing for the transmission of digital signals in broadcast mode. This would complement terrestrial packet networks for long distance or large bandwidth transmission.

An extremely promising domain of satellite transmission is the possibility of using small antennas without terrestrial link to a ground station. This would make television, telephone, and computer services available in any point of the world, at the cost of a small portable ground station.

PACKET TRANSMISSION

As seen as specialized store and forward message switching, the characteristics of packet switching are following :

Transit delay

Typically 100 ms to a few seconds, depending on network parameters. The delay is made up of as many queueing times as there are switching nodes to be traversed. Thus, short packets, fast lines, and few nodes are recommended to get the shortest delay.

Error rate

Since bits are wrapped into packets, transmission errors may be detected and corrected with high reliability. With 2000-bit packets and 16-bit checksums, the probability of bit error is lower than 10^{-10}. This may be considered as error-free for most purposes.

However, there is not such a high reliability in terms of complete packets. There may be losses or duplication in some pathological conditions. Although very unfrequent, it cannot be completely ignored.

Adaptive routing

Enough redundancy may be introduced in the network topology so that there is always a minimum of two distinct routes between any two points. Component failures have no other effect than reducing the network throughput.

Transparency

There is no difficulty in making a packet network transparent to information exchanged through it. However, due to transit delays, it may not be transparent to transmission conventions (also called procedures or protocols). Therefore, in most situations packet networks require some interface adaptation because most existing transmission procedures were designed for plain circuits, on which transit delays are negligible.

Supervision

Packet networks introduce a larger set of transmission components equipped with more intelligence. Locating failures would be extremely time consuming if there were not built in mechanisms intended for fault reporting, statistics, etc. Such facilities could have been inserted much earlier, but the problem was not well identified until the first packet networks were built.

PROTOCOLS

The principle of independent layers has been generally accepted as a sound approach in the construction of large systems. In networks, it was logical to introduce system boundaries between the major distinct functions.

The transmission network is the lower layer. Its function is to carry packets from one point to another. This can be accomplished through a variety of transmission techniques, including leased circuits. Packet switching is an adequate solution.

A transport function is a second layer. Its role is to move blocks of data from one user to another. A user may be an application program running in a computer, or a sub-system such as a transaction processor. A user may also be a terminal controller, or an intelligent terminal.

The transport function takes care of the constraints associated with the existence of a transmission layer, e.g. error correction, fragmentation into packets and reassembly into larger blocks. It also provides for additional facilities such as multiplexing independent streams of data, and flow control between sender and receiver.
The transport function operates end-to-end so as to provide for the highest guarantee of error free service (Fig. 2).

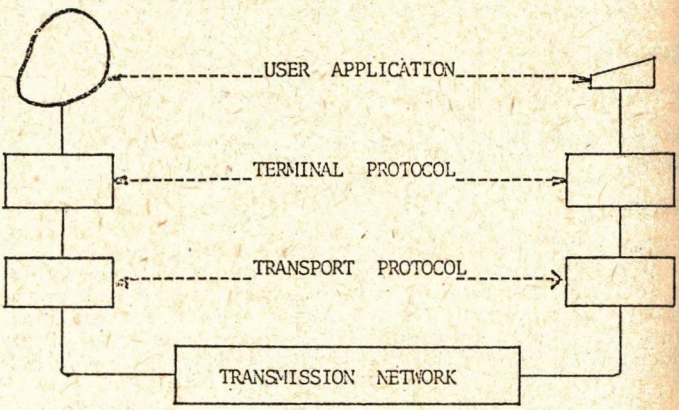


Figure 2 - PROTOCOL LAYERS

A terminal access function is usually a third layer, since the most common form of using computers is from a terminal. The role of this layer is to allow the control of remote terminals through a set of message conventions.

These end-to-end conventions are called protocols in the network lingo. They make no assumption about local implementations, as long as messages are exchanged with proper formats. Owing to this discipline, it becomes possible to associate heterogeneous systems without putting too many constraints on their design.

In typical implementations terminal and transport protocols are wrapped into a network access method which can be located in a front-end minicomputer. On the terminal side, it is located within a remote concentrator, or even micro-programmed within an intelligent terminal.

When data exchange takes place directly between application programs, they may call only on the transport function. This form of association opens the way to distributed processing in which several computer systems are simultaneously involved for the same application.

ADAPTATION OF EXISTING SYSTEMS

Future computer systems will be designed for a network environment. But present systems need adaptations to become network partners.

A first problem is the insertion of packet switching, which is not compatible with most existing transmission procedures. A second problem is the insertion of end-to-end protocols within existing access methods. Both problems would normally be solved with proper adaptations in computer manufacturer software. However, very few customers are willing to take the responsibility of mingling with their manufacturer's products. An interim, but popular, solution is a gateway.

This term of gateway is commonly used to designate an interface and protocol converter. As seen from a computer system, it behaves as a cluster of friendly terminals. As seen from the network, it implements all necessary network protocols (fig. 3). Converting local terminal protocols into network protocols and vice versa may be a tricky game, since they are not necessarily functionally equivalent. But there exist usually some acceptable expedients.

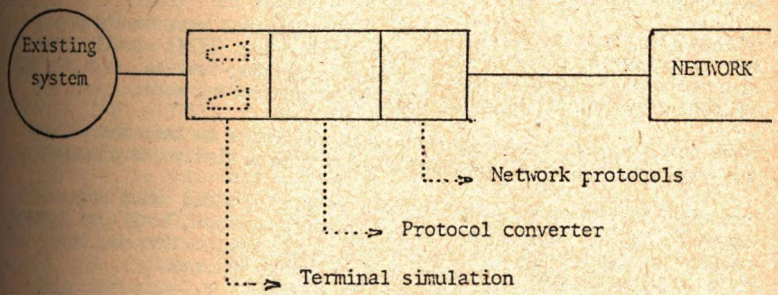


Figure 3 - GATEWAY

Typically, a gateway may be implemented in a mini-computer. It can use an existing front-end communication processor, although this may require some adaptations in manufacturer software

RESOURCE SHARING IN AN INFORMATIQUE NETWORK

Data bases are a typical form of resources, likely to be shared, since they are not easily moved, and their up-dating requires a well defined authority in charge of maintaining them. Accessing simultaneously several data bases from a single application opens the way to distributed data bases physically disseminated, but logically integrated.

NETWORKS IN WESTERN EUROPE

D L A Barber

Director
Executive Body
European Informatics Network

ABSTRACT

This paper gives an overview of the development of data communications networks in Western Europe. It describes, briefly, a number of public, private and research networks and touches upon some of the problems yet to be solved before people with terminals are able to communicate freely with computers and other terminals, using a public data network.

1. INTRODUCTION

There are now a surprisingly large number of data networks planned or in operation in Western Europe. This makes it impossible in the present paper to treat all of these in any detail, and raises the problem of how to classify and relate them to each other. One method would be to deal with them on a national basis, but this would make a division into types and purpose difficult; also, the international networks would pose a special problem with such an approach. Instead, it seems better to consider networks according to purpose, ie public, private and research, and to comment on national aspects where this seems appropriate. Very comprehensive descriptions of many of these networks are available in the literature (ref 1), so only a brief summary of key features will be given here.

Public networks in Europe will almost certainly be provided only by the Postal and Telecommunications Authorities (PTTs) and national versions are likely to differ in their detailed design because they will depend on the existing infrastructure of telecommunications resources. However, the kinds of service they will provide and their interface with the subscribers are the subject of CCITT recommendations and it is to be hoped that these will be interpreted in the same way for all networks, so that a subscriber's equipment can be used equally well in any country.

Apart from the public networks which have been announced by a number of European countries, an important development has been the Euronet Project. This is an information dissemination network with a communications subnetwork provided by a consortium of PTTs. A noteworthy aspect of this network is its possible use for carrying public traffic in addition to that of the Euronet Centres.

The public networks have arrived relatively late on the scene for there have been private and experimental networks for many years. The success of such networks has shown the need for public services, and this is perhaps particularly true of the private networks which are operated on a commercial basis. Often these are very large compared with the planned public networks and it will be interesting to see how these private networks are affected when the new public services become available.

Ideally, one might suppose that the private networks should be discontinued, and their traffic thereby transferred to the public network. This would provide a revenue for the new networks that would justify a large capital investment and an implementation on a large scale, with all the benefits that would bring to users. However, to achieve this, the public networks would have to offer equivalent services and the amortisation of existing private networks would have to be taken into account.

The research and experimental networks have played a vital role in illustrating techniques and the feasibility of using new methods for data communications. Indeed many basic features of public and private commercial networks were first demonstrated in experimental networks. But the growth of public services would suggest that there is a diminishing requirement for research networks, and this is certainly true with regard to the simple data transportation facilities they provide.

However, the attention of research workers is now moving towards solving problems arising when users systems interact through the medium of a network; an area which will become particularly important when public networks are available and can, potentially, connect any user with any other user. With this in prospect, the sad lack of standards at the user level becomes apparent, making it

difficult, if not impossible, for two users to communicate effectively, unless they belong to the same organisation and use the same procedures.

This paper considers these various aspects of European data networks before touching upon some other features and problems of networking which seem not yet to have been considered in any depth in Europe.

2. PUBLIC NETWORKS

The findings of the Eurodata Study (ref 2) has prompted many European PTTs to announce their intention of providing public data networks, although in many cases detailed plans have not been given. Three networks, however, have been described in considerable depth and, since they are based on rather different principles, it is useful to consider each of them more closely, for it is very likely that other public networks will be designed along lines similar to one or other of these examples. They are the French Transpac network, the Nordic Network of the Scandinavian countries and the integrated telex and data network of the Federal Republic of Germany.

However, it is first worthwhile to outline the various CCITT recommendations that are relevant to the new public networks, and that will, hopefully, ensure that all of these networks appear to be similar as far as users are concerned.

2.1 CCITT Recommendations

The CCITT (International Telegraph and Telephone Consultative Committee) is an organ of the International Telecommunications Union. It is responsible, inter alia, for preparing recommendations concerning data communications for the various telecommunications

administrations. Nearly thirty X series recommendations (ref 3) appropriate for public data networks have been prepared by Study Group VII and others are under discussion. Typical of these are the following:-

General Services and Facilities

X1 - User classes of service, covering categories such as start-stop/ synchronous and gives various speeds to be made available

X2 - User facilities are given in the form of essential and optional features.

Physical Interfaces Between Data Terminal Equipment (DTE) and Data Communications Equipment (DCE)

X20/X20bis - Covers connection to the telex/ telephone networks eg V series modem interfaces

X21/X21bis - Covers connections to new digital circuit switched data networks

Functional Specifications Covering the Behaviour of Components and Facilties

X3 (XA) - Specifies the properties required in a PAD (Packet Assembly Disassembly) facility

Procedural Interfaces Controlling Data Transfers Between DTE and DCE

X25 - Describes a packet mode interface offering permanent virtual circuits and switched virtual calls

X28 (XB) - Deals with start/stop DTE accessing a PAD

X29 (XC) - Gives procedures for exchanging control information and users' data between a packet DTE (Host) and a PAD

Among the many other recommendations the following are also worth mentioning here:-

X50 -	A multiplexing scheme between synchronous networks
X7x -	An internetwork interface protocol
X95 -	A hypothetical model for a data circuit

Key issues today are the question of whether public networks should be circuit-switched or packet-switched, and whether packet-switched networks should provide datagram or virtual-circuit based services. These questions remain only becaue they cannot be resolved by rational considerations, and so operational experience is urgently required to remove the many prejudices that tend to cloud discussion at the present time.

Ideally, the type of network they are using should not matter to subscribers, who should receive similar facilities wherever they are located. Fortunately, one of the main purposes of CCITT is to seek compatibility at the international level, and this inevitably has an impact on the design of national networks, making them appear more similar to users than if they were developed independently of eachother.

2.2 France

Following an earlier assessment of packet switching techniques using the RCP (Reseau Comutation de Paquet) (ref 4) the French Post Office decided in 1976 to introduce a switched/permanent Virtual Circuit public service in the form of the Transpac network (ref 5), due to come into service in June 1978 with 12 exchanges. By 1980 there will be 25 exchanges, and the number could rise to 100 by 1985, depending on the demand from subscribers.

The exchanges are designed around the CII Mitra 125 computer for call management, and a purpose designed processor for basic packet switching in the data transfer phase of a call.

Access to the exchanges will be by direct connection, by the telex network, by leased circuits or by the switched telephone network, and the speeds available will range from 50 bits/s to 48 k bits/s, using V24 or V35 physical interfaces as appropriate. Recommendation X25 and X29 will be used with packet terminals and X28 for character terminals which will be connected to packet assembler disassemblers (PADs) X29. The services provided will include closed user groups, and reverse charging.

The tariff will be based on a monthly rental together with a duration and a volume charge, all of which are independent of distance. Typically, the connection charge will be 800 FF for a two-wire circuit, the annual charge will vary from about 3000 to 15000 FF depending on speed, while the volume charge will be 0.05 FF per 1024 octets.

2.3 Scandinavia

The PTTs of Denmark, Finland, Norway and Sweden are cooperating in the development of the Nordic Public Data Network (ref 6). Following studies of the most appropriate techniques for a Scandinavian network a circuit switched system was chosen and will come into service during 1979, with four exchanges (Copenhagen, Helsinki, Oslo and Stockholm): there are expected to be 12 to 14 exchanges by 1985. The possibility of introducing packet switching services is also actively under consideration.

The basic building blocks of the network are the Data Switching exchanges (joined by 64 k bit/s links derived

from the carrier network), and the data circuit concentrators and multiplexers to which are connected data circuit terminating equipment located in subscribers' premises. Access by subscribers is either directly using baseband transmission, or through modems and leased circuits. Speeds available will range from 600 k bit/s to 9.6 k bit/s, using an "8 plus 2" bit envelope structure in a synchronous network, which is compatible with the PCM 64 k bit/s telephony standards.

From the beginning, a wide range of services will be offered, including "closed user group","direct call", "abreviated address", etc. By 1980 further facilities such as "connect when free" and "charge transfer" will be added to give virtually all the user features that have been considered within CCITT recommendations X1 and X2.

The tariff structure is still being worked out, but it will be possible for subscribers to have a detailed record of their calls upon request, as well as a count of items such as call attempts and connections less than five seconds, calls over five seconds, and the types of facilities used.

2.4 Germany

The electronic data switching (EDS) system (ref 7) was developed by Siemens for the Federal Republic of Germany's Post Office, primarily for handling telex and datex traffic at speeds up to 200 bit/s, although it was designed to take some higher speed traffic also. The original system has been extensively developed to handle a greater proportion of high speed data traffic and is now called the Integrated Data Network (IDN). In addition to datex 2400 (2.4 kbit/s) which has been available since July 1976, speeds of 4.8 kbit/s are planned for late 1977 and 9.6 kbit/s for 1980. IDN is essentially a circuit switched system, but packet

switching facilities will be added in 1977/78 in the form of a pilot project. Presently seven exchanges are in operation and there will be 21 in service by 1980.

The facilities and services available are those detailed in CCITT recommendations X1 and X2 and the interfaces conform with recommendations X20 and X21 and (in due course) X25.

The tariff for the datex network comprises a monthly fixed charge depending on speed, a duration charge and a distance charge based on the areas of the public telephone network. The duration and distance components of the tariff are lower during the night. As an example, for datex 2400 the fixed charge is 290 DM per month, the local call charge is 200 seconds per DM during the day and 900 seconds during the night. These times are reduced to 54 and 200 respectively for areas which are more than 100 km apart.

2.5 Other PTT Networks

Two other PTT networks are worth mentioning in this paper. One is a packet switching service operated by the Spanish PTT (ref 8), the other is the experimental packet switching service of the UK Post Office (EPSS,ref 9).

The Spanish network RETD (Red Especial de Transmission de Datos) was the first public network to enter into operation in Europe, and is somewhat intermediate between a packet switched and a message switched system. It currently has two nodes and about 700 terminals but this is expected to rise to 7000 terminals by 1978, when there will be 9 remote concentrators and three switching centres. At present, the network does not offer the X25 interface and the services provided do not line up with the CCITT recommendations.

RETD has been operating with a regular tariff for some time now, although it is difficult to compare this with those of

other networks since the services are not the same. The tariff is based on the quantity of data transmitted and includes components for distance and time of call. There is also a connection charge.

In the United Kingdom, The Post Office experimental network became operational in early 1976, although it was not officially opened until early 1977. It has three nodes based on highly reliable packet switches and can handle nearly 2000 character terminals and over 80 packet terminals. Access is by leased lines and the switched network, while both datagrams and virtual circuits are provided. EPSS is noteworthy in being the earliest true packet network put into operation by a European PTT but because it was implemented before the X25 recommendation was published it does not offer this interface at present. It can however provide most of the facilities recommended by CCITT.

The key purpose of EPSS was to allow potential users of packet networks to experiment with the adaptation of their systems and a free service has been provided for this purpose. However, the tariffs have now been anounced. These are structured similarly to those of other networks in that there is a connection charge, a monthly access charge and a volume charge.

2.6 Euronet

Euronet (ref 10) is particularly interesting because it is an international network that will be operated by the PTTs on behalf of the Commission of the European Communities and in that respect it is quite different from other networks. Before the advent of Euronet one might have supposed that international services would be provided by the interconnection of national public data networks, and indeed this is still one way in which future public

services might be organised. But an attractive alternative could be to provide an international high level network linking national public networks, and connecting to each of them by a similar interface. This would avoid the need for many bilateral agreements between individual countries covering the connection of different national networks.

Euronet has the technical capability to serve as an international high-level network although at present it provides services directly to subscribers. In this way it apparently by-passes the services that may be provided in future by bilateral connection of public networks and it will be interesting to see whether the growth of public services which are likely to be required to meet the demand in the 1980's will be met by expanding and adapting Euronet or by connecting national networks to each other.

The technical basis for Euronet is the Transpac design already described above and the appropriate CCITT recommendations are, of course, to be employed. In addition to the basic service, however, it has been foreseen (ref 11) that some remote terminal handling and concentration may be necessary in addition to the PAD facility, and the design of Euronet may be extended to include the use of character interface processors (CHIP) and packet interface processors (PIP) connected externally to the network. The former handle clusters of character terminals while the latter are concerned with clusters of packet terminals.

2.7 Network Tariffs

The cost of using a data network is obviously one of the key considerations for subscribers planning new systems, or considering the transfer of existing systems to use the new services. Some brief mention of tariff structure and, in

some cases, actual charges, has already been made above, when describing the various public networks. This is sufficient to indicate that the cost of using new services is by no means easy to determine and, when this is further complicated by the costs involved in adapting equipment to use the new services, it is not easy to estimate how rapidly subscribers will wish to move over to the new networks. The principles adopted by the various PTTs in formulating their tariff structues and charges obviously differ, and indeed are bound to do so due to the differing constraints they must meet. This can lead to anomolies with international services that are sometimes puzzling and may perhaps be exploited by some subscribers. Some interesting comparisons of published tariffs are contained in ref 12.

3. PRIVATE NETWORKS

There are a wide variety of private networks now in operation throughout Europe; these are mainly national networks, although there are a few international networks owned by multinational companies. The technical principles used by most of the networks are similar (ref 13), with multidrop lines joining remote concentrators and terminals to a central computer complex. However, two networks operated by consortia, the SITA network and the SWIFT network are worthy of a brief description because they are shared networks serving groups of users with differing needs.

3.1 SITA

One of the oldest commercial networks is the SITA network (ref 14) which first went into service in 1970. SITA (Societe Internationale de Telecommunication Aeronautique) was established in 1949 by a group of airlines in order to provide them with an economical means of exchanging messages.

Originally the SITA network was a low-speed message-switched network, but this had evolved into a packet-switched network by 1970 and has since grown until it now handles over 10 000 teleprinter terminals and nearly 1 500 visual displays. The SITA network is therefore much larger than the planned public data networks at the present time, and it will be interesting to see, in the longer term, whether these public networks can provide facilities for the SITA network which would enable it to continue to offer services to airlines, without having to modify its operations to any signficant extent.

3.2 SWIFT

The SWIFT network (Society for World Wide Interbank Financial Telecommunications) (ref 15) was founded in May 1973 by 239 banks in 15 different countries following a study in the late 1960's initiated by 68 banks from some half a dozen countries. Membership of SWIFT has now reached over 480 banks and this certainly reflects the vital need for the SWIFT network.

As its name implies SWIFT is world wide but it is worthwhile including it as a European network becaue its headquarters are located in Belgium, and, in its first phase, it covers most of Western Europe and North America. The size of the network, as is the case with the SITA network, is potentially larger than the presently planned public networks, and similar considerations apply to its long term relationship with these networks. The user-level standards being developed by the participants have not been widely publicised, but the emphasis placed on security and privacy could have significance for network users in other fields.

4. RESEARCH NETWORKS

The research networks in Europe are important because they have played a significant role in the development of the public networks and because they allow on-going study of the development of techniques for using these public networks.

The European networks that will be discussed briefly below are: the NPL network, the Cyclades network, the GMD network and the EIN network.

4.1 NPL

In the United Kingdom, the research network designed and built at the National Physical Laboratory (ref 16) was the first packet switching network to come into service in Europe, and began experimental operation in early 1970, and has long since become a reliable data transportation mechanism for the whole Laboratory. Unfortunately no trunk network was developed to match the NPL local network, and this has inhibited the use outside NPL of the advanced techniques for using packet switched networks that continue to be developed by the Laboratory.

Architecturally the NPL network contains all the elements of a well-structured data network, but uses only a single switch to provide a datagram service for exchanging packets between various computers (known as user machines). One of these is resonsible for terminal handling, others provide the various computing facilities to the Laboratory, while some act as gateways to other networks.

Over 200 terminals, mostly Visual Displays, are connected to the network through Peripheral Control Units and any terminal may access any of the user machines. This is done by establishing a virtual circuit between the peripheral

control unit and the terminal handler which assembles and disassembles packets on behalf of terminals. These packets are exchanged with the other computers in the network through the medium of the packet switch which, untypically, is located in the same computer as the terminal processor.

The connection of the NPL network to the EIN network has been an important recent development which has greatly widened the opportunities for further research on the use of data networks, by allowing fruitful cooperation with other European research centres.

4.2 IRIA

In France, the CYCLADES network (ref 17) with its packet switching subnetwork Cigale, was designed by IRIA (Institut de Recherche d'Informatique et d'Automatique) as a research network, and came into operation in 1973 eventually growing to seven nodes at five sites. Following the decision to develop the French public network Transpac, the CYCLADES network has reduced in size, but is expected to continue in operation as a research network and a tool for experiments by universities and other experimental establishments. It is presently a three node network, with two nodes in Paris and the third in Grenoble.

The CYCLADES network has had a considerable impact on the thinking about the architecture of data networks, because a purist approach was adopted towards the development of a network based on well-defined levels of protocol. These are implemented in the Host computers which are served by a pure datagram subnetwork, Cigale.

The formal approach to protocol definition that was adopted for the CYCLADES network has had a significant influence on other networks. In particular, the European

Informatics Network has adopted the work of CYCLADES as a basis for the development of the end to end protocol, or Transport Station. The work of IRIA in connection with other levels of protocol has also been a major influence in the thinking about the EIN network.

4.3 GMD

In the Federal Republic of Germany, the Gesellschaft Fur Mathematik Und Datenverarbeitung MBH (GMD) has developed a research network, GMDNET, (ref 18) with four nodes at three sites. The project did not begin until 1974, so active research network studies are relatively recent in Germany. However, developments are now very rapid and there is a very effective cooperation between GMD, the Ministry of Research and Technology, the PTT and various user groups in a number of pilot projects (ref 19). Of these, PIX seems most significant in the research area. PIX is an association of several groups interested in networks, cooperating in the specification of various user level protocols based on the X25 interface. These include an end-to-end transport protocol, a virtual terminal protocol, and various applications oriented protocols.

4.4 EIN

The European Informatics Network Project (ref 20) was conceived as a joint European research venture as long ago as 1969, although it took until November 1971 to sign an international agreement to establish to Project, and until 1973 before the treaty was ratified. Since then, the Project has passed through the stages of network specification by EIN experts followed by its design and construction by contractors, resulting in the completion of a five node subnetwork in May 1976. This presently provides a datagram service between research centres located in the UK, France, Italy and Switzerland. These

are NPL and IRIA (mentioned above) together with CETIS (Centre Europeenne de Traitement de l'Information Scientifique, Euratom, Ispra), CREI (Centro Rete Europea Informatica) in Milan and ETH (Eidgenoessischen Technischen Hochschule) in Zurich.

EIN is the first government sponsored international research network and has undoubtedly played an important role in bringing together people working with other experimental networks. With the announcement of the various public networks, the EIN communications subnetwork has become less important, but the efforts of the participants in the Project have now moved on to solving the problems of using the network and of reconciling the differences between the various systems that it interconnects (ref 21). These systems range from large computers at CREI, ETH and CETIS, to the local network at NPL, and the CYCLADES network connected through IRIA.

A recent decision to adapt EIN to operate with X25 interfaces makes it a much more attractive test bed for experiments in user-level protocols, and its use in this role could well prove one of the more significant aspects of the Project.

5. USER ASPECTS

All networks are similar in so far as they provide data transportation facilities to interconnect the component parts of users' data processing systems. It therefore seems appropriate to consider them according to their impact on users, particularly future users who will be profoundly affected by the new public data networks offering features not available at the present time.

The agreement of the CCITT X25 recommendation is very significant for users, because the public networks will appear largely identical through this virtual circuit interface. However, the lack of standards for use within the users' systems will continue to inhibit their ability to interact effectively.

The research networks are actively involved in the development of suitable protocols (ref 22) for linking users' systems, but there remains much to be done. Furthermore, the aspects which most affect the users themselves, such as a standardised user interface and control language to manipulate network services have received very little consideration so far.

6. OUTSTANDING PROBLEMS

Looking ahead, it seems likely that the advent of a relatively cheap data terminal based on the teletext system (ref 23) may well soon bring terminals into ordinary people's homes. This will create a demand for communications facilities which cannot be provided by present plans, as far as one can see.

The first requirement is the provision of efficient and economical means of connecting large numbers of terminals to the proposed public data networks. These have been largely intended to serve a low density of terminal population on a nationwide or international scale. But when many home terminals have to be handled, connections using people's local telephone circuits would seem essential, and this must be achieved without disrupting the normal telephone service. Possibly a subscriber's loop carrier scheme is required (ref 24), but this could be difficult to introduce in some networks.

A second issue is the agreement of a uniform method for users to manipulate the network, so that wherever a user may wish to work, he still uses the same commands and procedures to carry out a particular task. The telephone networks in different countries differ in minor but irritating ways, and there will be great scope for even bigger differences between data networks, unless the problem of standardisation is addressed now as a matter of urgency. Unfortunately there seems little sign that this is happening yet.

There are many other problems that will arise due to the growth of public data networks in the next few years, but the economic and social benefits that they undoubtedly will bring to ordinary people, as well as to large public and private organisations, will, hopefully, justify the considerable investment that has been made in the study and development of data networks in the past ten years.

7. REFERENCES

1a. JOHNSON T
Packet Switching Services and the Data Communications User
Ovum Ltd London (1976).

1b. ALLERY G D
Data Communications and Public Networks
Proc IFIP 1974 Part 1, North Holland, Amsterdam,(1974)

1c. KIRSTEIN P T
Planned New Public Data Networks
Computer Networks 1 (1976) 79-94

1e. KELLY P T F
An Overview of Recent Developments in Common User Data Communications Networks
Proc 3rd Int Conf on Computer Communications, Toronto (1976) 5-10

2. EURODATA STUDY,
PA Management Consultants (1972)

3. CCITT VIth Plenary Assembly Document 55, Part III
Proposals for New and Revised X Series Recommendations for Data Transmission Over Public Networks

4. DESPRES R
RCP: The Experimental Packet-Switched Data Transmission Service of the French PTT
Proc 2nd Int Conf on Computer Communications, Stockholm, (1974) 171-175

5. DESPRES R et al
The French Public Packet Switching Service: The Transpac Network
Proc 3rd Int Conf on Computer Communications, Toronto (1976) 251-260

6. LARSSON T
A Public Data Network in the Nordic Countries
Proc 3rd Int Conf on Computer Communications, Toronto
August (1976) 246-250

7. ALTEHAGE G, STAUDINGER W
Economic Aspects of a Public Switched Data Network
Exemplified By the Deutsche Bundespost Electronic Data
Switch System (EDS)
Proc 3rd Int Conf on Computer Communications, Toronto
(1976) 159-164

8. ALARCIA G et al
CTNE's Packet Switching Network, Its Applications
Proc 2nd Int Conf on Computer Communications, Stockholm,
(1974) 163-169

9. BROOMFIELD C F
Packet Switching - The Experimental Packet Switched
Service
Computer Communications Review 3 (1975) 2-11

10. EURONET NEWS,
Directorate General Scientific and Technical Information,
CEC, Luxembourg.

11. FACCHIN M
Presentation d'Euronet
SEAS Winter Meeting 1977, Luxembourg.

12. JOHNSON T
Tariff Levels and Structures for the New Public
Switched Data Networks
Proc.Online Conference on Data Communications Networks
(to be published).

13. DAVIES D W, BARBER D L A
Communication Networks for Computers (Chapter 4)
John Wiley (1973)

14. HIRSCH P
SITA: Rating a Packet-Switched Network
Datamation, March (1974) 60-64

15a HALL W
SWIFT - The Revolution Round the Corner
The Banker (1973) 633-639

15b REUTERSKIOLD C
The SWIFT Network
Proc.Online Conference Data Communications Networks
(to be published)

16. SCANTLEBURY R A, WILKINSON P T
The National Physical Laboratory Data Communications Network
Proc 2nd Int Conf on Computer Communications,
Stockholm August (1974)

17a POUZIN L
Presentation and Major Design Aspects of the CYCLADES Computer Network, Data Networks Analysis and Design
Proc 3rd Data Comm Symp (1973) 80-85

17b POUZIN L
The CYCLADES network - present state and Development Trends - Symp Comp Netw trends and applications
IEEE (1975) 8-12

18. RAUBOLD E
The GMD-Net: Goals and Structure.
INWG General Note No.105.

19. SARBINOWSKI H
Network Activities in the Federal Republic of Germany: Pilots to a Public Packet Switching Service.
Telecommunications (to be published)

20. BARBER DLA
A European Informatics Network: Achievement and Prospects
Proc 3rd Int Conf on Computer Communications, Toronto August (1976) 44-50

21a European Informatics Network: End-to-End Protocol
EIN/76/003

21b SCHICKER P, ZIMMERMANN H
Proposal for a Scroll Mode Virtual Terminal
EIN/CCG/77/02 INWG Protocol 52

22a CERF V, MCKENZIE A, SCANTLEBURY R, ZIMMERMANN H
Proposal for an Internetwork End To End Protocol
CCITT Study Group VII September (1975)

22b CERF V G et al
A Protocol For Packet Network Intercommunication
IEEE Trans Comm 5 (1974) 637-641

22c EUROPEAN INFORMATICS NETWORK
End To End Protocol
EIN/76/003

22d SCHICKER P, ZIMMERMANN H
Proposal for a Scroll Mode Virtual Terminal
EIN/CCG/77/02 INWG Protocol 62

23 FEDIDA S
Viewdata: an Interactive Information Service for the General Public
Proc Eurocomp 75, Online (1975) 261-266

24 JELSKI R
Subscriber Subcarrier System - a New Life.
Communications International, May (1977), 29-30.

AN OVERVIEW OF THE EUROPEAN INFORMATICS NETWORK

D L A BARBER

Director,
Executive Body
European Informatics Network

ABSTRACT

The achievements of the European Informatics Network and the oustanding issues now under consideration are reviewed briefly in this paper, with the intention of giving an overview of the present (June 1977) state of the project.

A brief history of the project and its objectives is followed by a description of the communications sub-network and the problems it has raised. This leads on to a discussion of the work of the national Centres connected to the sub-network, and the paper concludes by considering some future possibilities.

1. HISTORY

EIN was conceived when a number of co-operative projects were proposed in 1968 by the PREST Committee (Politique de la Researche Scientifique et Technique) of the European Economic Community. In 1969 the COST group (Co-Operation Europeenne dans le domaine de la recherche Scientifique et Technique) took up these proposals, forming study groups to examine them in detail. As a result, project number 11, to establish a European Informatics Network, became the subject of an international agreement (ref 1), which was signed on November 23rd 1971, between:

France, Italy, Norway, Portugal, Sweden, Switzerland, the United Kingdom, Yugoslavia and Euratom.

The Kingdom of the Netherlands joined in mid-1974 and the Federal Republic of Germany in early 1976, while more recently both Spain and Belgium have also expressed interest in taking part.

Two levels of participation in the project are possible: Signatories may receive information only, or may also nominate and operate a Centre connected to the network.

Information only: Norway, Portugal, Sweden, Yugoslavia, Netherlands, Germany
Approximate Total Cost: 13M Belgian Francs each Signatory

With Centres: France, Italy, Switzerland, United Kingdom, Euratom
Approximate Total Cost: 100 MBF each Centre.

2. PURPOSE OF THE PROJECT

The Agreement states:

(Art 1) that the network will facilitate research at the Nodal Centres into data processing problems and will permit the sharing of resources, and

(Annex) that it will:

1) allow the exchange of ideas and the co-ordination of research programs

2) allow the comparison of ideas for national networks and promote the agreement of standards

3) be a model for future networks, whether for commercial or other purposes, and will reduce the differences between future systems.

The Annex goes on to say that the hardware and software developed should be suitable as a basis for any permanent international network which might be built in future.

3. PROJECT STRUCTURE

The organisation for carrying out the project is as follows:

- Management Committee, set up by the Agreement, comprising representatives of all Signatories, and having ultimate responsibility for the project

- Executive Body, consisting of a Director who reports to the Management Committee and three technical assistants;

- Technical Advisory Group, also comprising representatives of all signatories, to advise the Director;

- Centres Coordination Group, responsible for guiding work that requires coordination between the Centres;

- ad hoc Working Groups and Special Interest Groups, for example, concerned with drafting the specification and assessing tenders.

4. CONTRACTORS

In order to ensure that maintenance and extension of the system can be done with rapidity and assurance, it was decided to entrust the development and installation of the communications sub-network to the contractor. The Signatories proposed about 50 firms in total from which to make the selection. Nineteen firms expressed keen interest, and five Consortia were formed and submitted tenders. Eventually:

SESA (France) and Logica (UK) as main contractors with SELENIA (Italy) and FIDES (Switzerland) as subcontractors

were awarded a fixed-price contract - signed 1974 October 17.

This "Initial Contract" was to design and demonstrate a "Network Switching Centre" (NSC) suitable for use in EIN (Contract price - 30 MBF). The technical specification was prepared by the TAG, with observers from CEPT; it includes requirements on facilities, performance and acceptance testing. The processer selected is the Mitra 15, from CII. The demonstrations called for under the Initial Contract tasks were completed in 1975 October, according to schedule.

The Initial Contract gives proprietory rights in the resulting software to the Contractors, but free use is allowed to a Signatory of the Agreement, to the PTT of that Signatory, or to a non-profit making body for its own internal purposes, provided such use is only within the territory of that Signatory.

The Contractors have given an assurance that they would make the software available on a fair commercial basis for purposes not covered by the above provisions. Use for other purposes would be relatively easy because of the adaptability of the design and the exceptionally high standard of documentation.

Supply contracts, separately negotiated between the five Centres and the Contractor, covered the installation of NSCs to form the initial network. As planned, this was handed over to Centres on 1976 May 26 (approximate cost for 5 Centres = 35MBF).

5. SUB-NETWORK DESIGN

The sub-network design (ref 2) is based on techniques proved by earlier networks. Key points are:

- a Datagram type of service is provided, together with optional end-to-end sequencing and traffic control facilities;

- the HDLC frame is used for transmission;

- adaptive routing is used within the sub-network;

- connection of a subscriber to more than one node is permitted to enhance reliability;

- well proven hardware is used, as over 1000 Mitra 15s are now in operation;

- all relevant international standards have been met;

- extensive facilities are available at the nodes for statistics collection, recording of alarms, modification of network parameters, and the modification and remote loading of node software through the sub-network;

- nodes are capable of entirely unattended operation.

6. NETWORK CONFIGURATION

The Computer Network comprises the Nodal Centres with their data processing systems connected by the communications sub-network developed under the Initial Contract. The Centres and their installations are:-

London
- National Physical Laboratory
- NPL network & services - KDF 9

Paris
- Institut de Recherche d'Informatique et d'Automatique
- Cyclades network & services - IRIS 80

Zurich
- Rechenzentrum der Eidgenoessischen Technischen Hochschule
- CDC 6400

Milan
- Centro Rete Europea Informatica
- Univac 1108

Ispra (N.Italy)
- Centre Europeen de Traitement de l'Information Scientifique (EURATOM)
- IBM 370/165

The communications sub-network uses one NSC at each Centre, connected thus:

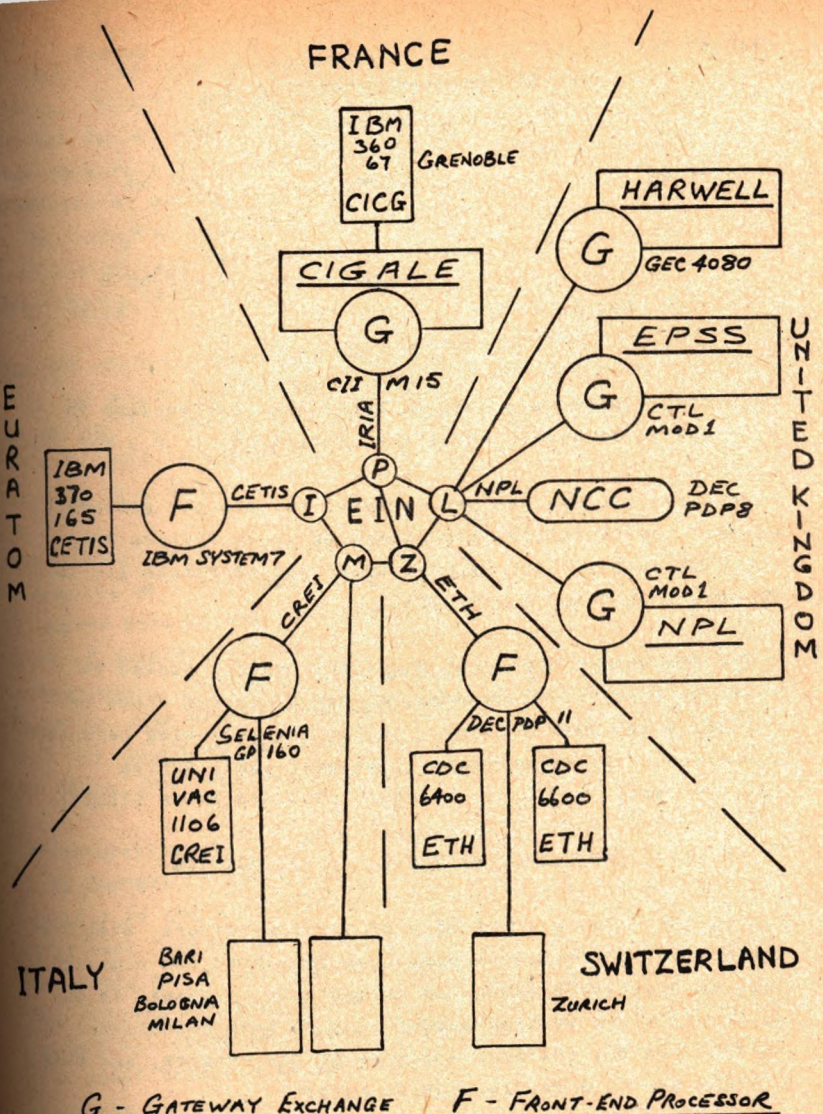
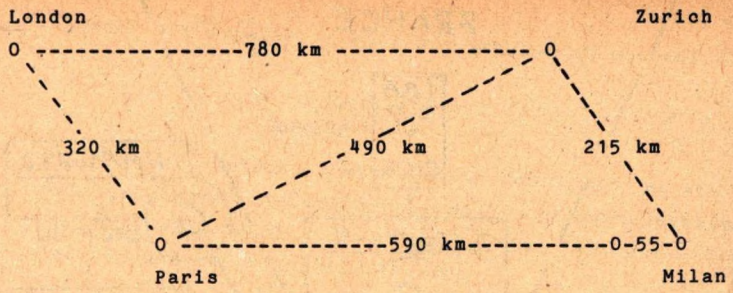


figure 1 EIN Centres Systems



Each link comprises a channel bandwidth line working in full duplex at 9600 bit/s (total rental cost: 7.4 MBF/yr).

Other Research Centres are expected to join the network, which is designed for easy expansion up to fifteen NSCs. Possible future Nodal Centres under consideration are in Sweden, Norway and Yugoslavia.

There are also a number of Secondary Centres linked to their National Nodal Centres, either by leased lines or through a national network. So far the following have been connected: Grenoble (France), Harwell (UK), Zurich (Switzerland), and Bari, Bologna, Pisa and Milan (Italy).

7. EXPERIENCE WITH THE EIN SUB NETWORK

During the first year of life of the sub-network, standard fault finding techniques have been developed and compiled into an Operator's Manual to augment that provided by the contractors. A Working Group has also been established to control the day-to-day operations of the network, and this has now become a fairly routine matter.

A noteworthy problem has been the location of faults in lines between Centres. Unfortunately, the original specification did not provide adequately for this task, and

no facilities were requested for looping lines at inputs to modems. Without this feature, the identification of whether faults are in switches or lines is difficult and time consuming. A solution now being considered is the provision of commercially available units which allow lines to be looped by command over a dial-up telephone line. This will permit line checking independently of the sub-network itself.

The NSC hardware installed at the five Centres has suffered only two major faults, and a dozen or so minor ones - such as switch failures - since the network was commissioned. This represents a high degree of reliability over the network as a whole.

The software has proved to be robust, and the basic functions of packet switching are performed well. Minor problems have arisen, but these are not untypical of the kind of complexity that is characteristic of the sub-network. Following discussions with EIN technical experts, some changes are being made by the Contractors, and the resulting detailed improvement in the design will be one of the benefits obtained from the Project.

Unfortunately, it takes an exceptionally long time to deal with changes to the design of the sub-network. This is due largely to the fact that each of the NSCs is owned by a different Centre, and there is therefore no single authority which has complete charge of network operations. The success achieved so far in spite of this decentralisation is a great tribute to the high calibre of the staff involved at each of the Centres. But it suggests that the successful operation of any international network will demand the attention of dedicated people of the highest quality.

7.1 Sub-Network Control and Measurement

The sub-network design allows many features of its operation to be controlled from the operator's console, using an external command language which is processed by a Command Language Interpreter (CLI) to produce commands to invoke statistics gathering and control processes in the various NSCs. Commands can be to control the network, such as to shut or reopen a line or node, or to request information such as the error rate on a line. Information arising inside the network can be directed to the NCC (the location of which is moveable) and to the teleprinter log of the NSC in which the information originated. Such information can be generated in three basic ways:-

- Automatically on a periodic basis
- When an "event" count passes a preset threshold
- On request from the NCC (or local operator)

However, a much more complete control of the network can be obtained by using a Network Control Centre. Apart from the benefit of having an NCC, the problems encountered in its development have led to a much increased understanding of the NSC design. One implementation of the NCC at the National Physical Laboratory (ref 3) is based on the use of a small computer (PDP8E), with a floppy disc backing store, connected as a subscriber computer. This NCC collects certain statistics using the network command primitives and also provides more sophisticated control facilities to the user through the medium of a more user-friendly control language. The activities of the NCC are classified into four headings:

- Routine status monitoring
- Error detection reports
- Diagnosis
- Recovery

Routine status reports may be obtained in three ways. Firstly, reports of traffic and performance statistics and snapshots of selected areas of store can be sent periodically to the NCC. Secondly, the NCC can request any of the above reports on demand as well as partial dump of any core store area. Thirdly, values of important parameters may be requested periodically in order to indicate a more continuous state of these parameters than is achieveable in the other ways.

Error detection reports are generated by the NSCs and sent immediately both to the NCC and the local NSC teleprinter log. In addition, if some selected measurements exceed a pre-determined threshold, occurrence reports are generated. These occurrences are not transmitted immediately but are scanned periodically and reports generated as appropriate. From the routine status and error detection reports, the NCC maintains a map of the current sub-network topology with a list of current NSC and link states and their failure rates and down times, together with a global log of network occurrences.

The NCC can, in some cases, perform automatic diagnosis by comparing reports from several nodes. It may, for example, be able to determine that a link failure reported by one NSC is, in fact, a link failure, instead of a failure of an adjacent NSC. After any such refinement in the location of a fault, or if none is possible, the NCC operator is informed by a system of messages to the NCC operator's console and - in the case of probable serious faults - by a radio paging system. The NCC operator has several diagnostic tools at his disposal. He may, if there is a route open to an NSC, examine the memory of an NSC and modify it if this is appropriate. He may remove lines and NSCs from service selectively, and request a dump of a failed NSC for later analysis and he may request test patterns to be generated to obtain information about the path of traffic through the network.

Recovery following a node failure may sometimes be achieved by reloading it through the sub-network. This may be done automatically by the NCC or on request by the NCC operator. More often it will be necessary to start maintenance activities. The reload mechanism is also used to modify the system software in the NSCs to correct errors, to make configuration changes and to release a new version of the software.

Another problem area related to network control is sub-network measurement. There is a great deal of information that may be gathered from the network about its day-to-day operation. To facilitate this a number of statistics and parameter values can be recorded by commanding the network appropriately. Some Centres are interested in measurements of the sub-network, but no Centre has yet taken on the task of acting as a network measurement centre. Until this is done, systematic measurements of the operation of the sub-network cannot be undertaken in a comprehensive manner, so it is difficult to know how effective are the measurement facilities that have been provided in the design of the NSCs.

8. USER ASPECTS OF EIN

Within the EIN community the majority of users are interested mainly in problems related to the communications sub-network, although some people are now beginning to consider how to use the facilities comprising the sub-network together with the computer systems at the Centres. So far, most of the work within the Project has centred on the adaptation of the Centres' systems to communicate with and through the sub-network. Because these systems are so different, a considerable degree of adaptation is necessary before effective communication can take place.

The inherent incompatibility between different computer systems has always been a serious problem, but the advent of data networks has made it much more worthwhile to find a solution. The current approach is to define a set of layered protocols which offer standard methods of interfacing at a variety of levels; it then remains for each of the systems to be adapted to communicate using these protocols in addition to, or possibly instead of, their existing methods.

The protocols being developed for EIN are shown in Fig 2. They fall broadly into two categories: those which augment or adapt the basic datagram facilities to have a more suitable interface for certain users, and those which make users' systems mutually compatible.

8.1 Network Related Protocols

Protocols to adapt the users' systems to use or control the sub-network are as follows:-

THE LINE PROTOCOL (LP) deals with problems of error detection and retransmission and the identification of line and switch failures. It is responsible for ensuring packets arrive error free at the level of the Transport Station, and, in effect, creates the datagram service interface within the users' own system. This protocol was defined by the EIN contractors and is fully described in reference 4.

THE TRANSPORT STATION (TS) essentially provides a multiplexer and demultiplexer attached to a link into the sub-network. It operates on the packet stream flowing to and from the Line Protocol and assigns and examines addresses which allow the stream to be merged and sorted into many sub-streams. This creates many 'Ports' which

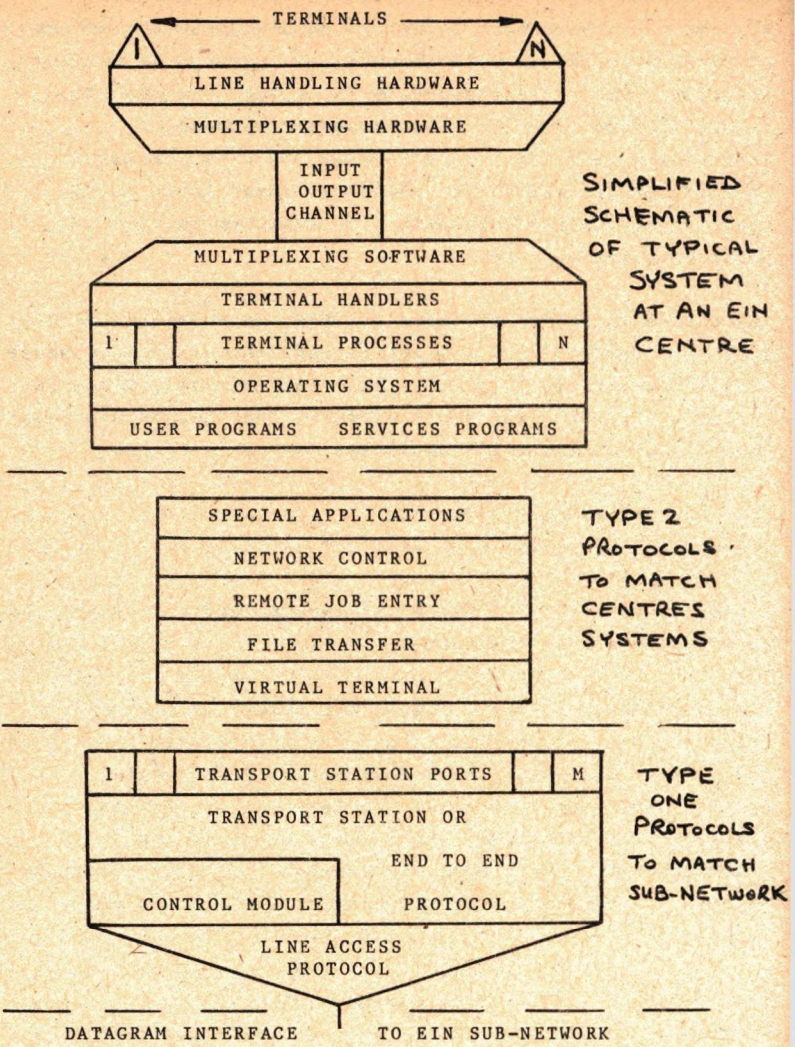


figure 2 EIN Protocols

time share the sub-network link to provide communication with Ports on other Transport Stations. A protocol is defined governing the interactions between Transport Stations so they may provide services to processes, representing higher level protocols, that are connected to their ports. Services include the maintenance of the order of packets in a sequence flowing between ports; the exchange of groups of packets forming messages or 'lettergrams', and the creation of liaisons (or calls) between pairs of ports. It is also possible to send an interrupt signal or 'telegram' which bypasses any queue of packets waiting for transm ssion via a liaison. This protocol was defined by experts from the signatories, and has been implemented by the Centres (ref 5).

THE NETWORK CONTROL MODULE (NCM) is situated above the line protocol at the same level as the Transport Station. It has the task of interacting with the sub-network facilities and with network control modules at remote sites in order to monitor the behaviour of these facilities. The NCM may also contain a mechanism for signalling alarms to operators' consoles at attached computers, and for allowing them to communicate with and to control the communications sub-network.

Once the end-to-end protocol has been implemented at a Subscriber Computer (SC) attached to a network, a large number of ports are available for a liaison with ports in other SCs . The communication between these ports can be regarded as error free, because problems of retransmission, sequencing, etc. are taken care of in the lower levels of protocol. It is therefore convenient to regard each pair of ports as connected directly together with just a delay inserted in the path between them. The line protocol and the transport station, therefore, greatly simplify the use of the communications sub-network.

8.2 System Matching Protocols

The second type of protocol is intended to make computer systems more able to interact together. Very active discussion of such protocols is in progress within the EIN community and a number of papers now exist on various aspects of high level protocols. It is likely that a number of changes in current ideas will occur, but an indication of present thinking on possible higher level protocols is given in the centre part of Figure 2, which is briefly described as follows:-

THE VIRTUAL TERMINAL (VT) is able to perform an arbitrary set of terminal-like operations, typical of a range of real terminals that might be connected to the network. Handlers in each of the Centres' systems can be written to manipulate the virtual terminal and this allows the applications programs in Subscriber Computers to drive the virtual terminal through the sub-network in a standard way. It is then necessary to associate a terminal handler with each type of physical terminal, in order to translate the virtual terminal's commands and responses into those appropriate for the real terminal.

A NETWORK CONTROL LANGUAGE (NCL) or network command language is highly desirable to allow users to communicate with the network services in a standard manner. It is, however, difficult to define a universal language, and a number of application oriented languages may be the best practical way to suit different users.

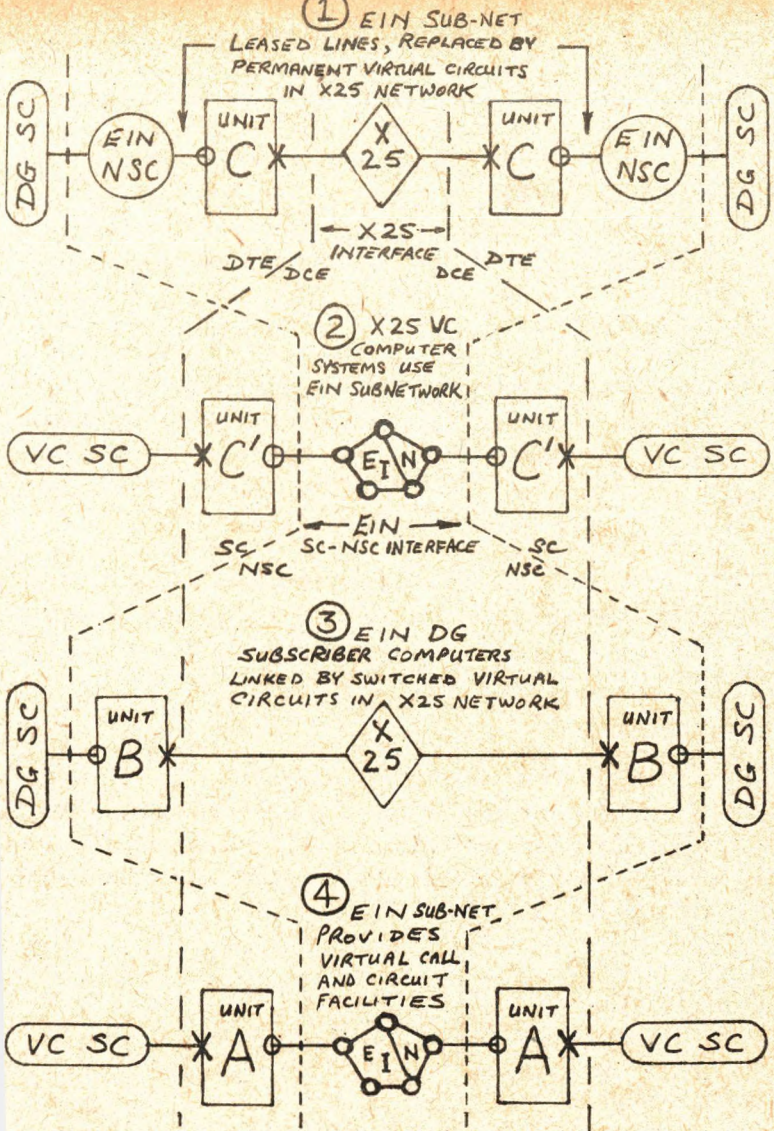
THE FILE TRANSFER PROTOCOL (FTP) is required to transfer a structured file of information from one system to another. The definition of a uniform structure for files exchanged between different systems connected by a network makes it possible to map from the particular file structures used b each system onto the standard FTP.

In 1974 when the design of the EIN sub-network was frozen, so that construction could begin, the CCITT discussions centred on the Datagram facility. Accordingly, this was made the basis of the design. Since then, work in CCITT has continued, and agreement has been reached on recommendation X25 (ref 7) which describes a virtual circuit interface for packet terminals.

The agreement by CCITT of recommendation X25 has been an important development for the authorised carriers who are providing new data networks in several parts of the world, for it offers a uniform interface for users of all of their networks. This poses some problems for EIN, because X25 duplicates some of the facilities that are provided by the Transport Station, described earlier in this paper, which has already been implemented by the Centres. However, the higher levels of protocol are independent of the transport mechanism, and the work of EIN in these areas should be relevant for users of all types of network.

In May 1977 the Management Committee for EIN gave its approval for the development of an adaptor unit to convert EIN to operate with X25 interaces. A number of variations of this unit are possible as is shown in Figure 3.

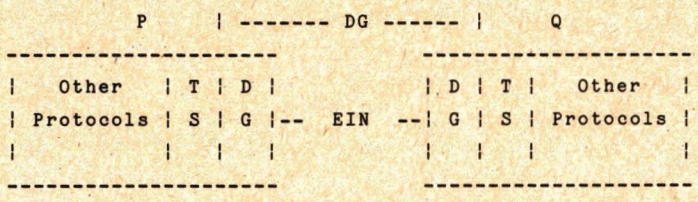
The simplest version, Unit C, has two roles; firstly it allows the leased lines presently connecting the NSCs to be replaced by permanent virtual circuits and secondly it permits interworking between existing EIN subscribers, and a subscriber using an X25 interface. These possibilities are detailed below:-



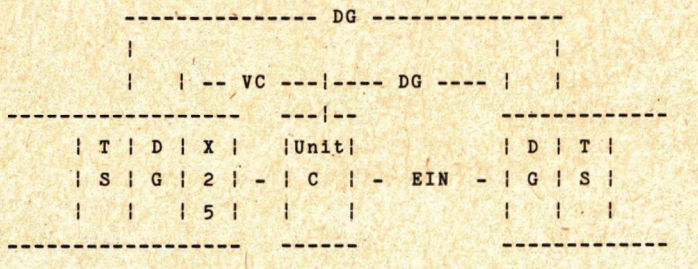
NOTE – All units have one datagram interface O and one virtual circuit interface X

figure 3 Types of X25 Adaptor Unit

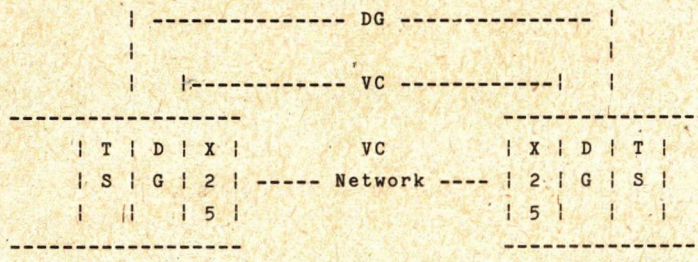
1. Existing EIN Centres P and Q communicate using datagrams



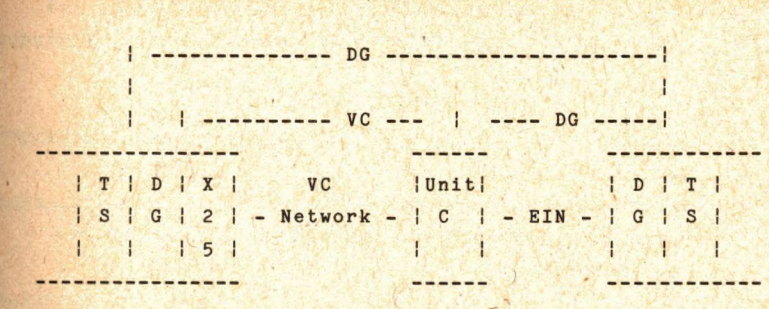
2. New EIN User(X25) connects through Unit C, using virtual circuit but implements existing EIN protocols on top of the Virtual Circuit Protocol.



3. Two independent users of a VC network may implement DG and upper levels of EIN protocols, thus exploiting EIN protocols and retaining an ability to communicate with EIN Centres as in 4.



4. New X25 User communicates through VC Network with present EIN Centres



Of the remaining two units, unit B allows an existing EIN subscriber to link to a network offering only an X25 interface, while a pair of units A allow the EIN sub-network to simulate an x25 network.

The development of the three varieties of the X25 adaptor unit will bring the sub-network up to date, and make EIN more suitable to cotinue in its role of an advanced network research project for Europe.

10. CONCLUSIONS

The European Informatics Network Project has already fulfilled many of the early expectations of the signatories to the Cost 11 Agreement, and, apart from its intended purpose of providing an operational computer network for European research scientists, has acted as a valuable catalyst in discussions about a future public network for Europe.

In many ways, however, the most important role of EIN is just beginning: namely, the fostering of co-operation between users of the new data services that will be coming into use in the near future. EIN has already served to bring together research workers co-operating in studies on

the design and implementation of networks, and now attention is concentrating on the problems of their use. It is in this area that EIN may offer its greatest rewards.

11. REFERENCES

1a Agreement on the Establishment of a European Informatics Network,
Brussels, 23 November 1971. Miscellaneous No.14 (1972) HMSO

1b BARBER DLA
A European Informatics Network: Achievement and Prospects
Proc. of ICCC76, Toronto, August 1976, 44-50.

2 PONCET F and TUCKER J B
The Design of the Packet Switching Network for the EIN Project
Proceedings of Eurocomp 75, London, September 1975, pp.301-314

3 WILKINSON K
A Network Control Centre for the European Informatics Network.
Proc Online Conference on Computer Networks (1977) to be published

4 REPTON C S and PONCET F
The EIN Communication Sub-Network: Principles and Practice
Proceedings of ICCC76, Toronto, August (1976) pp.523-531.

5 European Informatics Network End-to-End Protocol
EIN Report 76/033

TRANSPAC / A PUBLIC NETWORK PACKET DATA TRANSMISSION

FREDERIC PLATET

Administration Française des PTT

ABSTRACT

TRANSPAC is a public packet- switching network to be introduced in France by the French PTT by mid 1978.
This paper first provides a list of the objectives witch the French PTT have had in designing TRANSPAC. It then goes on with a description of the services offered, giving particular consideration to the public data network aspects of TRANSPAC. The network structure and its components at switching centers are then examined, giving importance to technical considerations.
Finally the general structure of the tariffs is described, emphasizing the distance-independent volume-sensitive characteristics of TRANSPAC Tariffs.

CONTENTS

1. INTRODUCTION

In 1972,.the CNET and the CCETT started studies on packets switching. In 1973, the "Direction Générale des Télécommunications" gaves priority to the development of such a technique because it could apply to a public Service. This resulted in a confirm commitment to start the development of a public packet switched data network named TRANSPAC. This network will be based on the results of the experimental network RCP (Réseau de commutation par Paquets) which has been operating at the beginning of 1975.
Expanded marketing studies were made involving in particular an association of large potential users of TRANSPAC.
Furthermore France took an active participation in the work of relevant international standards; in particular a strict collaboration between Canada, USA, U.K, Japan and France has lead to a CCITT recommendation for standardization of packet data-transmission network access interface: the X25 recommendation.
The following is a description of the services offered and the network structure.
Some indications will be given concerning the planning and the tariffs.

2. OBJECTIVES

The network objectives and consequently the characteristics can be presented in terms of public service, universal service, and security.

. Public service first means that any user must have access to the network witch, then, must offer a big capacity to satisfy large-scale wide users. But Economy due to rèssource sharing must allow small users to go to teleprocessing. Furthemore the service has to be extended throughout the French territory so that any terminal or computer in any location can be connected to the network. And this is reflected in the tariffs witch are distance independant.

. When speaking about universal service, we understand three main aspects:

- the service is conformm to international standards. Users and manufacturers are sure of generality of procedures. Users can buy theirs equipments on a world wide market, and manufacturers dispose of a wider market than a national one. Moreover interconnections with similar foreign networks are possible.
- access links are various; leased lines, switch telephone network,and also telex network.
- both kinds of services are offered, switch and permanent virtual circuit.

. At last, users have to be guarantee for performances and disponibility. This is achieved by a mesh network and a duplication of equipments. Furthermore to palliate failure of access link, a multiline procedure is offered, which is such that service is disrupted only when all physical lines are out of order. A single line failure only reduces the overall throughput on the access link. The availability of operations is insured by operating people working 24 hours a day. By day, operators monitor every node on-site. At night, the network is operated by the Management Center, with control over

the whole network.
The confidentiality of the information being passed over the network is preserved by locking devices in the nodes, while access to users'index can be restricted by protective mechanisms incorporated within the network, supplementary to those in existence at user system level. Where switched virtual circuits and datagrams are concerned, a firm or any group whatsoever, can indeed be protected from access to or from the outside thanks to the definition 'closed subscriber groups'.
Each subscriber can belong to one or more groups which can include the group of ordinary subscribers. For each one of these groups authorization can be given either for the transmission or for the reception of calls, or for both. Protection is guaranteed by means of two controls made on each call.
(1) The caller subscriber has to have authority for the transmission of calls in the group he has named in his call command.
(2) The subscriber being called will be authorized to receive calls in the group named by the caller.

3. SERVICES OFFERED

As far as access interface is concerned,TRANSPAC offers two main types: one for synchronous terminals which is X25, and one for asynchronous terminals which corresponds to three projects of recommendations: X3, X28, and X29.

3.1. The X25 protocols establihes the set of rules regulating the exchanges between a terminal or a computer (ETTD) and the network in order to allow communications with one or several others subscribers. The service offered is the "VIRTUAL CIRCUIT".
The main characteristics are:
- preservation of the order of the packets: the packets are

send in sequence and delivered in the same order to the receiver without loss or duplication.
- Flow control: each of the correspondent can regulate, in a selective manner, the sending bit rate of the other.
A virtual circuit can be:
- either permanent: that is to say it is setting up in a fixed manner between two given correspondents
- either switched: in this case, it is setting up and clearing down when any of both correspondents wants so.
X25 is a three layered interface:
. Level 1 is the physical and electrical parts: it is the modem interfaces of CCITT Recommendation V24 and V35: this means that transmission is performed in a four wires, full duplex manner.
. Level 2 is the frame level where HDLC (High level data link control) procedure is used, which is standardized by ISO. A special byte, the flag, is devoted to synchronisation of receiver. The transmission is bit sequence independant (data transparency) because a systematic insertion of one bit"0" is made after five consecutive bits"1". At last, transmission errors are detected and corrected; the detection is made by a frame Control Sequence(FCS) which is a cyclic redondant code of 16 bits; the correction is achieved by a mechanism of retransmission of wrong frame. This mechanism is based on the principe of frame numbering,acceptance, and permit to send with anticipation thank to the use of positioning window.
. Level 3 is the packet level which indicates the functions to be provided, the packet format and the rules of exchange.
- a multichannel service is offered; it enables tc dialog simultaneously with more than one subscriber over a single link.
- Control of virtual circuit: setting up and clearing down and resetting a circuit when necessary. Data transfer including a selective flow control applying to each communication, which preserves the order of the packets and is

based on the principe of the window: it allows the slavery of the speeds .Other parameters include the adaptation of packets length(with bit M=more data) and the interrupt packet which is not submitted to flow control.

3.2. For asynchronous compatible TTY terminals, adaptation is made by the network. Actually, the access node contains a PAD - Packet Assembler Disambler program - which main function is to made packets from characters - chiefly by Return Carriage or by time constant - and in the reserve direction to deliver character from packets.
Physical and electrical level are the modem interfaces up to 1200bit/s. Two projects of recommendations deals with the control procedure between PAD and remote ETTDX25 and between PAD and asynchronous terminal as well as data transfer.

4. NETWORK STRUCTURE

4.1. Means of access (fig. 1)
It will be possible for subscriber equipment to access the nearest TRANSPAC entry point(switching centre, or multiplexor) by one of the following means:
- leased line for all speeds(up to 48 kilobits per second per line)
- switched telephone network in the case of asynchronous terminals, for speeds of up to 300bauds;
- 50 baud telex network.

Base band converters or modems on the subscriber's premises will be automatically supplied by the French Administration as part of the TRANSPAC service(except for 300 baud asynchronous modems). The speed categories offered will be suitable for all asynchronous terminals up to 1200 bauds and for synchronous speeds of 2,400, 4,800, 9,600, 48,000 bits per second(also 19,200 bits per second close to access points)

The packet interleaving principle used on the network can be retained for the station line of a subscriber(generally a computer) who converses simultaneously with more than one subscriber: this type of connection is known as multi-access or multi-channel and enables this multiple dialogue to be conducted over a single link.

4.2. Architecture (fig. 2)

The network is made of components geographically distributed which are the following:

. Switching centers making up a mesh network, and local centers to give a better penetration of country and making no transit. Both of them are made of same equipments, so the distinction is due to operating mode.

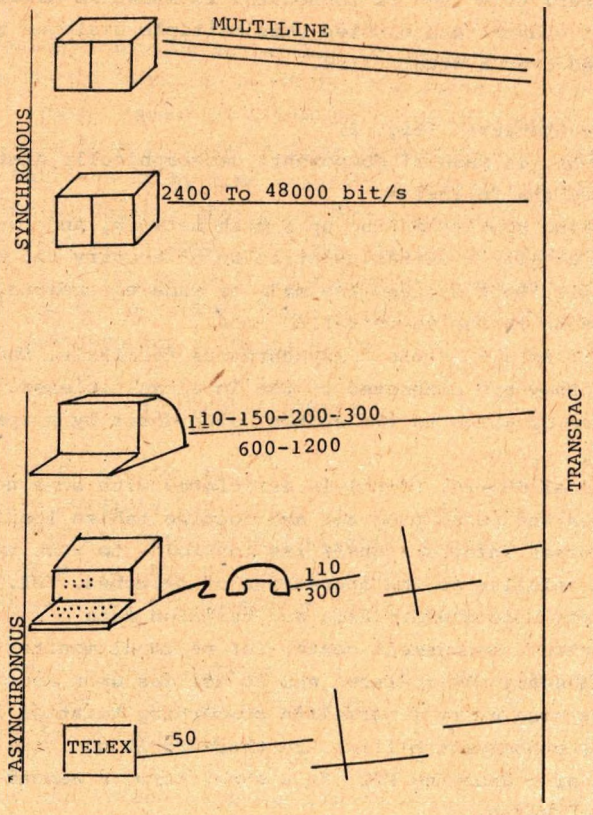
. Multiplexors to connect asynchronous terminals. When remote, they are connected to the local multiplexor, the interface of which to the switch being dealt by a special handler.

. PCL(Local Control Point) is associated with some nodes and it allows the local programs and routing tables loading in the switches which are under its control - to pick up informations relative to the operations of switches, MUX, and lines - to record alarm data and taxation data -

. The Network management center for national monitoring. It receives every major alarm and it decides upon routing. It receives also every information concerning taxation in order to allow subsequent billing. Furthermore it can take the control of a decaying PCL. In a short term, a second center will be created.

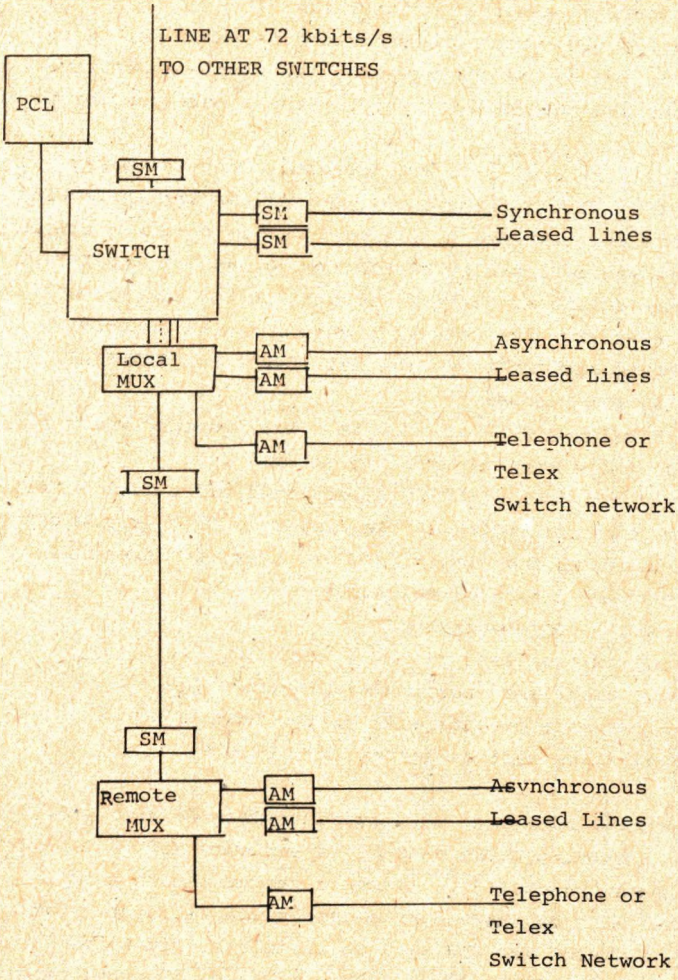
. The links between switchs are, at last, two 72 kbit/s on telephone primary groups. They are managed according to a multiline procedure.

. The subscribers links include modems(except 300 bauds) witch give possibility of telemaintenance(by loops)



MEANS OF ACCESS

- FIG. 1 -



ARCHITECTURE

- FIG. 2 -

4.3. Switching center structure (fig. 3)

. The principle, which the structure is based on, is to share, in two separate computers, the simple repetitive functions and the function which needs a more complex treatment.

. Level 1 and level 2 of X25 protocol. That is to say physical level and frame level, as well as data switching functions are realized in a specific mini computer called CP50. It was designed by french compagnies TRT and TIT.

. The management of virtual circuit and the control of switch operation are made in a standard mini computer playing the part of a control unit(C.U). For TRANSPAC, MITRA 125 was choosen.

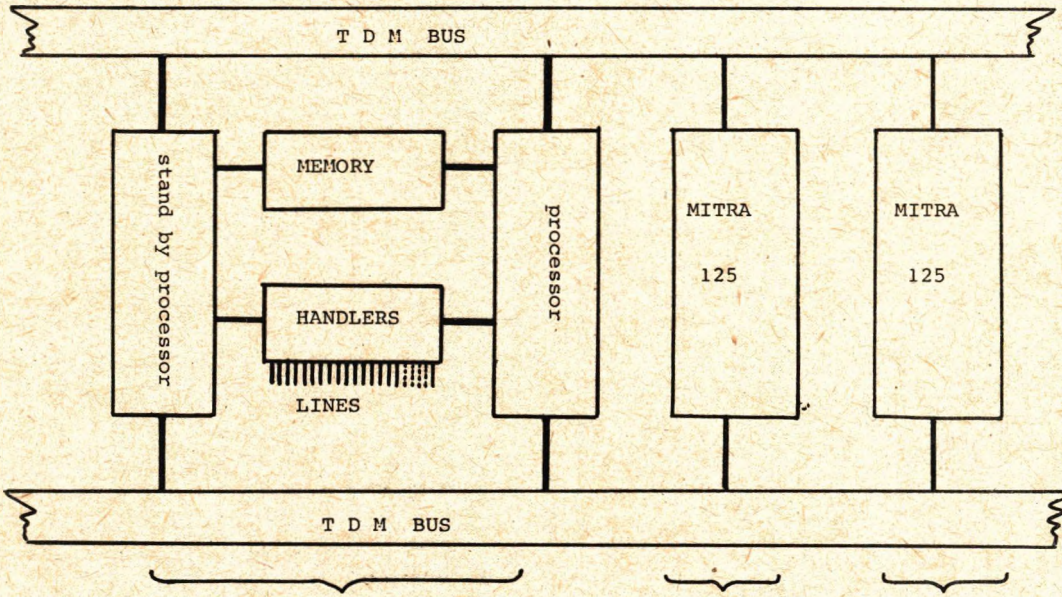
The link between CP50 and C.U is a bus temporel which is able to handle several CP50 and several C.U.

Each CP50 can deal with 500 subscribers and switch 1 Mbit/s Now, we have limited at 16 the number of CP50 in one center. Except handlers and memory in CP50, every equipment is duplicated by reason of reability.

. The main performances are:

- Setting up time less than 1.5 sec.
- Transit time: the mean value will less than 150 ms
- The error rate on bits will better than 10^{-12}
- Disponibility will be better than 99,99%

4.4. Network operation will start in June 78 with 4 Switches In the following three months, 12 switches and 25 asynchronous access points(13 of witch in remote places from witches) will be installed in the French territory. Later on new switches and MUX will be installed. There, we foresee that there should be 25 switches and about 40 asynchronous sites in 1980 - 1981.



SWITCHING CENTER STRUCTURE - FIG. 3 -

5. TARIFFS

An extensive study has been conducted on the economical aspects of using TRANSPAC in the framework of teleprocessing systems, consulting with several potential customers. It has checked for typical samples of applications that the proposed tariffs enable offering in most cases an economically attractive solution.

The tariffs presented here do not have a firm value. Not taking into account general economic variations constraints, the official tariff when published may change their level in a range of ± 10%.

One may notice several innovations in these tariffs, detailed in the tables in the following pages:

- monthly rental (1)is "all inclusive"(line and subscribers modem) and independent of the distance to the switch: no geographic region is at a disadvantage.
- the usage of the service is mainly charged according to the volume of data transmitted (4) ,in the case of switched virtual circuits, the charge is a function of the duration of the communication (3), remain very low, and is mainly intended as an incitation to the user to clear the communication if he remains inactive during a long period.
- the charges for using the network are independent of the geographic locations of the parties.
- installation charge:

800 F per connection up to 300 bit/s
1600 F above

(1) Monthly Subscription

Package including all the network access harware rental

- subscriber port at the switch
- subscriber modem(except for the 300bit/s modem which has to be supplied by the user)
- dedicated link to the Transpac access point

subscriber line data rate	monthly rental
up to 300 bit/s	270 F per month
600	570
1200	600
2400	650
4800	680
9600	750
19200	800
48000	1300

- rental is independent of geographic location of the user, however:

. Access at 19200 bit/s is only available close to the switches.

. Access at 48000 bit/s is available everywhere. In certain geographic locations a subscription fee of 5000 F/month may be required. For these both access types consult the sales department.

(2) Rental of a permanent virtual circuit OR

(3) Call duration charge of a switched virtual circuit

Either one or the other of these two tariffs applies according to the type of virtual circuit. Charging depends on the"throughput class" of the circuit(typical data transmission rate of the virtual circuit link).

This value is choosen by the user, either at subscription time or dynamically on a per call basis, to cover its needs, taken into account that it could reach this maximum data transmission rate contracted but not exceed it.

Troughput Class	.2.Rental of a permanent virtual circuit	.3. Call duration charge (*)
up to 300 bit/s	90 F per month	0.01 F per min
600	90	0.01
1 200	90	0.01
2 400	180	0.02
4 800	270	0.03
9 600	360	0.04
19 200	720	0.08

(*) OFF PEAK DISCOUNT

.(4.) Volume charge

The unit of volume charge of transmitted data is a segment of 64 octets.

This permits to the user to be charged on the same way if he uses a maximum packet length of 32 or 128 octets(those two maximum packet length are available on TRANSPAC user interface).

After having chosen a maximum packet length (either at subscription time or at the Call request on a per call basis), the user chains its packets into SEQUENCES by the mean of a marker called the "MORE DATA BIT".

For the purpose of charging, the network measures each SEQUENCE with an integral number of SEGMENTS.

Example:

User A transmits a sequence of 900 octets:

$\frac{900}{64} = 14.0625$

Volume charge: 15 segments

i.e. $\frac{15 \times 64}{1000} = 0.96$ Koctets

Value of the volume charge: 0.05F per Koctet (*)

A minimum charge of n segments per call applies

Off peak discount

Off. peak discount	Applicable for ITEMS 3 and 4 only		
	8 AM-7 PM	7 PM-12 PM 6 AM- 8 AM	0 AM-6 AM
Working days		40%	80%
Saturdays	40%	40%	80%
Sundays and Holidays	80%	80%	80%

Options

- Multiline access
The host computer is connected to its normal access node by more than one physical line:
The full monthly subscription rate applies for each line.

- Multichannel access
The number of channels subscribed for defines the max. number of virtual circuits that can be simultaneously established at a given time:
10 F/month per additional logical channel

- Closed user group
Per subscriber belonging to a closed user group: 10F/month
A single bill is provided totalling the dues of all the subscribers of the group.

A quantity discount applies on the total volume charge of the group:

0 —— 10^6 Koctets/month : 0.05F/Koctets
10^6—— 2.10^6 Koctets/month : 0.04F/Koctets
$> 2.10^6$ Koctet/month : 0.03F/Koctets

This discount is cumulative with off peak discount.

- Switched network access (telephone or telex)

The user can choose either a:

. General use port . Allows a general call in TRANSPAC network.

. Private port ; Assigned permanently to a fixed TRANSPAC subscriber, it is equivalent to a permanent virtual circuit between the port and one host.

	Private port monthly rental	General use port Duration charge
Telephone access	180 F / month	0.05 F/ minute
—OR—		
Telex access	280 F / month	0.08 F/ minute

Access port charge(either item 5 or 6) includes virtual circuit charge.

In any case the volume charge (4) is the same as for Dedicated Line Access.

6. CONCLUSION

To answer the needs of data market, we believe that packet switching technology is the best appropriate one, in particular to satisfy the need of spreading the traffic, we must give the possibility to switch data towards changing destination, and to establish a communication between equipments of different speed.
Furthermore, several countries have got already such an operating network or intend to do so. They offer their own standards, but offer, or will offer, the Standardized X25 Interface.
It will be possible to realize interconnections with those similar network and then an extended service all over the world will be available.

REFERENCES:

- R. DESPRES, P. PICARD, F. PLATET "Discussion of technical choices made for TRANSPAC" ISS 1976 - p. 511-4-1

- F. PLATET "les besoins en commutation de données _ Les différentes solutions _ La commutation par paquet _ Le réseau public TRANSPAC"

- J-F. GUILBERT "Un nouveau moyen au service de la Télégestion: TRANSPAC"_ "Travail et methodes n° 320-DEC 75 p 3 à 14 .

- Brochure generale"TRANSPAC _ reseau public de transmission de données par paquets" . Service technico-commercial de TRANSPAC

CIGALE, THE PACKET SWITCHING MACHINE OF THE CYCLADES COMPUTER NETWORK

Louis POUZIN
Director PROJ-PILOT
Institut de Recherche d'Informatique et d'Automatique /IRIA/
FRANCE

ABSTRACT

CIGALE is designed to handle message transfer between computers at the best possible speed. Messages are handled independently, as letters in the mail. There are no end-to-end functions, but any higher level protocol may be used to control message flows. Hosts may be connected through several lines, and there can be several hosts on a line. The host adress space is independent from the topology. Regions make up a 2nd level adressing and routing. They can be viewed as local networks. Also, hosts can be networks. A network name is a 3rd level intended for inter-network traffic. Various services may be added for some classes of users and terminals may be connected through concentrartors, or a packet interface. Congestion control and routing are associated to control the message flow. A simple message interface is all that is needed to interconnect networks. Complicated interfaces make networks incompatible. The best value is transparency.

1. INTRODUCTION

CYCLADES [1] is a general purpose computer network being installed in France under government sponsorship (Fig. 1). Its goals are to foster experiment and develop know-how in computer-to-computer communications, data transmission techniques, and distributed data bases. CYCLADES is also to be an operational tool for the French Administration. The project started on the beginning of 1972. Various demonstrations of distributed activities have been presented in November 1973, including 4 host computers and a packet switch. The complete network of 16 computers, linked by a 5-node packet switching network, is to be on the air by mid-1974. Additional hosts and nodes may be connected later on.

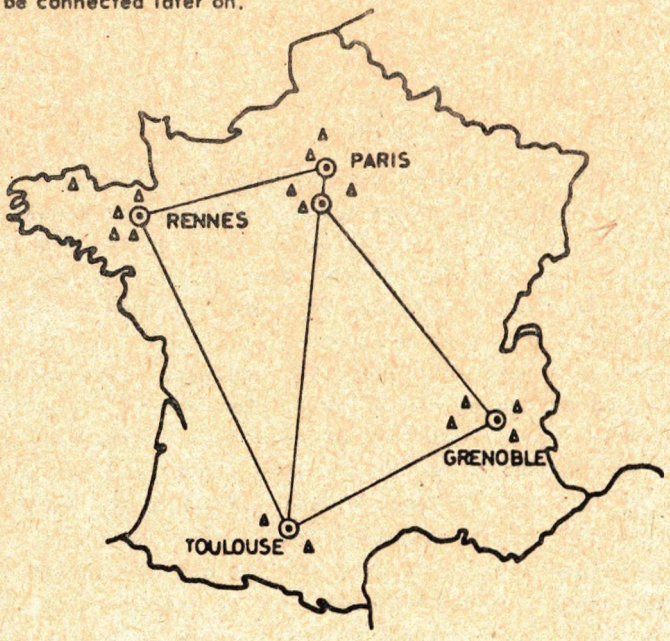


Fig. 1 - CYCLADES topology

The communications network within CYCLADES is called CIGALE [2, 3]. It is a store-and-forward packet switching network similar to the one included in Arpanet [4]. A brief summary of the CIGALE description will be given here. More details may be found in the referenced papers.

2. THE CIGALE NETWORK

Nodes are MITRA-15 minicomputers, with 16 K words (16 bits). Except for a teletype, no other peripheral is necessary. Communication lines are point-to-point PTT leased lines ranging from 4.8 Kbs to 48 Kbs. All host computers are connected via telephone lines and V24 interfaces, typically using 19.2 Kbs base-band modulation on voice grade circuits (Fig. 2). Transmission procedures between hosts and nodes are those of the various computer manufacturers, in transparent binary synchronous mode.

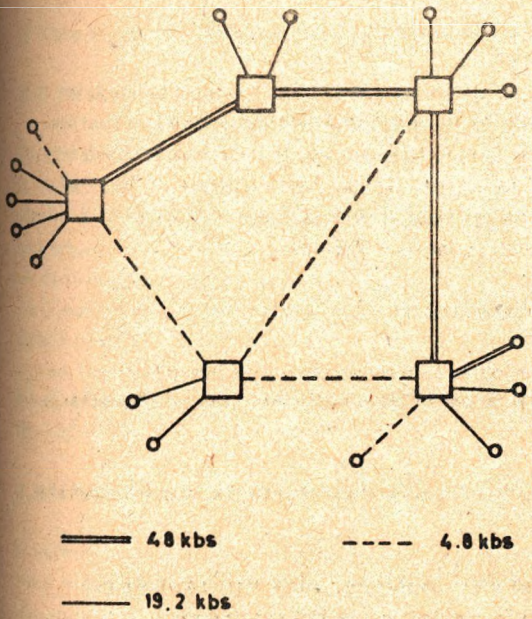


Fig. 2 - CIGALE topology

Specific components within CIGALE provide for some basic services, such as : echoing, statistics recording, debugging aids, node configuration, artificial traffic, clock. Other services may be requested with regard to a particular packet : priority, tracing, routing. Switching is implemented on a very simple manner by asynchronous processes feeding each other through queues.

A control center receives various reports and helps in locating failing components. Every node can be reloaded individually from this center by sending its program in packets.

3. DESIGN OPTIONS

The remainder of this paper will concentrate on the choices made at the design stage, and discuss the rationale behind. Indeed, a distinctive character of CIGALE is not its gadgetry, but its basic simplicity. The result is an unusual flexibility in handling all sorts of protocols and connections. It will be a trivial task to connect CIGALE with other packet switching networks, as long as they do just that.

3.1. Computer-to-computer traffic :

No statistics are yet available. Consequently, only prospective assumptions can be made. A set of frequently mentionned characteristics are following :

a) Bursty traffic, ranging from sparse short messages to a file transfer steady flow.

b) Very short delay, ideally nil. Actually, a few milliseconds or seconds may be acceptable, depending on the environment.

c) Error rate less than 10^{-10} bit.

d) If an initial set up time is required, it should be small as compared to transmission time.

e) Transmission availability at least 98 % of the time.

f) Throughput as high as suitable for an I-O channel (several Mbs).

g) Traffic between host and network multiplexed on a small number of I-O ports, typically 2 for reliability.

h) Standard communication interface where applicable.

i) Host computers are heterogeneous.

j) Data to be carried unaltered, in a transparent mode.

3.2. Packet switching :

Direct circuits between computers would not meet conditions e and g. If circuits are switched, conditions d and f would not be met. Other media could be considered [5, 6, 7] (radio, satellite, coaxial cables), but equipment and infrastructures are not available. Packet switching on mini-computers has been introduced recently, but its validity is now well established. It can be implemented with existing hardware at a small cost. All constraints above can be reasonably met, within the speed limits of presently available transmission lines, (typically 48 Kbs). When PCM links are installed, they will provide for higher throughput and shorter delay.

3.3. Transit delay :

Conventional message switching systems, using secondary storage, take a few seconds, or a few hours, to carry a message through, depending on priority [8]. On the other hand, in the range of 100 ms, Arpanet is the only known network of large geographical extension. CIGALE is aiming at this domain of performance.

Very distinct architectures result from these two different ranges of objectives. CIGALE is not designed to cater for traffic with deferred delivery and long term storage. However nothing prevents a conventional message switching system from using CIGALE as a communication tool between centers. As CIGALE basic service is fast transit delay, this objective carries a major weight in the choice of other characteristics of the network.

3.4. Added services :

This objective of fast transit delay suggests eliminating some mechanisms, which certain kinds of users might find desirable in a data communications system, e.g. for terminal handling[9]. But various services may be added on, as custom-tailored devices, for certain classes of users. E. g. asynchronous character terminals, data conversion, hot circuits, restricted traffic, multiple addresses, etc... This can be implemented as pluggable pieces of hardware/software. It may also entail some additional delay or restricted throughput in carrying messages.

This approach, which is classical in properly designed systems, insulates basic functions from market oriented services [10]. Indeed, the former must be kept very stable and customer independent if it has to be reliable, while the latter may be introduced on demand for changing needs. Customers who rely on an interface providing only basic facilities are guaranteed to protect their investment against modifications likely to occur in more specialized services. Furthermore, they are not penalised in being forced to use functions that they do not need.

Computer-to-computer traffic is predictably going to be the most demanding data transmission consumer in terms of throughput and transit delay. Furthermore, computers do not require any help in managing their own data transfer. Therefore, the communications system they need should offer the simplest possible functions and the highest possible throughput.

Along these lines, CIGALE offers a basic service, message transfer, and allows grafting additional functions through more specialized interfaces. The development of micro-components with a parallel decrease in cost will gradually make these devices appear as trivial hardware integrated in user equipment. For example, character oriented devices will soon be equipped with a message interface, so that they be linked directly to a network. They will look like simple computers. This is one of the reasons not to consider terminal handling in a packet switching network, since present terminal interfaces will fade out.

3.5. Packet size :

The term *packet* refers here to bit strings exchanged between CIGALE nodes. Various studies might be made to evaluate an optimum size depending on a number of parameters. Practically, as long as efficiency and delay stay within an acceptable range, it seems that its main virtue is to be stable. This allows an adequate optimization of customer software and hardware.

Furthermore, it will be necessary to interconnect different packet switching networks. A generally agreed maximum size for a packet would be highly desirable in order to simplify interface mechanisms. A proposal has been made to set 255 octets as the maximum text size [11]. This is long enough to accept packets from existing networks, and it suits nicely present computer hardware. This choice has been made for the British Post-Office network presently being implemented[12]. So it is for CIGALE.

3.6. Packet header :

It contains the following items :

- Header format	4 bits
- Header length	4 bits
- Text length	8 bits
- Packet identification	16 bits
- Facilities, accounting	16 bits
- Destination network	8 bits
- Source network	8 bits
- Destination host	16 bits
- Source host	16 bits

The *header* length is 96 bits, a multiple of 6, 8, 12, 16, 24, 32. This facilitates formatting on most types of computers. The *format* field anticipates future adjustments in connection with other networks. CIGALE may have to handle various packet formats in a mixed traffic. They will be segregated easily.

The *identification* field is left for the user to identify his packets. It is not processed nor altered by CIGALE, but it is used to report about anomalies, or when special services are invoked. An obvious use is to contain all or part of a tag for multiplexing internal users within a host along with some serial number. But this belongs to host level protocols, and CIGALE does not require any specific scheme, since it is transparent to any outer protocol.

A 3-bit *time-out* field will be introduced in a future version, so that older packets be exterminated rather than roam about the network for any reason. It will not appear at the host interface.

3.7. Host interface :

The term *host* refers here to any computer connected to CIGALE. A host is assumed to be located at some distance away from a CIGALE node, and must use PTT lines. Indeed, putting a node on host premises carries several drawbacks :

- it increases the network cost and places an additional transit delay.
- it reduces reliability, since the node is under user's control.
- it is a security risk, as it can be tapped and tampered with.

A single line between a host and CIGALE would be somewhat unreliable. Two lines in parallel would not help in case of node failure. Therefore, it is possible to run several lines between a host and several CIGALE nodes. The traffic can be dispatched arbitrarily over all lines, according to speed and load. It is also a way to increase the maximum throughput when lines cannot be speeded up.

In order to avoid any predicament usually associated with special hardware, the electrical interface is typically CCITT V24 up to 19.2 Kbs. At higher speeds, V35 is required.

Also, it was found undesirable to introduce a special transmission procedure between a host and CIGALE. Current manufacturer procedures are accepted, even though they are generally not very efficient. Hence no modifications to operating systems are necessary, as long as a transparent binary synchronous procedure is available, which happens in most cases. A consequence is that host-to-host protocols may be implemented at user level without modification to operating systems. This is of course immaterial in CIGALE, but it makes quite a difference for users who want to keep their system under manufacturer's responsibility.

A 16-bit cyclical checksum is a common feature of synchronous transmission. Any polynomial is acceptable, but the ISO standard $X^{16} + X^{12} + X^5 + 1$ is recommended to maintain the error rate below $10 \cdot 10^{}$ per bit.

3.8. Message size

In some networks the quantum of information exchanged between nodes is different from the one exchanged between host and node [13, 14]. It is called a *message*, in order to prevent confusion. This need arises when message sizes adapted to user traffic are too different from an adequate packet size. But computers can exchange messages of many different sizes :

- a few characters for control messages
- a few dozens of characters for transactions
- a few hundreds of characters for file records
- millions of characters for complete files.

Since we lack statistics it would not make sense to pick an ideal size. Furthermore, packetizing costs overhead. Since a host is to fragment his data anyway, the advantages of putting an additional level of fragmentation do not appear to compensate for its inconvenience [15]. Consequently, a host message is simply a packet.

A possible objection is that too short host messages increase I-O overhead. Let us remark that this is mainly traceable to a clumsy design of communications software in some operating systems. I-O would be better handled by direct memory access. On the other hand, if this turns out to be a critical factor, some blocking scheme might be installed, to wrap several unrelated packets into a single I-O burst.

3.9. Name space :

Existing networks consider message addresses as meaning some physical component : a node, a line. As a result, there is a solid coupling between host addresses and network topology. CIGALE implements an abstract name space independent from the physical components. Consequently there is no constraint in naming hosts.

Furthermore, messages may be delivered to a single host address from several distinct nodes. Also, several host addresses may be reached over the same line. The first case is intended for reliability, as said before-hand. The second case allows several *logical* hosts to be connected on the same line. This is particularly convenient for having several distinct host protocols within the same host (as in CYCLADES), or to have several virtual hosts (as in IBM-CP 67), or to reach several hosts through a front end computer, or to have a set of hosts making up a private network. Actually, the exact nature of a host is immaterial in CIGALE ; it is a name capable of sending and receiving messages over one or several lines.

Part of the CIGALE name space is reserved for its internal components, such as software modules providing special services for network control and diagnostics aids. This saves the overhead and complexity associated with specially formatted packets. As far as switching is concerned, CIGALE handles only one packet format. But an internal component may be a real host, considered as part of CIGALE. This is a handy way to plug in any desirable function without making any technical modification to the CIGALE software, except putting a name in tables. This would be quite of a hang-up if network services were designated with special bits in special fields of special packets. Some variations in the address format allow to locate an internal component within a particular node, some nodes, or all the nodes. On this way network services may be distributed according to traffic patterns, and even relocated dynamically, should the need arise.

A brute application of the previous principles would result in every node containing the list of all possible addresses (internal components and external hosts). This is perfectly acceptable for small networks, but not large ones, as lists would be too bulky. Therefore, the CIGALE name space is divided up into *regions*, and host names are prefixed

with a region name, like telephone numbers. Thus, only local hosts and other regions need be listed in any node. Addressing and routing is based on host name within a region, and on region name across regions. In this respect, CIGALE is an aggregate of local CIGALE's (Fig. 3).

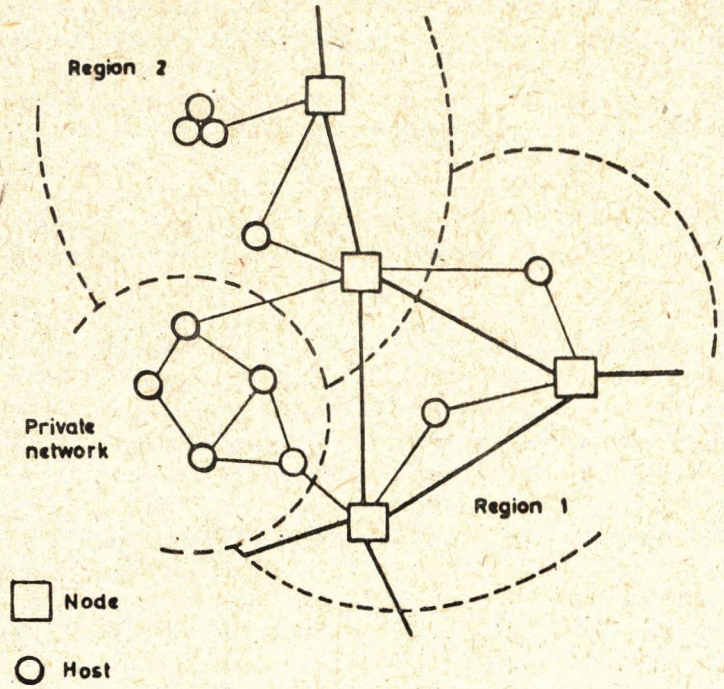


Fig. 3 - Address space

A region can contain one or several nodes, or not at all. This allows implementing an addressing plan without necessarily putting immediately as many nodes as regions. In addition, a host can be linked to nodes of different regions, even though its name belongs to one. This is intended ly restricted to hosts which cannot avoid straddling regions due to their geographical location. Or else address tables could become too large.

This name space is only a first step towards a general network interconnection scheme. In a multi-network context, the total name space is to be tree-structured at several levels [16]. Local networks (or regions) are to carry local traffic using shorter addresses, and long distance traffic using longer addresses. A common addressing plan is a prerequisite to network interconnection.

3.10. Circuits :

This term means a point-to-point logical connection established between two network users, be they hosts, or some entities within hosts. This can be simulated on both ends within the network [17], or even implemented physically as a fixed path through specific resources allocated within intermediate nodes (buffers, names, slots) [14]. Another common terminology is *call set up* [12].

Circuits can be predefined, and their use requires some allocation schen [4]. Or they must be established on demand [12, 14]. In both cases a initial phase is mandatory before message transfer can take place.

Circuits may be useful for simple terminals which carry on a conversation with a single correspondent for a certain period of time. This is typical of time-sharing and remote batch users. On the other hand terminals may switch rapidly between various correspondents, when they participate in distributed activities, such as data base inquiries, monitoring. They may even have to handle several conversations in parallel, i.e. several circuits.

The host software is necessarily designed to manage a variety of parallel and independent activities, including logical connections with other hosts over a network. This is precisely the core of host-host protocols, which should be completely insulated from packet network characteristics [10]. Setting up network circuits is an additional constraint without any useful counterpart. Presumably, it is also an additio-

nal trouble-maker, since circuits will have jinxes of their own requiring special recovery procedures. Moreover circuit shortage or overhead can create congestion without any real traffic. One also looses the reliability of multiple links between host and network.

Since circuits present far more undesirable aspects than useful ones they do not exist in CIGALE. However, they may be implemented on various ways as added services, as long as some users request it. Fixed or switched circuits may be custom-tailored through private interfaces[17].

Actually packet switching networks with virtual circuits are a completely different approach to data communications. Instead of offering a message transfer service, like a mail system, they simulate a circuit switch. But the nature of the circuits do not allow transmitting bits at any time, like on a couple of wires. That sort of circuits can only transmit packets of bits, in some predefined manner, which is imposed by the network. In other words, a specific transmission procedure is required. Packet switching is carried out inside the network, but this is no longer a service. As far as the user is concerned, he only gets a circuit with a transmission procedure, which is not too unfamiliar.

3.11. Sequencing :

This is a usual corollary of circuits, i.e. the property of delivering packets in the same order as they have been sent over a particular circuit.

Bit transmission should be independent from applications, but data transmission is not. Indeed, data are bits plus structure plus semantics. So far semantics is transferred through human channels. Structure is application dependent and is taken care of by specific user protocols. Some of them are sequential, some are not. Transactions, labeled records, statistics, are usually independent pieces of data and can be delivered in any order.

Since there are no circuits in CIGALE, there is no sequencing.

3.12. Error control :

In CIGALE packets are checked and acknowledged between nodes. However node and line failures coupled with adaptive routing may result in packets being lost or duplicated. Consequently, some control mechanism is necessary to catch this type of error. It can only be done as part of a transmission procedure between a pair of correspondents. There is none in CIGALE, for several reasons :
- due to multiple links between host and CIGALE, there is no correspondent pair at network level,
- user protocols require end-to-end error control, since host mechanisms and host-node lines may also fail.

Consequently, end-to-end error control within CIGALE would not be sufficient, would not easily fit multiple links, and would introduce additional overhead [15].

As a general rule, protocols using CIGALE should include message error control. This is done in CYCLADES [18, 19]. Actually these mechanisms are an intrinsic part of host-to-host protocols.

3.13. Flow control :

This term covers usually mechanisms intended to keep a sender process from overrunning a receiver. Again this implies end-to-end control between a pair of correspondents. Only transmission lines are controlled in CIGALE. On the other hand, error and flow controls can be identical mechanisms, as they are no more than producer-consumer relationships [10]. They should normally be part of host-to-host protocols. This is done in CYCLADES.

End-to-end flow control has sometimes been considered as a mechanism for controlling traffic congestion within a network. E.g. this idea was behind the *link* mechanism in Arpanet, which is a variety of circuit [4]. Since then, it has been shown that congestion could still develop, while throughput on links is restricted [20, 27]. Actually they are two different classes of problems.

3.14. Congestion :

This term means a state within which network throughput drops to nil, or almost, due to saturation of network resources, such as buffers, line bandwidth, processor time, etc... It is a ubiquitous problem in resource sharing systems, where supply is constant, while demand is at random. [21].

Congestion may be local if it is only limited to a few nodes, e.g. when a receiver stops working, but not senders. Total congestion may also develop if the network gets so crammed with packets that they can hardly move. This phenomenon is somewhat elusive on real networks, as most of them are too small to allow for significant experiments. Actually most investigations are based on models and simulation [21, 22, 23].

Research pursued in Arpanet and NPL indicates that global network control should be obtained by using simple queue disciplines and storage allocation [24, 27]. Although CIGALE is small enough to dispense with sophisticated control techniques, it appeared worthwhile to experiment some variation of congestion control. Modelling studies have been undertaken, but no results were available at the time of the preparation of this paper. The basic idea is to couple routing and buffer allocation as two related facets of a global resource sharing exercise over the network [25].

Similarly to Arpanet [26], routing information propagates continuously throughout CIGALE. In addition to an estimation of transit delay, an estimation of available buffers is given per destination. Due to the hierarchical name space of CIGALE, a destination is actually a region, or a local host in a region. Again, this 2-level structure saves considerable overhead in handling routing tables. When solicited for entering packets, a node allows in only a limited number, based on buffer availability towards the requested direction.

At this point, it is clear that various traffic classes might be accomodated, e.g. a shortest delay class, within the limits of a few packets per destination, and a highest throughput class using most of the buffer supply on alternate routes towards the same destination. This scheme is expected to prevent local congestion instead of curing it once an increase in delay shows that it is developing.

Furthermore, each node may evaluate the total available capacity in the network, by adding up the supply for each destination. By convention, each node may be allowed to accept no more than a certain fraction of this total supply, until it gets fresh reports. Assuming that all nodes are solicited for an upsurge of entering traffic, the network might fill up to its maximum allowed capacity, but no more, so that traffic keep flowing. On the other hand, it is statistically predictable that only a few nodes will have to choke excess traffic at any given time. In this case, the propagation of the available network capacity results in a gradual absorption of the transient, following roughly a logarithmic law.

Thus, both local and total congestion are expected to be in control. Actually, this scheme is akin to the isarithmic technique [21], at least for controlling total congestion. A significant difference is that buffer supply is reevaluated constantly, instead of being assumed fixed. Indeed, a weakness of the isarithmic technique is that no error is supposed to occur. If a node smuggles permits in or out, the whole network gets out of hand. Also, network partitionning may throw traffic off balance. Such accidents should be corrected automatically in CIGALE.

As a safeguard, old packets will be destroyed, so that solid hang-up be excluded. But this is only a way to recover from a pathological condition, not a normal management policy. Actually, eliminating old packets is mainly intended to reduce the deviation of transit delays, so that host protocols be tuned for quick reaction on lost packets.

These are just examples of a more general capability (Fig. 4).

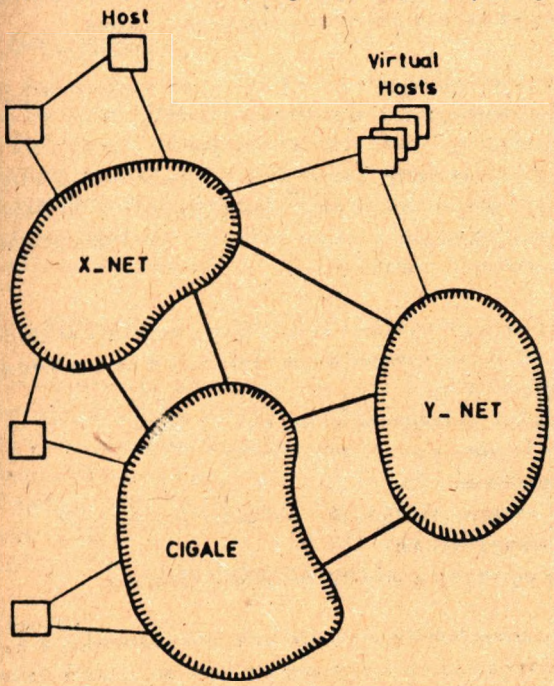


Fig. 4 - Network inter-connection

As long as a network carries messages as in a mail system, putting several networks together does not alter the picture, so that :

<network> : : = <network> <network> ... <network

Passing a message between networks reduces to a local border problem, i.e. a mutually agreed transmission procedure.

Routing a message is a distributed activity. A common naming scheme is necessary at network level, so that each network know on which direction it should send a message to the network containing the final

destination. Routing to the local user is only taken care of within the end network. As far as addressing and routing are concerned, inter-connected networks behave like nodes of a super-network.

This means that one might apply recursively the same approach to design a network, or a network of networks. Actually, there are real life differences which put some restrictions. Nodes of a network are homogeneous, and abide by the same rules. Networks tend to belong to different organizations, which do not necessarily wish to be homogeneous However, they may agree that they are willing to carry one another's messages under minimum constraints.

In that context a network interface like CIGALE is likely to be the least constraining. A few common agreements would be required among all networks [16]:
- an addressing plan to designate networks
- a basic header format
- a maximum packet size
- a plain packet delivery service, without additional function
- a set of accounting practices.

Other necessary agreements are just neighbour problems.

In view of inter-connection experiments with other networks, e.g. NPL, the CIGALE header contains source and destination network names, according to the CATENET proposal [16]. Linking with Arpanet would not raise any particular difficulty, as long as message length is restricted to one packet. On the other hand, experience now acquired tends to support the desirability of redesigning Arpanet protocols along simpler principles, such as in CYCLADES-CIGALE [28]. But there remain unsolved issues with EPSS-like systems [12].

It is clear that the CIGALE transparency is its major trump to provide a communication service between existing systems. Any additional well-wishing function tied with the external world is likely to be incompatible and detrimental to a good service. In particular, communi-

cations networks studded with all sorts of bells and chimes will end up as one of a kind networks, unable to communicate, unless an ad hoc kludge be interposed so that they at last exchange packets.

5. CONCLUSIONS

CIGALE has been developped as the communications tool of a computer network. However, it is designed so that it can be inserted as a component of larger systems to carry any data traffic in a transparent mode. Except for tim ng parameters, CIGALE does not impose any particular protocol. Furthermore, it may include custom-tailored components to implement specific services.

ACKNOWLEDGEMENTS

The CIGALE development has been a joint effort by the Délégation à l'Informatique, the French PTT, and IRIA. Many dedicated people have put some contribution at various stages of the construction. Among individuals who created major parts of the software are A. DANET, J.L. GRANGE, D.PREVOST and S. SEDILLOT. Packets would not have traveled very far without F. DENJEAN's interfacing skill. D. WALDEN (BBN) suggested many good ideas.

REFERENCES

[1] L. Pouzin, Presentation and major design aspects of the Cyclades computer network. 3rd Data Comm. Symp. Tampa (Nov. 73), 80-87.

[2] J.L. Grangé, L. Pouzin, Cigale, la machine de commutation de paquets du réseau Cyclades, Congrès AFCET, Rennes (Nov. 73), 249-263.

[3] L. Pouzin, Les choix de Cigale, Congrès AFCET, Rennes (Nov. 73), 265-274.

[4] F.E. Heart et al, The interface message processor for the ARPA computer network. SJCC (1970), 551-567.

[5] L. Roberts, Extensions of packet communication technology to a hand-held personal terminal. SJCC (1972), 295-298.

[6] L. Roberts, Dynamic allocation of satellite capacity through packet reservation, Nat. comp. conf. (June 1973), 711-716.

[7] I. Switzer, The cable television system as a computer communication network. Symp. on computer communication network and teletraffic, New York (Apr. 1972), 339-346.
[8] G.J. Brandt, G.J. Chretien, Methods to control and operate a message switching network, Symp. on computer communication network and teletraffic, New York (Apr. 1972), 263-276.
[9] S.M. Ornstein et al, The terminal IMP for the ARPA computer network. SJCC 1972, 243 254.
[10] L. Pouzin, Network Architectures and Components, 1st European workshop on computer networks, Arles (Apr. 1973), 227-265. IRIA édit.
[11] D.L.A. Barber, Hierarchical control of a communications channel, Nat. Phys. Lab. (Nov. 1972), 10 p.
[12] Experimental packet switched service, British Post Office Telecommunications (Nov. 1972), 50 p.
[13] BBN report n° 1822, Specifications for the interconnection of a host and an IMP (1973), 120 p.
[14] L. Tymes, Tymnet, a terminal oriented communications network. SJCC 1971, 211-216.
[15] L. Pouzin, Network protocols, NATO Advanced Study Institute on computer communication networks, Brighton (Sept. 1973), 25 p.
[16] L. Pouzin, A proposal for interconnecting packet switching networks, EUROCOMP, (May 1974), 13 p.
[17] L. Pouzin, Use of a packet switching network as circuit switching, Réseau CYCLADES, TRA 507 (April 1973), 5 p. Also IFIP-TC6.1, doc. 31, NIC 16735.
[18] M. Elie, H. Zimmermann et al, Spécifications fonctionnelles des stations de transport du réseau CYCLADES, SCH 502 (Nov. 1972), 105 p.
[19] M. Elie, H. Zimmermann, Vers une approche systématique des protocoles sur un réseau d'ordinateurs, Congrès AFCET, Rennes (Nov. 1973), 277-291.
[20] L. Kleinrock, Performance models and measurements of the ARPA computer network. Semin. AFCET, Paris (May 1972), 37 p.
[21] D.W. Davies, The control of congestion in packet switching networks, 2nd Symp. on problems in the optim. of data comm. Sys. (Oct. 1971), 46-49.
[22] H. Frank, R.E. Kahn, L. Kleinrock, Computer communication network design. Experience with theory and practice. SJCC 1972, 255-270.

[23] W.L. Price, Survey of NPL simulation studies of data networks, 1968-72, NPL report Com. 60 (Nov. 1972), 27 p.

[24] W.L. Price, Simulation of packet switching networks controlled on isarithmic principles. 3rd Data Comm. Symp. Tampa (Nov. 1973), 44-49.

[25] L. Pouzin, Another idea for congestion control in packet switching networks, Réseau CYCLADES, SCH 504 (Jan. 1973), 6 p. Also IFIP-TC6.1, doc. 21, NIC 14498.

[26] G.L. Fultz, L. Kleinrock, Adaptative routing techniques for store-and-forward computer-communication networks. Internat. Conf. on communications, Montréal (1971), 39-1 à 39-8.

[27] R.E. Kahn, Flow control in a resource sharing computer network, 2nd Symp. on problems in the optim. of data comm. sys. (Oct. 1971), 108-116.

[28] V. Cerf, An assessment of ARPANET protocols (To be published), 20 p.

FORMAL APPROACHES TO DESIGNING PROTOCOLS

J. Harangozó

Central Research Institute for Physics
Budapest, Hungary

ABSTRACT

Some approaches to describing formally the protocols of data communication or computer networks are dealt with. These approaches represent the first phase of a logical designing or implementing process. The paper discusses the basic idea of the approaches, the formal tool and the method of description. The suitability of the approaches the advantages and limitations are examined and a brief comparison of eight techniques – one of which was elaborated by the author – is made.

1. INTRODUCTION

For the logical design of a digital system, two related requirements [1] need to be satisfied, viz.

(1) to define the digital system in a formal way
(2) to translate the formal description into a logical diagram of the system.

The resultant logical diagram can be implemented by hardware, by software, or by firmware [2].

The present paper deals with the solutions of the first requirement, i.e. with the formal descriptions of a special kind of digital system, the protocols.

A protocol is a set of rules which determines the main conditions of the interaction between two elements of a data communication or a computer network. As the architecture of such a network is composed of different levels, the construction of protocols follows this multilevel structure, and possesses a system hierarchy [3].

The formal description is very important from the viewpoint of the implementation, particularly if such a protocol system has to be implemented which is described in the form of a living language, e.g. English. This form of description can be transformed by a formal method into another form which is much more convenient and useful, because the system hierarchy and the relationships between the adjacent levels of the hierarchy can be elucidated and the implementation will be easier. The formal way is applicable for examining the protocol system from the viewpoint of the possibility of deadlock situations. The interaction between the communicating processes will be clear, and the faults occurring in the coordination process can be avoided

2. FORMAL APPROACHES

As the necessity for formal definition of protocols arose in all data communication and computer networks designed and implemented earlier, some formal techniques have been suggested for this purpose. A brief discussion of these approaches is the subject of this section. The basic idea of the presented approaches, is examined as are the formal tools and the method used for the description, for which protocol level, and for what kind of protocols the approaches are suitable and what are the advantages and limitations. The first seven subsection present the approaches elaborated earlier. The eighth subsection is a brief discussion of the approach worked out by the author. The next main section compares the approaches.

2.1. Multilevel iteration

The technique of multilevel iteration [4] was not new when Bjorner [2] attempted to use it for the formal definition of data communication line control procedures as a special type of data link level protocols. The essence of the method is that initially it describes the total system, then its major subsystems and subsequently the components of each subsystem. Finally a complete logic of each component is given. This "top-down" presentation technique, which results in a system hierarchy on the basis of the layers of abstraction, is in close contact with Dijkstra's suggestions [5] and it is this approach that Bjorner [2] presents.

The half-duplex nonswitched multipoint data communication line control procedure with centralized operation [6] is defined by means of finite state automaton graphs. The levels of the abstraction are the following:

1) The first level reflects the basic relationship between the central device, e.g. the computer, and one or more

remote devices, e.g. keyboards or displays.

(2) The second level examines the major subsystems (the central and remote devices) separately taking into consideration the transactions between them on the basis of polling and selecting.

(3) The third level results in a more detailed graph reflecting the operation of the components of the subsystems.

(4) The fourth level describes the generation of characters flowing through the transmission lines.

(5) The fifth level shows the generation of bits of characters. As this procedure is a character oriented one the previous level is chosen for further examination and for implementation.

This methodology lends itself to the provision of layers of abstraction and to a well structured system for implementation. The finite state graph was chosen for modelling the system, because the data communication control procedures are of a finite state nature.

2.2. Formal grammar approach I

A linguistic approach to describing communication line control procedures 6 is presented by Hoffmann [7]. When such a procedure is followed, signals may be observed on the lines which represent a string of characters, transmitted one after the other in half-duplex mode. If another communication link is observed probably another sequence of characters occurs.

Are the sequences provided by the same control procedure or not?
Which is the set of sequences satisfying a communication line control procedure?
What are the common characteristics of such sequences?

– these are the main questions concerning a linguistic

approach to describing a communication protocol.

On the basis of the theory of formal languages 8 Hoffmann gives a grammar definition of a control procedure with a 5-tuple

$$C = (\Sigma, \Pi, P, \langle \text{frame G} \rangle, \langle \text{frame R} \rangle)$$

where

Σ is a finite, nonempty set of a terminal alphabet of characters;

Π is a finite, nonempty set of a nonterminal alphabet of phrases disjoint from Σ;

P is a finite, nonempty set of productions of the form $\Gamma \rightarrow \tau$, where $\Gamma \in \Pi$ and $\tau \in (\Sigma \cup \Pi)$;

$\langle$ frame G $\rangle$ and $\langle$ frame R $\rangle$ are the sentence symbols.

The terminal set consists of three subsets: the subset of graphics, control characters and alphabet extensions (special characters for observation of beginning and end of the existence of the link, for observation of the time-out condition and the redundancy check).

The set of productions is composed of five subsets; establisning productions, status switching productions, terminating productions, generation productions, and reception production.

The two sentence symbols are necessary for the description of generation and reception statuses.

A full-duplex transmission can be modelled by considering it to be a simultaneously operating pair of links each operating in unidirectional half-duplex mode [9,10,11]. Contention during the link establishment is not considered explicitly.

2.3. Automata composition

The serial composition of finite state automata is used af-

ter a method suggested by Rusbridge and Langsford [12]. An individual process having a single input and a single output may be specified abstractly as an automaton. The automaton seems to be a useful formalisation of a process because every state transition is associated with an imput-output event. An extension of the classical definition of automata is needed to allow the triggering of a transition by the arrival of an input symbol.

As tne protocol can be considered as a set of rules governing the interaction of two cooperating processes, and as these rules are part of the process definition the formal description of the protocol will be given after the formal description of the interaction of two cooperating processes.

As was shown earlier a single process may be formalized as an automaton. Now there is a need to represent the interaction of two of these generalized automata. These automata are linked by communication lines, which can be represented as a third relatively passive automaton, as between the departure of a symbol from the first automaton and the arrival of the same symbol to the second there is a non-zero time-interval, i.e. the symbol is transferred at a finite rate. As the interations of the automata models of processes are triggered by hand-shaking, the serial composition of these three automata provides an appropriate formal definition of a protocol.

The related paper [12] presents the technique how the composite automaton can be constructed from the component automata models of processes and tne line automaton. The method is illustrated by examples, some of which were taken from the host-to-host type of protocols.

The approach provides an unambiguous, machine independent specification of protocols, it provides a technique wherby

a given protocol may be proved, it provides a prescription of the protocol which can be easily implemented, and provides a description of protocols which could be adopted as a standard. The method has some limitations too. The formalism does not express the hierarchical structure of the protocols used in the networks, and does not clarify the effects of the adjacent levels i.e. the interfaces between the levels.

2.4. Graph module approach

The UCLA-Graph Model of Computation (developed by Estrin [13] and improved by Gostelow [14] and by Cerf [15]) is the graph modules modelling technique worked out by Postel [16]. He points out that the communication protocol can be successfully analysed by graph models, but the analysis leads very quickly to very complicated graphs. Extending the power of the graph model with a useful definition of the graph module concept a methodology can be given for the design of communication protocols which ensures that the resulting protocol is "well behaved": i.e. it has no deadlocks. The concept of reducing the complexity of the graph model by modularizing is useful from two points of view: (1) the computational complexity of the graph analyses grows combinatorially with the size of the graph and for the sake of easier handling it is necessary to break them into smaller pieces; (2) in a complicated graph there is the possibility of finding identical structural element, and if such elements occur, it is sufficient to replace them by the corresponding concise module representation. The result of modularization is a box containing a few concise transformation expressions describing the function of the subgraph in terms of its input to output relationships. Two types of graph modules are defined: (1) proper and completely reducible modules and (2) constructed modules. These modules substantially reduce the complexity of graphs of communication protocols.

This graph modelling technique allows designers to check every step of their protocol designs. The use of this technique to investigate a protocol may result in errors (1) in deriving the transformation expressions from the graph, (2) in the mapping from the protocol to the graph, (3) in interpreting the protocol, and (4) in the protocol itself.

No information in the related work is given about the applicability of this method to all aspects of protocols (hierarchy of the protocol system, effects of the adjacent levels, etc).

2.5. Time Petri-net (TPN) approach

Based on works which studied the recoverability of computing systems [18, 19] Merlin [17] presented a methodology for the design and implementation of communication protocols. In Ref.15 Gostelow pointed out that the UCLA Graph Model of Computation and the Petri-nets are equivalent. This means that the same approach can be applied using Petri-nets and graph models, or any other equivalent model of computation which is able to represent a concurrent environment (e.g. a computer network). The complexity of such a model can be reduced by dividing it into modules, and so a structured, hierarchical, "top-down", modular design can be provided and relatively easily described by Petri-nets by other equivalent tools [20]. If the process is represented by Petri-nets there will be a lack of knowledge about the execution time of its parts, in certain cases (e.g. in the communication protocols) however, it is important to have a way of representing the time spent by the events, or the relationship between the different types of time-periods.

The time Petri-net (TPN) introduced by Merlin is convenient as a means of considering the effects of different time-pe-

riods. The original definition of Petri-nets is extendended by additional elements. Each bar has two elements specifying the elapsed time: the first of them denotes the minimal time which needs to elapse from the time when all input conditions of a bar are evabled until this bar can be fired; the second denotes the maximum time which can elapse from that moment when all input conditions of a bar are enabled and the bar does not fire, but after this time the bar must fire. This extension acts as a means of changing the original nondeterministic Petri-net into a deterministic one.

This approach takes into account the problems encountered during the design of a process and a communication protocol system; among these problems are deadlocks, incomplete or inconsistent specifications, etc. The approach does not differentiate between the hardware and software parts of the processes. The Petri-net, however, is not always the best design tool; the representation of data and other structures modelling large systems creates difficulties. This technique has all the limitations that the Petri-nets have.

2.6. Variable structure sequential machine (VSM) approach

The variable structure sequential machine is a tool for describing protocols. This tool was elaborated by Mezzalira and Schreiber [21] on the basis of Gill's work [22]. The VSM can be considered as a Mealy automaton whose transitions depend on secondary inputs too. The primary input symbols are the symbols of the input alphabet (as in the usual sequential machines) which influence the state transition and cause the emission of an output symbol. The secondary input symbols have no direct influence on either the state or the output emission, they are considered as parameters during these actions. The symbols of the primary input alphabet can be regarded as a set of messages, and the symbols of the secondary input symbols as a set of commands that guide

the behaviour of the machine. The VSM can be represented by a state transition (or s-) graph or a message (or m-) graph, which is a directed multigraph with labelled and coloured nodes and labelled edges. The nodes of the first colour are labelled with the symbols of the primary input alpabet (i.e. input messages) and the nodes of the second colour are labelled with the symbols of the primary output alphabet (i.e. output messages). The edges are labelled with an element of the secondary input alphabet (i.e. a command), and two elements of the set of states (old state and new state).

The VSM itself is not a model of the protocol system but on the basis of the VSM a formal description can be given. The syntactical rules of the interaction of two VSM-s – connected with each other in such a way that the output of the first is the same as the input of the second – will be reflected as the formal representation of the protocol. The most convenient tool for demonstrating the interaction seems to be the m-graph.

The VSM concept allows one to describe the interaction formally thereby elucidating the rules of this interaction, but from the other details of this technique there is no information.

2.7. Interlocutor approach

The interlocutor approach was developed by Danthine and Bremer [23, 24] after some extentions and improvements of the theory of colloquies proposed by Le Moli [25]. The interlocutor is a mechine which contains 3 types of inputs, 3 types of outputs and a set of internal states. The three types of inputs are: (1) messages coming from the other interlocutor, (2) commands coming from the user of the interlocutor, (3) texts coming from the user and to be trans-

mitted to another user. The three types of outputs are: (1) messages for the other interlocutor, (2) commands for the user of the interlocutor, (3) texts for the user coming from another user. The set of internal states are defined by the set of two state vectors. The internal structure of an interlocutor contains a processing unit with two registers according to the two state vectors. The protocol representation is based on the idea that any interaction between two processes can be modelled as a colloquy between two interlocutors. The two interlocutors are linked such a way, that the output of one is the input of the other. Each interlocutor receives and sends both commands and text. An axiomatic description of a colloquy can be given by defining the behaviour of the processing unit of the interlocutor when it receives an input command or an input message. Each group of relations of the axiomatic description defines a Mealy automaton. These automata are, however, really separate ones because the set of the first state vectors is influenced only by the set of input commands and the set of the other state vectors is influenced only by the set of input messages. In order to describe a protocol it is necessary to define (1) the set of the input and output commands and the set of the first state vectors, (2) the set of the input and output messages and the set of the other state vectors, (3) the operations important from the viewpoint of command and message generation.

For the sake of formal representation of a type of host-to-host protocols there is a need to extend the model develepod earlier to take into consideration the effect of the multiplexing-demultiplexing, the context and the internal mechanism.

The approach is suitable for the description of the system hierarchy of protocols as well as the protocol semantics.

2.8. Formal grammar approach II

Another formal grammar approach different from that is Ref.7 was developed by the author [26, 27]. The idea of the present approach is the following. Let us consider the protocol as a set of rules which influences the process in a remote computer during the interaction. It is found that the "dialogue" consists of sentences communicated by the participants of the inferaction. The sequence of sentences shows an orderliness so it can be supposed that it might be generated by a grammar. If we can find the sets of symbols from which the participants compose their sentences, and we can constructed the grammar containing the rules correctly generating these sentences, we can get the formal description of the protocol system. But for the sake of design and implementation we have to decompose the grammar described earlier and have to construct some other grammars too according to the lower layers of abstraction produced by the decomposition. Having constructed them we must define the interface between the adjacent layers. The steps for constructing such a grammar are:

(1) To construct the grammar describing the dialogue:
   a) To determine the set of symbols sent by the first partner to the second, and to determine the set of symbols sent by the second to the first. The union of these two sets forms the set of symbols flowing along the transmission lines, i.e. the set of terminals.
   b) To provide the set of nonterminals.
   c) To construct the set of production rules following the sequence of ideas of the text of protocol recommendation or the sequence of other individual ideas.
(2) To decompose the process of "dialogue" into separate layers of operation.
(3) To construct the grammars of the decomposed lower lay-

ers of the protocol system in the same way as was shown in the step (1).

(4) To define the interface between the adjacent layers.

(5) To construct a system grammar which reflects all the decomposed layers and the interfaces too.

To define the interfaces between the adjacent layers it is necessary to demonstrate the technique used. The idea of the definition of the interface is the following. When a production sequence is completed at the lower layer, i.e. during a whole production sequence only one production rule is completed at the upper layer. This means that some transformation rules are necessary in connection with each production rule of the grammar at a layer from the left-hand side nonterminals to the sentence symbols of the grammars at the lower layer, and some other transformation rules are required from the nonterminals of the last production of the production sequence of a grammar at the lower layer to the right-hand side nonterminals of the production rules of the grammars at the upper layer.

The existence of a system grammar allows the design and implementation of a protocol system with a hierarchical way from separate modules, and the relationship between them are elucidated. This approach is convenient mainly for syntactical (structural) definition of protocols. Further research is necessary to examine the applicability of the approach for describing other protocol levels as well as for the describing the hierarchy of the protocol system.

3. COMPARISONS

Some significant feature of the approaches presented in the previous section are collected in a table, thereby enabling a number of comparisons to be made.

		Model of Ref. 2	Model of Ref. 7	Model of Ref.12	Model of Ref.16
Object of modelling		a,b	a	a	a,b
Means of modelling		B,(A)	A	B	C
Level of the protocol model in the hierarchy		α	α	β	α,γ
Parameters taken into consideration by the models	time-out	x^1	x^1	-	+
	unknown character/ frame/message	x^1	x^2	x^3	+
	sequence numbering	+	-	-	+
	request for retransmission if there is sequence numbering	+	-	-	+
	half-duplex transmission mode	+	+	+	+
	full-duplex transmission mode	x^4	x^4	-	-
	serial operation with hand-shaking	+	+	+	+
	parallel operation	-	-	-	-

Legends:

a syntax of the protocol

b sematics of the protocol

c recoverability of the protocol

α data link level

β end-to-end level

γ user level

+ the parameter is taken into consideration by the model

- the parameter is not taken into consideration by the model

() indirect way

(Continued on the next page

(Cont'd)

		Model of Ref.17	Model of Ref.21	Model of Ref.23	Model of Ref.27
Object of modelling		b,c	a	a,b	a
Means of modelling		D	F	E	A,(B)
Level of the protocol model in the hierarchy		β	α	β	α
Parameters taken into consideration by the models	time-out	+	-	+	+
	unknown character/ frame/message	+	-	+	+
	sequence numbering	+	-	+	+
	request for retransmission if there is sequence numbering	x[3]	-	+	+
	half-duplex transmission mode	+	+	+	+
	full-duplex transmission mode	-	-	-	+
	serial operation with hand-shaking	+	+	-	+
	parallel operation	-	-	+	+

Legends: (Cont'd)

A formal grammar

B automata

C graph module

D time Petri-net

E interlocutor

F V.S.M.

x^1 the parameter is in the definition but its effect is not shown

x^2 the parameter is taken into consideration but its effect is not clear because of inner loops

x^3 the parameter is taken into consideration in an implicit way

x^4 the parameter is taken into consideration but with assumptions

4. SUMMARY

Some approaches to describing formally the protocols of data communication or computer networks were dealt with. These approaches represent the first phase in a logical designing or implementing process. The paper discussed the basic idea of the approaches, the formal tool, and the method of description. The suitability and the significance of the approaches for particular kinds of protocols and protocol levels were examined and a brief comparison of eight techniques – one of which was elaborated by the author – is made.

REFERENCES

1. G.B. Gerace, Digital System Design Automation – A Method for Designing a Digital System as a Sequential Network System. IEEE Trans. on Comp. vol. 17, no. 11, p. 1044-1061, 1968.
2. D. Bjorner, Finite State Automation – Definition of Data Communication Line Control Procedures. AFIPS Conference Proceedings, vol. 37, 1970.
3. D.W. Davies and D.L.A. Barber, Communication Networks for Computers. John Wiley and Sons, London, 1973.
4. B. Randell and F.W. Zurcher, Iterative Multilevel Modelling, a Methodology for Computer Sytems Design. IFIP Congress 68, Edinburgh, 1968.
5. E.W. Dijkstra, The Structure of T.H.E. Multiprogramming System. Commun. ACM, vol. 11, no. 5, p. 341-346, 1968.
6. General Information, Binary Synchronous Communication. IBM Systems Reference Library, Form No. A27-3004.
7. H.J. Hoffmann, On Linguistic Aspects of Communication Line Control Procedures. IBM Report RZ 345, 1970.
8. S. Ginsburg, The Mathematical Theory of Context-free Languages, McGraw - Hill, New York, 1966.

9. W.C. Lynch, Reliable Full-Duplex File Transmission over Half-Duplex Telephon Lines. Commun. ACM, vol. 11, no. 6, p. 407-410, 1968.
10. K.A. Bartlett, R.A. Scantlebury and P.T. Wilkinson, A Note on Reliable Full-Duplex Transmission over Half-Duplex Link. Commun. ACM, vol. 12, no. 5, p. 260-261, 1969.
11. W.C. Lynch, Commentary on the Foregoing Note. Commun. ACM, vol. 12, no. 5, p. 261, 1969.
12. R.E. Rusbridge and A. Langsford, Formal Representation of Protocols for Computer Networks. Harwell Report, AERE-R 7826, 1974.
13. G. Estrin and R. Turn, Automatic Assignment of Computation in a Variable Structure Computer System. IEEE Trans. on Computers, vol. EC-12, p. 756-773, 1963.
14. K.P. Gostelow, Flow of Control, Resource Allocation and the Proper Termination of Programs. Ph. D. Dissertation, UCLA - ENG - 7179, 1971.
15. V.G. Cerf, Multiprocessors, Semaphores and a Graph Model of Computation, Ph. D. Dissertation, UCLA - ENG - 7223, 1972.
16. J.B. Postel, A Graph Model Analysis of Computer Communications Protocol. Ph. D. Dissertation, UCLA - ENG - 7410, 1974.
17. P.M. Merlin, A Methodology for the Design and Implementation of Communication Protocols. IEEE Trans. on Communications, vol. COM-24, no. 6, 1976.
18. P.M. Merlin, A Study of the Recoverability of Computing Systems. Ph. D. Dissertation, Univ. of California, Irvine, 1974.
19. P.M. Merlin and D.J. Farber, Recoverability of Communication Protocols – Implications of a Theoretical Study. IEEE Trans. on Communications, vol. COM-24, no. 9. p. 1036-1042, 1976.
20. P.M. Merlin and D.J. Farber, A Note on Recoverability of Modular Systems. AFIPS Conference Proceedings, NCC 1975, p. 695-699, 1975.

21. L. Mezzalira and F.A. Schreiber, Designing Colloquies. First European Workshop on Computer Networks, Arles, 1973.
22. A. Gill, Time Varying Sequential Machines, Journal of the Franklin Institute, no. 276, p. 516-539, 1963.
23. A.A.S. Danthine and J. Bremer, Communication Protocols in a Network Context. ACM SIGCOMM - SIGOPS Interface Workshop on Interprocess Communications, Santa Monica California, 1975.
24. A.A.S. Danthine and J. Bremer, An Axiomatic Description of the Transport Protocol of CYCLADES. Professional Conference on Computer Networks and Teleprocessing, Aachen, 1976.
25. G. Le Moli, A Theory of Colloquies, First European Workshop on Computer Networks, Arles, 1973.
26. J. Harangozó, A Method to Describe the Data Link Level Protocol of Computer Networks with a Formal Language. Second Hungarian Computer Science Conference, Budapest, 1977.
27. J. Harangozó, An Approach to Describing a Data Link Level Protocol with a Formal Language. Fifth Data Communication Symposium, Snowbird, Utah, 1977.

THE MEASUREMENT OF COMPUTER NETWORKS

DR. K. TARNAY
CENTRAL RESEARCH INSTITUTE
FOR PHYSICS, BUDAPEST

ABSTRACT

A survey is given about the monitor systems of computer networks. Three levels of the measurement: diagnostic, performance and analytic methods are summarized.

1. INTRODUCTION

The measurement of computer networks can be derived from two rather different fields of measurement techniques:

- the measurement of telecommunication
- the measurement of computer systems.

The measurement of telecommunication has a past of several decades. Nearly all steps of the measurement are determined by wellconsidered recommendations of CCITT and the theoretical fundamentals are clear and obvious.

At first, the measurement of computers was a very simple task and only the development and the complexity of the computers necessitated the production of special measuring sets and the elaboration of suitable measurement methods. The theoretical fundamentals of "compumetrics" are in the process of developing yet and the different "task-oriented" methods of the individual computer manufacturer firms sometimes lead to erroneous conclusions. It is also a fact, however, that computer industry is one of the most dynamically developing ones and this dynamical development moves powerful financial and intellectual capitals.

It is necessary to emphasize in the same way, as no up-to-date packet switching computer network can be produced by simply connecting a telecommunication net and several computers, similarly no up-to-date measurement of computer networks will be identical to the joint application of telecommunication measuring sets and computer monitors and to the joint adoptation of measurement methods of both fields.

The first measurement results took their origin in the same time with the appearance of the first computer networks, and the networks them selves were modified on the basis of the

evaluation of these measurements.

Why is it necessary to measure computer networks? The main reasins for this can be arranged in three groups:

a/ diagnostic inquiery
- the control of correct operation
- the exact detection of errors
- the diagnostics of the reason of errors

b/ recognition of performance
- the examination of performance
- the control of the network traffic
- the determination of the utilization of the resources

c/ analysis of relationships
- the analysis of interdependence of different software and hardware elements
- assurance of corresponding statistics for model parameters
- the analysis of operation concerning the essence of computer networks.

2. MONITORS, MONITOR SYSTEMS

The components of measurement systems of the computer networks are the software, hardware and hybrid monitors used for the measurement of computers. Since the measurement of a computer network is rendered more difficult by the fact that, as a rule, the nodes are geographically widely dispersed, it is necessary to distribute the activity of the monitor system along the whole network. One part of the tasks is concentrated in a measurement center (control and coordination of the measurements, analysis of the results) while the other parts of tasks (collection of data) are distributed along the nodes.

The same is characteristic of the measurement of ARPANET,too.

All IMP-s perform a measurement task, but some host machines composing the Network Measurement Center and the Network Control Center, also play an important part in the control and the evaluation of the measurement.

The geographical and functional distribution of the tasks is shown in Figure 1.

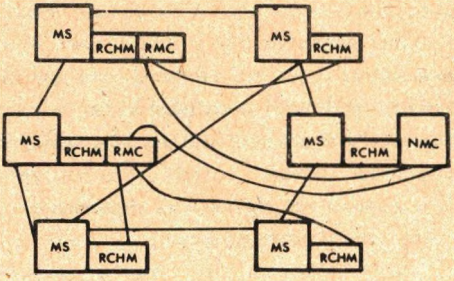


Fig.1. Network measuring monitor system

The hybrid monitor system [3] has a Measurement Software (MS) distributed along the nodes, and connected directly to the Remote Controlled Hybrid Monitor (RCHM). The Network Measurement Center (NMC) controls these Hybrid Monitors through the Regional Measurement Center (RMC). An up-to-date test facility has be developed by Bell Company for the analysis of store-and-forward message-switching systems [1]. The principle was: "using the system to test the system" namely, the test facility is an existing sys-

tem configuration: a modified version of the BISCOM (Business Information Systems Communication System).

BISCOM is a large-scale, computer-based, store-and-forward message-switcher, the test set based on BISCOM contains hardware and software monitors, terminal and network simulators, as well as data-reduction and analysis packages. The hardware monitors collect the data on the activity of the hardware components. Their output data are stored on magnetic tape and later these are used as input data of data reduction routines. Software monitors log in statistical reports in a terminal every three minute and collect the significant events on a magnetic tape. BISCOM applies a link control protocol in accordance with ANSI X.3.28.

3. WORKLOAD

The analysis of a system in only under given conditions an unambiguous task, therefore a system has to be analysed under a determined workload [4].

Benchmark mixes, traces and synthetic jobs used for measuring computer systems and the traffic samples suggested by CCITT for testing communication lines can form the base of the workload of computer networks, after due consideration [6].

The workload of computer networks is produced by artificial traffic generators. The changeable parameters for these generators are e.g. the proportion of long and short messages [2], the number of messages per time unit the proportion of overhead to real information.

I should mention the test tape library of BISCOM, the tapes of which represent different workloads, message mixes and test durations. The creation of a test-message tape can be

seen in Figure 2. The selection of message workload and message mixes was performed after the forecasting of Bell Laboratory.

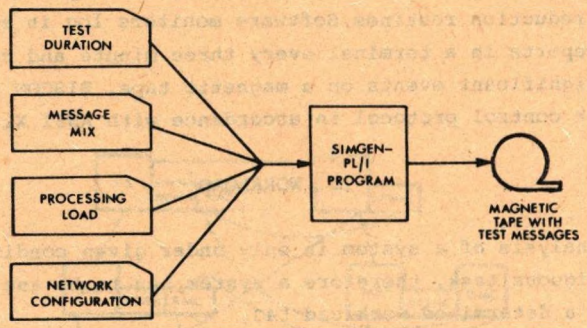


Fig.2. Creation of a test-message tape

4. THE LEVEL OF MEASUREMENTS

A computer network is characterized by its throughput, delay cost and reliability. To determine these is the direct or indirect aimof measurements. The measurements can be classified according to several points of view, e.g. which parameter is measured, whether the parameters are deterministic or stochastic, which type of monitor is used, whether the user or the Post Administration has a priority.

The comprehensive analysis of the measurement is made easier by considering its hierarchy. The diagnostic test controlling the correct operation of the system represent the bottom

level while the performance measurement means the middle level and the analytic measurement constitutes the topmost level. Let us follow this logical structure.

5. DIAGNOSTIC MEASUREMENTS

The purpose of diagnostic measurements is the control of correct operation of computer networks, the detection and diagnostics of errors. The operation of a computer network is characterized by numerous events. An event e_k can be regarded as a logical function defined on subset E_k, where $E_k \subset X$. X means the set of possible states of a computer network. An event e_k starts when the network steps from state x_{old} to state x_{new}, supposing that $x_{old} \notin E_k$ and $x_{new} \in E_k$. An event e_k terminates in the opposite case (Fig.3.).

The essence of adiagnostic measurement means observation of the logical functions belonging to the events.

The observed events can be classified in two groups:

- current network configuration,
- current operational status.

The current network configuration gives in up-to-the-minute report on the living links, node and host machines.

The current operational status represents the state of network elements, the errors and their causes.

The measurement components are carried by additional parts of standard packets or by special packets. The additional diagnostics of the standard packets can be represented by the acknowledgement of the header of link control protocols and the error control in its trailer. Special packets inform the source if the packet hasn't arrived to its destination

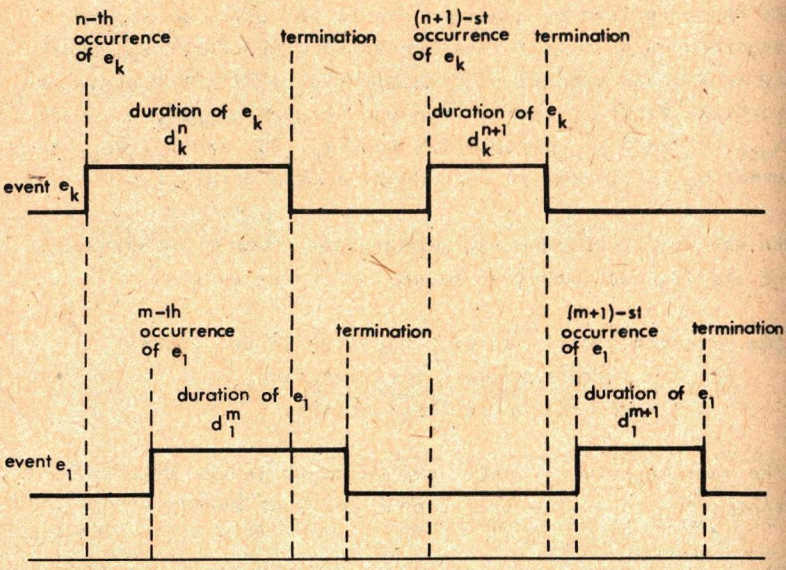


Fig.3. The definition of events.

(Non-delivery Diagnostics) while packets tell the users the network time (TIME). The special packets indicate the information packets crossing the nodes (TRACE). These special packets are different network by networks.

A rather simplified model of computer communication is considered at the diagnostic measurements. The measurements are time or event controlled. It is difficult to evaluate the extraordinary large number of the data. The results can be surveyed by a hierarchical groupping. E.g. the automatic monitor system of the international network of General Electric [5] makes available the results in three levels:

- the state of all nodes hosts and links is accessable from many nodes
- the code numbers of the elements near the overload or showing increasing error rate are depicted in displays in some nodes
- the list of erronous components can be read on a single monitor by the central operator.

6. PERFORMANCE MEASUREMENTS

The performance of a computer network means the quality and quantity of services by given workload. The purpose of performance measurements is the analysis of effective operation under dynamically changing conditions. The measures of performance are the functions of events e_k defined in Chapter 5. The countable measures of performance give the occurrence of events, the timeable measures the duration of events. The measures of performance are generally conditional probabilities interpreted in a set of workload characteristics [4].

The general form p_k of performance can be described as an integral:

$$p_k(t) = \frac{1}{t-to} \int_{to}^{t} f_k(t)\, dt$$

where $f_k(t)$ is the density function belonging to event e_k.

The average value of timeable performance measures is

$$p_k^T(t) = \frac{1}{t-to} \int_{to}^{t} e_k(t)\, dt \qquad t \geq to$$

where $e_k = 1$ if $x \in E_k$
$e_k = 0$ otherwise

The average value of countable performance measures is

$$p_k^C(t) = \frac{1}{t-to} \int_{to}^{t} i_k(t_n - t)\, dt \qquad t \geq to$$

where $i_k = 0$ if $t \neq t_n$
undefined for $t = t_n$

and t_n is the time for which $x(t_n)_- \notin E_k$ and $x(t_n)_+ \in E_k$

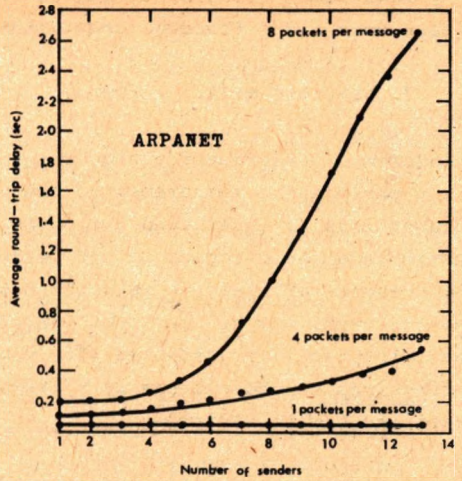
i_k is the impulse function, thus $\int_{to}^{t} i_k(t_n - t)\, dt = \sum_n 1$

Two tipical examples can be seen in Figure 4. Figure 4.a shows the average delay of ARPANET, the Figure 4.b the average delay of ALOHA as a function of the user's number [2].

It is worthwhile to mention that CCITT is working on standardization of probabilities characrerizing the performance of data networks:

- grade of service (GOS)
- quality of service (QOS)

The computer network models used by performance measurements are essentially more complex than models by diagnostics. The right selection of workload parameters and the correction of



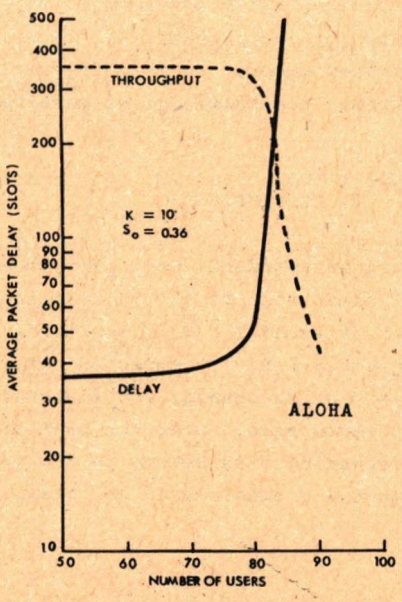


Fig.4. Performance measurement of ARPANET and ALOHA

errors caused by measurement loss according to the traffic necessitate a hard work and a thorough consideration.

7. ANALYTIC MEASUREMENTS

The analytic measurements represent a higher level as compared with the performance measurements. The goal of analytic measurements is to understand the processes in computer communication and their impact on the performance of the system. The measures of analytic measurements are determined in the most general way by the joint density function of the performance probability variables. The number of joint density functions can be reduces by filtering the independent probability vector variables.

The determination of the average line effectivity of ARPANET is a good example for analytic measurements [2]. The analytic models are based on the probability theory, decision making theory and system theory. It is difficult to imagine a right analysis without the knowledge of measurement theory.

8. CONCLUSIONS

Many computer network measurements were performed, some of them have been published. Most of the publications deals with the measurement of ARPANET. Since the different network measurements have given a lot of unexpected results, further development of the measuring methods and elaboration of the theory have a great importance. Among the problems to be solved the most interesting ones belong to the interdisciplinary area of computer communication and measurement technique.

REFERENCES

[1] W.E. Bracker, E.R. Sears: Test facility for a Message- Switching System
The Bell System Technical Journal
1976 Sept No.7 Vol.55 pp. 857-873

[2] L.Kleinrock: Queueing systems Vol.II.
John Wiley & Sons, 1976.

[3] D.E. Morgan, W. Banks, D.P. Goodspeed, R. Kolanko:
A computer network monitoring system
IEEE Trans. on Software Engineering
Vol. SE-1 No.3 1975. pp. 299-311

[4] L. Svobodova: Computer performance measurement and evaluation methods
AD-A013 318 1974.

[5] G.H. Wedberg, L.W. Hauschild: The General Electric network monitor system
Information Processing 74. pp.24-28

[6] CCITT Green book Vol.VIII.
Data transmission
International Telecommunication Union, 1973.

A LOCAL NETWORK FOR THE SUPPORT OF SOFTWARE DEVELOPMENT

A.Arató - I.Sarkadi-Nagy - F.Telbisz
Central Research Institute for Physics, Budapest

ABSTRACT

In order to improve the efficiency of software production a network is under development in the Central Research Institute for Physics. The network will contain ES-1040 as host computer, TPA-70 as front and processor and mini or microcomputers as well as alphanumeric display units as terminals.

The system will have two main services: a Text Editor and a File Manager. The Text Editor provides three kinds of services: a/ interactive text editing, b/ job submission to the batch processing, c/ the results of the batch processing can be called down and examined through the Editor.

The File Manager provides an access for the programs running in the small computers to the disks of the large central computer. The small computers can treat these files as "local files", they can read, write or update them.

The planning considerations and the system architecture are discussed as well as the user level communication protocols.

1. INTRODUCTION

In a research institute - like the Central Research Institute for Physics - a considerable amount of work is spent for software development. In our institute there are three main fields of development:

- Large batch programs either for theoretical computations or for measurement evaluations;
- Real time systems both for process control and for collecting measurement data;
- Development of basic software for small computers and microprocessors.

Practically in such an environment a much larger percentage of computer time is used for software development /program modifications and debugging/ than for production runs. Therefore it is very necessary to produce tools which make these efforts as effective as possible.

In analysing the needs and working habits, we have found two bottlenecks.

Text editing is one of them. Many program modifications are necessary both for the large and for the small computers and preferably they should be done interactively [1]. Unfortunately, interactive editing is very time consuming on the computer, if no time-sharing can be achieved.

The other bottleneck is characteristic for the development of real time systems based on small computers or microprocessors. A configuration proper for a real-time application is not suitable for software development. Generally there is no high speed printer and probably no disk is at disposal in

the configuration although both facilities are very useful, almost indispensable for the development work.

On the basis of these considerations we have decided to build up a system to improve the efficiency of our work. This system is called OREMUS, which is an acronym for Online Remote Editor and Minicomputer User's Support. This system is a hierarchical network, in the sense as it is explained by R.L.Ashenhurst [2]. This system has two main services:

- Text Editor,
- File Manager service.

In this paper this two services will be described as well as the general architecture of the system.

2. THE TEXT EDITOR

In planning this editor we had in mind two very successful editors: the system WYLBUR at Stanford University [3] and system ORION at CERN [4], and in reality from the user's point-of-view our editor is very similar to these. We considered very important both the human engineering aspects and the reliability. This is reflected in the following facts:

- It is not a sequential editor, but during editing a random access is possible to the file. Like with the above mentioned two editors, in the CRJE of IBM, or in the very popular languages of BASIC or FOCAL, to each line a statement number is attached and reference to the lines can be made through these numbers.
- Completely free format of commands using meaningful words but allowing a possibility of almost completely arbitrary abbreviations. So the system is rather permissive and also the user can choose the level of verbosity he likes.

- There are two kinds of files: permanent files and work files, and all editing is done only on workfiles. So the permanent files are better protected against system crashes as well as against inadvertent changes by the user. After editing the workfiles can be saved as permanent files.

The Editor program provides three kind of services:

- text editing
- jobs can be submitted to the batch processing
- the results of the batch processing /e.g. lists/ can be called down and examined through the Editor.

2.1. Editor Commands

A brief summary of the editor commands is as follows:

2.1.1. File moving commands

USE - copying a permanent file /partially or entirely/ to the empty workfile

JOIN - copying a permanent file /partially or entirely/ to a non empty workfile

SAVE - save the workfile as permanent file

SCRATCH - scratching a permanent file

CLEAR clear the workfile

PRINT, PUNCH } printing or punching the workfile

2.1.2. Editing commands

Applicable only for the workfile

ADD - add new lines

DELETE - delete lines

REPLACE - replace lines

COPY, MOVE - move lines within the workfile

LIST - list lines on the display

CHANGE - change a character string in one or more lines

SEARCH - list on the terminal the lines, containing a given character string

EDIT - Editing within line

RENUMBER - renumbering the lines

2.1.3. Commands for remote job entry

SUBMIT - job submitting

FETCH - fetch the batch processing results

STATUS - job status enquiry

PURGE - deleting a system output file

2.1.4. Manageing commands

LOGIN, LOGOUT

HELP - asking for assistance of the system; how to use it?

TALK - message to other terminal or the operator of the large computer

EXIT - disconnecting the terminal from the editor

3. THE FILE MANAGER PROGRAM

The aim of the file manager is to provide an access for the users of the small computers to the disks of the large central computers. The file manager is a trusted task in the big computer administering the file handling for the tasks running in the remote small computers. Small computers see these

files like "local files". /Fig. 3.1./

Some restrictions had to be done:
- only disk or printer files can be used;
- only sequential or partitioned files can be used;
- the length of the records is limited /1 Kbyte/.

The File Manager has the following commands:

MOPEN opening a file. The file identifier and the access method is a parameter of the command. A logical file number is given back by the command. All further references can be done only by using this logical number.

MCLOSE closing a file

MREAD, MWRITE } Reading or writing the following record of the file

Three further commands can be used for partitioned files:

MADD adding a new member to a partitioned file

MFIND logically positioning the file to the beginning of a member

MDELETE Deleting a member of a partitioned file.

The access to the File Manager is done by a number of routines implemented in the small computers, and activating the link driver, described in chapter 5.

Small computer packages like PIP /Peripheral Interchange Package/ Loader etc. in the small computer can use these services.

4. SYSTEM ARCHITECTURE

4.1. Hardware

The central computer of the system will be an ES-1040 computer with the following configuration:

Memory	1 Mbytes
Disks	
30 Mbyte drives	8 units
7.25 Mbyte drives	12 units
Magnetic tape drives /800 bpi/	8 units
Card Readers	3 units
Line Printers	3 units
Card Punch	1 unit
Paper Tape installation.	

The network is connected to the host computer by a TPA-70 as front-end processor. The front-end processor will have a memory of 24 K words /16 bits each/, a cartridge disk, a matrix printer and paper tape I/O units. At the lowest level of the hierarchy we can find display terminals, small computers or microprocessors. As display terminals VT-340 units /VIDEOTON/ will be used. As small computers a whole range of processors will be connected: TPA/i, PDP 11/40, SM-4, Microprocessors /Intel 8080/ etc.

4.2. Central Computer

At the highest level is the ES-1040 computer. It runs under operating system OS. Most part of the central processor time will be used for ordinary batch processing. From our point-of-view only four tasks are interesting:

- Editor task
- File manager task
- System reader task
- Output writer task

The logical connections of these tasks can be seen in Fig. 4.1.

4.3. Front-end-processor

The front-end-processor will be dealt with in a separate paper. Only one point should be mentioned here: the Editor is

implemented in two computers, each one playing the role most suitable to it.

The large computer is doing file handling and changes at record level. It handles the requests sequentially, the requests are queued by the front-end processor. This means a minimal load to the central processor and fits well to the essentially batch operating system.

On the other hand the front-end-processor is doing
- the interactive, simultanous service of many users,
- editing within records at character level,
- search for character strings or changing character strings within records.

4.4. User Level

On the user level only the Editor service can be used from the display terminals. Evidently, the File Manager cannot be used. Small computers can have 3 different roles:

1. Real Time systems
These can be either systems under development, or data acquisition systems. The host computer can assist them in the following tasks:
- Text Editing
- Backing store: either programs or data can be stored on the disks of the big computer.
- Cross compilers. The result of the compilation can be directly stored automatically on disks by the big computer, and it can be accessed and called through the File Manager.

2. Software development bases
These are relatively large configurations. This is the only difference with respect to the above mentioned real time configurations, but due to this fact they can be conveniently used for software development.

Nevertheless it is worthwhile to connect even these configurations to the network as in this way the programs developed on these systems can be easily accessed from the small or not yet reliable configurations under development.

3. Display concetrators

If several display terminals are relatively near to each other but remote to the central computer, they can be connected more economically through a concentrator computer. This is a small, dedicated configuration, where no local development is permissible, otherwise the bugs of the software or hardware to be developed would always mess up the normal functioning of the computer.

5. COMPUTER COMMUNICATION

In our system there are two sorts of computer communication. One of these is the "host-computer - front-end" connection. This is essentially a channel-to-channel communication, the large computer sees the front-end as several local peripheral units.

The other kind of computer connection is between the front--end processor and the small computers. For the time being no direct connection is planned between the different remote computers. In the remote small computer all connection to the front-end is carried-out through the link-task. /Fig. 5.1./ This task runs under the local real time monitor.

The link-task has four basic services:

CONNECT a connection is built up to a task running in the front-end. This can be either the Editor or the File manager.

SEND a record is sent through the link.

RECEIVE reading a record through the link

DISCONNECT closing a connection.

The routines activating the File Manager are based on these services of the link task. These are short routines, exchanging buffers with the link task. These buffers can contain either data or commands. Other tasks - e.g. a console monitor task - may make connection to the Editor task.

LITERATURE

[1] J.M.Reaser - J.C.Carrow: Interactive programming: Summary of an evaluation.
NTIS AD-A013 553

[2] R.L.Ashenhurst - R.H.Vonderohe: A hierarchical Multiminicomputer system.
Infotech State of Art Conf. London March 8-12, (1976.)

[3] F.Fajman - J.Borgelt: WYLBUR: An Interactive Text Editing and Remote Job Entry System.
Communication ACM 16. 314 (1973 May)

[4] R.D.Russel - P.Sparman - M.Krieger: ORION - The OMEGA remote interactive on-line system.
Proc. International Computing Symposium (1973.) p. 143.

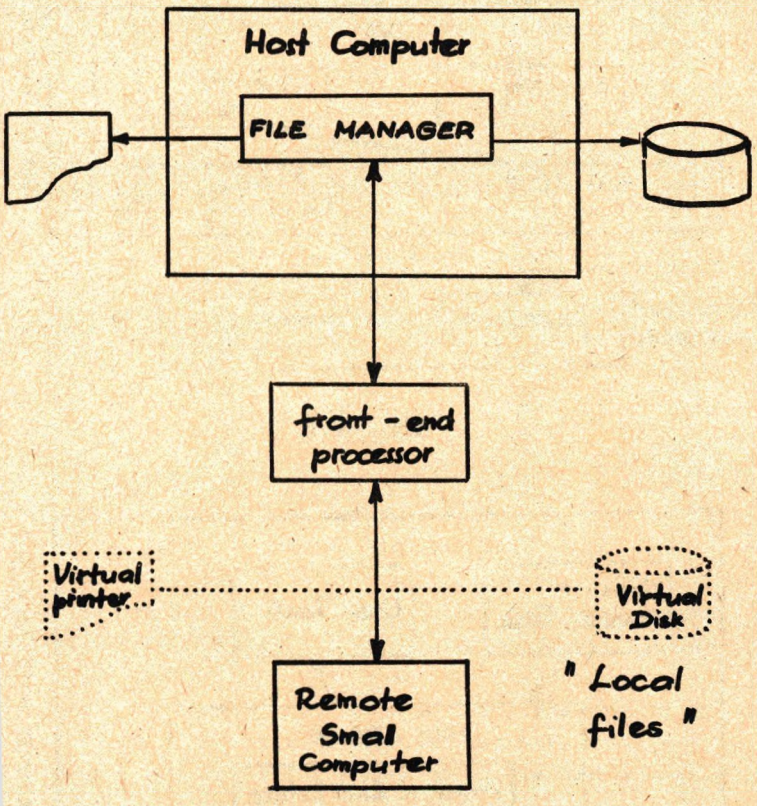

Host Computer
FILE MANAGER
front - end processor
Virtual printer
Virtual Disk
" Local files "
Remote Smal Computer

Fig. 3.1

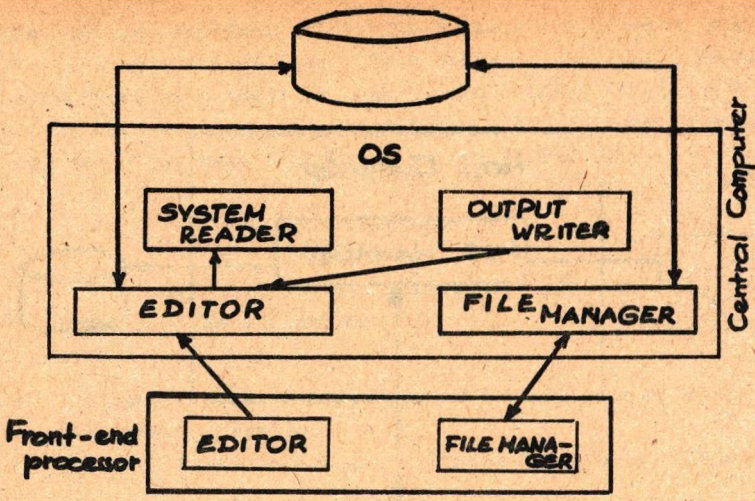

OS
SYSTEM READER
OUTPUT WRITER
EDITOR
FILE MANAGER
Central Computer
Front-end processor
EDITOR
FILE MANA-GER

Fig. 4.1

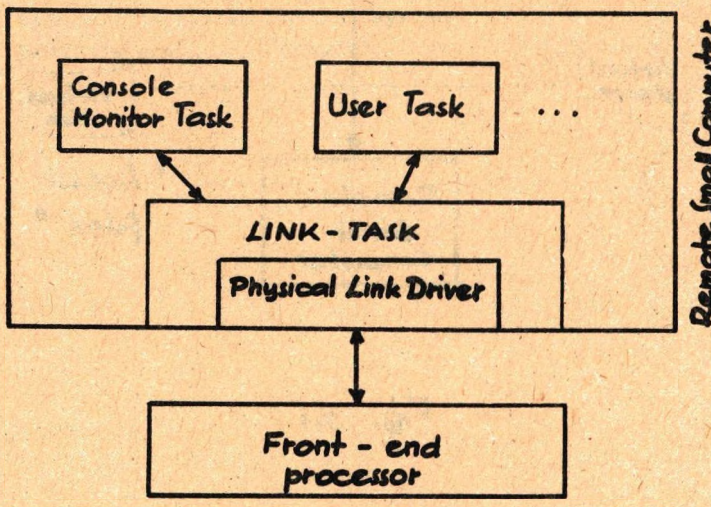

Console Monitor Task
User Task
...
LINK - TASK
Physical Link Driver
Front - end processor
Remote Small Computer

Fig. 5.1

NETWORK AND UNIFORM JOB CONTROL LANGUAGES

Peter Schicker
Ann Duenki

European Informatics Network
Federal Institute of Technology
Zurich, Switzerland

ABSTRACT

The technical problems of providing a common user access mechanism are discussed on the basis of uniformity, distribution, and language considerations. Uniformity is achieved when the user needs to know only one structure of operating systems independent of the individual systems. The goal of distribution is that the user can, from his location, make use of distributed resources that are interconnected through a computer network. The job control language gives the user the tool to handle the resources.

The highly integrated network places demands on a job control language which renders more traditional solutions to a uniform job control language unsuitable. The plausibility of such a highly integrated network is supported with a sketch of a network infrastructure which would be driven by the user's control statements. Finally, administrative implications of making distributed resources available are outlined.

1. INTRODUCTION

A job control language is a tool with which a user directs and controls a sequence of tasks or job steps within a particular environment. The environment is normally a single machine and its operating system; the job control language is the user´s interface to this environment, even more; the job control language gives the user a picture of the operating system.

By a ´uniform´ job control language we mean a job control language which can be used for a multiplicity of environments. The user expresses his needs, e.g. for

- the execution of a compiler
- the access to a file
- a call on operating system procedures

in every environment exactly the same way.

If we add to this concept the ability to direct and control subtasks of one job in more than one environment (i.e. in more than one of a number of interconnected machines), we then have a network uniform job control language.

The user has one interface to a multiplicity of environments which may, depending on the sophistication of the job control language and the structure which supports and executes this control, appear to the user as an agglomerate of mutually co-operating systems or as a single, extended environment. The transition from co-operating systems to a distributed system is not clearly outlined; the more facilities and ease and casualnes of their handling, the more the intelligence seen through the job control language appears as a single system.

In increasing order of sophistication one envisages facilities to:

- submit, execute and retrieve at different sites
- access intermediate files which are resident at sites other than the execution site
- execute sequential job steps on multiple systems
- execute parallel job steps on multiple systems
- access files without knowledge of their physical location
- execute single and parallel job steps without specifying an execution machine

One must decide what control can and should be manipulated through the job control language; and, of this, what shall be seen by the user and manipulated explicitly by him. There are certainly different classes of users and therefore different needs. Design considerations must be carefully reviewed having in mind the sometimes contradictory uses that are intended for a job control language. Design considerations in a network situation are interrelated and I will attempt here to separate them into the areas of uniformity, distribution, and language.

2. CONSIDERATIONS FOR A UNIFORM JOB CONTROL LANGUAGE

Because the job control language - or a compilation/translation thereof - must be interpreted on many different systems, the control which the language affords the user will be based on some set of common primitives, i.e. operations which have a logical equivalent on all the systems involved. With this set, one must decide which details of control are necessary, possible, or desirable.

For example, in many systems the file size must be specified with a request for file storage; in some it need not or even cannot be specified. The user would usually prefer not to think about such things, leaving it in the hands of the system, but some users in some situations would prefer to have the possibility to declare file space requirements ahead of the actual need rather than have, after a long and costly execution, an abnormal job termination due to insufficient file space.

There are innumerable such details which pose problems to the uniformity of a job control language. Below is a list of a few of the more obvious ones:

- On today's operating systems the file system is the most apparently differing entity. Especially, the different naming conventions and protection mechanisms are not easily adaptable to a uniform approach.

- Some of the file systems are device dependent; additional information must be provided to describe the physical separation of the data into records and blocks together with a specification on which device the data is to reside. One could require that this information always be supplied, but this would be unduly burdensome, and sometimes impossible.

- Some of the file systems require an early binding of files to devices or classes of devices; some don't.

- Some of the systems require information, prior to execution, of all resources needed; many systems allocate resources on demand.

- Conventions for calling upon system and application procedures differ. Especially, parameter passing and default conventions have a tendency to be different depending on whether an operating system primitive or an application is brought into execution. The division into the two (or more) classes is arbitrary and different from operating system to operating system.

- Most systems provide defaults for standard file names, for time, money and other resource allocations.

- Time, money, etc. have different 'buying power' on different systems. If there are such discrepencies, clearly it must not be the user's responsibility to cope with them. A uniform unit must be defined which, when employed on any system, has approximately the same meaning.

- Some existing job control language interpreters expect control statements to proceed the data sets on which they operate (i.e. they are dispersed amongst the data), others demand all control statements to be together forming an actual job control program.

To expect solutions to all these problems and achieve a truly uniform job control language is certainly optimistic. Some of the problems might never be solved as they influence the local management of operating systems. For example, a default time limit (or other default limitations for

resources) is very dependent on the total resources available as well as on the community of users of these resources. Defaults, like the one mentioned, might therefore never enter a uniform job control language.

It is certainly as optimistic to envisage only strict implementations of a uniform job control language. There are more than enough examples of standardized languages where nearly every implementation represents a different dialect. The objective for the design of a uniform job control language must be that the job control language is powerful and complete enough so as not to encourage the development of local dialects.

A possible approach to a uniform job control language could be based on the fact that all necessary information is to be found somewhere in the source job control (might be as defaults), or in tables relating to a target system. Most of the problems listed above could then be resolved by a compilor/translator constructing, from the uniform job control language, statements for the job control language interpretor of specific target systems. But this solution does not take into account the demands of a distributed environment.

3. CONSIDERATIONS FOR CONTROLLING A DISTRIBUTED ENVIRONMENT

Here one often distinguishes between the distribution of the files relating to the job execution and the distributed execution of the job itself. The former capability might be based on some special pre-processing and post-processing using the remote job entry or file transfer facilities of the network. The new considerations are:

- Site designation: The location of execution, of files referenced, etc. must be determined. This information might be explicitly stated by the user or a compile-time procedure might perform table look-ups to select an execution site (with the desired services) or to locate a file. If these tables are maintained locally, access is simple but the information may be out of date. Especially, a load sharing environment precludes purely local information and demands network directories which are centrally or distributedly maintained. But even here, the delay between compile-time and run-time may mandate run-time procedures to perform these queries to obtain the necessary information.

- Resource allocation: all resources might be pre-allocated. Here a pre-processor might communicate with appropriate network facilities to move entire files to the site of execution, reserve devices, etc. Such static allocation, even in a large local system, is usually too costly; one cannot, for example, reserve a magnetic tape station until the output of a job which has not even begun execution is ready. But dynamic allocation means not only difficult to detect deadlock situations, but also that the run-time interpreter must make allocation requests.

As soon as we say that the intelligence which interprets the job control language must know about the network we are changing and extending the environment and, implicitly, altering language considerations as well as demands on the supporting structure. The distinction between distributed file capability and distributed processing capability becomes more vague: for example, if the local job control language interpreter knows how to interface to network remote job entry or file transfer facilities and converse

with remote resource allocators to access a file or to dispose output there is very little additional intelligence and capability needed so that we can export job steps and retrieve their results.

In the situation of the truly distributed job, the following additional considerations come into play:

- Operating system variables: Items such as resources (time, money) allocated and consumed, status, etc., are now multiple. These variables must exist for each subsidiary portion of a job. It must be possible to assign values to all of them (individually), to monitor them, and to gather many of them after job terminatiion for statistical purposes.

- Resource allocation: A careful analysis must be conducted to derive what resources can and/or should be named and allocated prior to execution. For other resources an allocation on demand mechanism must be developed.

- Deadlock problems: They are inevitable, especially, where resources are not pre-allocated. The problem is similar to local deadlock problems, except that detection and operator intervention is more difficult. Access to local and/or central allocators is more complex and deadlock prevention algorithms have to take into account their most likely sparse information about distributed network jobs.

- Supervisoy control must exist for the various job tasks, implying status and control links.

- Abnormal situation handling: If a job step (might be one of several parallel job steps) aborts or if the network links connecting to such a job step break, the necessary control for an orderly clean-up must be performed.

- Data conversion: Data, moved from one site to another, can lose the signifigance of its information. Physical representations vary and the problem of logically describing the structure of the data and converting the data arises.

- Library procedures and parameters: If job tasks are to be executed at dynamically selected sites, compilation translation cannot be bound to a target machine. Since job control library procedures and application procedures must be called with the aparently same mechanism one must establish parameter passing conventions with a comprehensive set of default possibilities.

- Hierarchy of libraries: The name space in such a network environment can be extensive; conventions must exist for an orderly search for a named entity. For example:
  - User defined job control procedures
  - Standard job control procedures
  - Job control library procedures
  - Application procedures

Having enumerated a few of the more apparent problems to create a truly distributed system, the question arises: With today's technology, can a distributed environment, which is easily controllable (also by a casual user), be achieved?

4. THE DISTRIBUTED ENVIRONMENT

It is impossible that a fully distributed network environment can be adequately accessed and controlled through the medium of existing job control languages and existing job control interpreters. The network job control language, as seen by the user, must provide a broad possibility of expressions. As seen by the interpreter of the local operating system, it must provide network information and must anticipate a multi-process parallel environment. The interpreters themselves must interface to a supporting structure of network services as well as to the facilities and capabilities of the local operating system.

The requirements of the casual user do not ask for a set of separately controlled co-operating interpreters hosted on different operating systems. In general, the user must be able to disregard the differences and converse with a single (albeit distributed) interpreter. The difference must only become visible when a user with special needs wants to be aware of them and wants to control these differences.

To achieve this, we propose a job control language which is defined for an interpretative system. The job control language, as the user sees it, is compiled into a set of order codes which can be viewed as machine instructions for a hypothetical stack machine. Communication and synchronization between various subsidiary job steps occur through variables in the stack. The order codes reference variables through a two-dimensional address, the first dimension giving the block level, the second dimension the offset of the variable from the beginning of the portion of the stack for the block level indicated.

References to the outmost block imply references to the operating system variables. These variables need not be declared as they are the same for each job.

If the job is executing on a single machine the stack does not differ from the ones of stack oriented programming languages. At creation time of a subsidiary job, however, the stack becomes distributed. Variables local to a subsidiary job (or job step) reside on the portion local to that job step; they are not accessible from the parent job. Variables on the stack of the parent job at block levels below the ones of the subsidiary job are accessible from both the parent and the subsidiary job and can, therefore, be used for inter-job communication and synchronization.

To execute a subsidiary job (i.e. a single or a group of job steps) the interpreter at the site where the subsidiary job is to be executed receives a set of order codes, the current display (pointers into the stack describing the beginning for the various block levels) and initial values for the operating system variables (e.g. resource limitations). For portions of the stack that are not local the display contains references to the interpreters where the corresponding portions of the stack reside.

A reference to a variable in a remote portion of the stack implies access through the network which is performed by the various local incarnations of the distributed interpreter. This communication is the basic vehicle for distributed network jobs; it also has a strong influence on the structure of the job control language and on the presentation of a distributed network system to the user.

5. LANGUAGE

The job control language must allow the user to access and co-ordinate the facilities available in the network in a comfortable way. The structure presented in the previous paragraph is most easily orchestrated by the user with a language which provides:

- Structured statements: The nesting of subsidiary jobs is nicely equated with nested block structures. This nesting gives in addition a natural distinction between global variables and variables local to, for example, a subsidiary job.

- Procedure calls: The interface to job control procedures (i.e. procedures of job control statements) and application procedures (e.g. compilers, packages, etc.) is the same with the same parameter passing and default mechanism. The difference between these two kinds of procedures, therefore, disappear completely for the user.

- Implicit or explicit semaphore operations: Synchronization and access protection between different subsidiary jobs executing in parallel is performed by semaphore operations introduced by the compiler of the job control language or by the user himself.

- Type declarations: Special types are introduced to reflect the objects that are manipulated by the job control:
  - job reference (access path to subsidiary jobs)
  - virtual device reference (access path to files or devices)
  - semaphores
  - resource control types

- Standard procedures for:
  - resource allocation
  - resource monitoring and control
  - management of subsidiary jobs
  - creation and execution of new application procedures (link-edit, load and go, etc.)
  - private library maintenance

  etc.

The resolution of the complexity of such a language as well as the early detection of failures is assisted by a compilation phase. The compiler resolves the user's control into a more basic intermediary language which is commonly interpreted throughout the network.

The introduction of a compilation phase from the job control language to the order codes of the hypothetical distributed stack machine, of course, allows one to think about having more than one job control language. This may be desirable in that the majority of casual users will have simple demands and might find a 'high-level job control language too complicated. A simple language with restricted capabilities might very well cover the needs of this class of users.

Thus, a properly organized supporting structure can become the foundation for network job control languages and allow users of the current systems to become network users having at their control the multiplicity of services available in the distributed system.

6. ADMINISTRATIVE AND CONSULTING PROBLEMS

A uniform access to the many facilities and services of a network does not, of itself, resolve all problems. The job control language does not tell the user what is available nor how to use a service. The remote user has no assistance close at hand nor is he privy to the everyday organizational set-up of the remote system where some services are provided. Indeed, the user may or probably will not know which remote system is involved.

How then does the user obtain assistance? Local consulting will most likely be inadequate for all the possible services on a large network. If changes are introduced in a local service the remote user must be informed.

And how shall the network user, with files at several remote sites, know when he has exceeded some local resource restriction (time, costs, space, etc)? And how will he be billed for the services rendered? Shall we require that each system maintain information about every prospective user, his permissions, passwords, accounts, files, etc.?

The administrative problems of a large network are also large and one will probably turn to the network itself to provide solutions in the form of network-wide services:

- distributed consulting service
- network accounting service
- network file directories

etc.

But as more and more aspects of control and administration become network-wide rather than local considerations, the multiplicity of environments becomes less and less visible.

7. SUMMARY

The problems of developing a uniform network job control language are related to, and vary with:

- the number and variety of hard- and software architectures
- the distributed processing capability desired
- the desire of the user to see or be shielded from the complexity of the environment

A job control language is a balance of these considerations. It may be simple or complex; it may include a compilation phase; it may show the user a network operating system , or leave the various systems visible and distinct.

The language chosen is anyway a refelection of the technical and administrative organization which represents and to which it provides the access. Paradoxically, as the supporting structure becomes more integrated, the more it functions as a whole rather than many parts, and the more one may desire several specialized control languages to a uniform environment, rather than a uniform language to specialized environments.

8. BIBLIOGRAPHY

[1] Schicker, P.,Baechi W., Duenki A., Job Control in a Heterogeneous Computer Network, Proceedings of the conference on Communication Networks, Online 1975 Uxbridge, England

[2] Schicker P., Duenki A., Job Control and its Supporting Services, ICCC-76

[3] Sandmayr H.,Strukturen und Konzepte zur Multiprogrammierung, Ph.D. Thesis 1975, Federal Institute of Technology, Zurich, Switzerland

[4] Brinch Hansen P., Concurrent Pascal. A Programming Language for Operating System Design, Information Science Technical Report No. 10, California Institute of Technology

[5] Habermann A. N., Prevention of System Deadlocks, Comm. ACM, July 1969, New York, USA

[6] Randell B.,Russell L. J., ALGOL 60 Implementation, Academic Press 1964, London, England

[7] Unger C. (editor), Command Languages, North Holland Publishing Company 1975, Amsterdam, The Netherlands

[8] Barbar D. L. A., The European Informatics Network: Achievements and Prospects, ICCC-76

IFIP WG6.1 - INTERNATIONAL NETWORK WORKING GROUP

D L A Barber

Chairman
IFIP WG 6.1
International Network Working Group

ABSTRACT

This note briefly describes the achievements and the aims of the International Network Working Group (INWG). INWG is the title of Working Group 6.1 of Technical Committee 6 of the International Federation for Information Processing, which is fulfilling an important role as a medium for the exchange of information about data communications between research workers and network users from many countries.

1. THE PAST

In the late 1960s a few enthusiasts involved in early Computer Network research projects began to present papers at International Computing Conferences. As the interest in data networks, and in particular in packet switching, grew more widespread, these enthusiasts started to meet informally to exchange ideas and working papers and to compare experiences with their various projects.

During the first International Conference on Computer Communications in Washington in October 1972 some two dozen of these network buffs decided to call themselves the International Network Group (INWG), with the express purpose of promoting closer collaboration to minimise the overlap of research work, and to try to ensure some compatibility between the networks they were building.

At the Washington Meeting Prof Vinton Cerf, then of Stanford University, was asked to serve as Chairman, and the ARPA Network Information Centre agreed to handle the initial circulation of papers. Guided by Cerf, INWG became so popular that, in early 1973, a proposal was made that it should seek affiliation to the International Federation for Information Processing.

Following approval by the IFIPs council of the terms of reference given in the appendix, INWG became the first working group of IFIPs Technical Committee 6 (TC-6 Data Communication), then chaired by its founder chairman, Alex Curran of Bell Northern Research.

A period of steady expansion followed during which several sub-groups were formed. Meetings were held at most of the large conferences and an unexpectedly large number of papers were offered for private circulation by INWG members. As the membership multiplied so did the task of

handling the paperwork, and it was decided to charge a small fee to cover costs. This did not deter new members, and by October 1975 INWG had nearly 100 members, while some 130 papers had been circulated to form an invaluable body of knowledge on the development of ideas, techniques and experiences relating to data communications networks.

After four years of nurturing INWG, Vint Cerf resigned as chairman, and the post passed to Derek Barber, Director of the European Informatics Network. Meanwhile, Louis Pouzin, Director of the Cyclades Network, had taken over the chair of TC-6 following the resignation of Alex Curran and the chairman, protem Keith Uncapher of Information Sciences Institute. In this way the reins, at least for the present, have been handed to Europe. This has come at a time of growing world-wide interest in network standards, as the planned public data networks begin to come into service and their potential users become aware of the services they can look forward to in the next few years.

The continued growth of INWG poses real problems in organisation, and some changes were made at the beginning of 1977 to try to simplify the way it operates. These include the introduction of an occasional newsletter, and the classification of papers into two categories by singling out those dealing specifically with protocols, for separate circulation.

2. THE PRESENT

The terms of reference agreed by the Council of IFIP remain unchanged at the present time, except that the two sub-groups referred to in section 5 have become six, in order to deal with more specialised subjects. They are:-

a) assessment of services given by CCITT recommendation X25
b) experiments to evaluate facilities offered by networks
c) identification and definition of new protocols
d) formal methods for the specification of protocols
e) gateways and the interconnection of networks
f) security and related topics

Briefly, the main purpose of work in these areas is as follows:-

a) The potentialy wide adoption of the CCITT (International Telegraph and Telephone Consultative Committee) recommendation X25 for a Virtual Circuit Interface for public networks has led to a number of papers by INWG member on various technical aspects. As the interface begins to be implemented on a large scale, there will be a need to exchange information between people engaged on development work, and WG.6 can play a useful role in this area.

b) Despite several years of research and development work with private networks, there is still a wide diversity of opinion on what facilities should be provided. There is an urgent need to establish comparative data on the use of different services, and the design and conduct of experiments is a major task foreseen for the group.

c) With the prospect of the new public networks, the agreement of high level protocols, necessary for the exploitation of a heterogeneous research network, is becoming urgent. It is in this area that the work of WG 6.1 is most intense, with many working papers being circulated at the present time.

d) In the longer term, there is a pressing need for a way of specifying protocols in a formal way, so they can be rigorously studied and simulated to detect any flaws in

their basic design. This work will become more challenging and worthwhile as the complexity of some of the higher level protocols increases. It has an important bearing on the understanding of the basic mechanisms involved so that more efficient methods of communication can be devised.

e) There has long been an interest in gateways between networks from the point of view of linking the research networks together. With the advent of public networks the problem becomes more that of connecting private networks to the public network. This accounts for a continuing interest in this subject.

f) Security and the related topics of privacy and reliability are matters that received little attention in the early days when the problems of actually achieving intercommunications were so formidable. Now that some of these problems are being resolved there is a growing concern about ways of ensuring that information is not lost or passed to the wrong destinations. The awakening interest in this area is marked by the formation of this new group.

3. THE FUTURE

In these days of many public conferences and good communications between scientists, it is pertinent to ask what role can usefully be played by IFIP WG6.1 . In my view, its special achievement has been the bringing together of experts from many countries and many projects, allowing them to exchange ideas and papers that are at a very early and formative stage. In this way INWG serves a unique role in keeping its members informed on what is happening at the frontiers of their subjects. The membership now represents users of many types of computer systems and communication networks, and can fairly claim to mirror a cross-section of system users from many parts of the world.

APPENDIX 1

TERMS OF REFERENCE

W.G. 6.1 INTERNATIONAL NETWORK WORKING GROUP

1. NAME

The group shall be called Working Group 6.1 - International Network Working Group, abbreviated INWG.

2. SPONSORSHIP

INWG I will be sponsored by and will exist as a working group of technical Committee 6 (Data Communications) of the International Federation for Information Processing.

3. MEMBERSHIP

INWG activities will be co-ordinated by a chairman appointed by TC6. The chairman will provide regular reports of INWG activities to the Chairman of TC6.

The initial list of members will be approved by TC6. The chairman of INWG may appoint additional members who have demonstrated their ability and willingness to contribute to the work programme of INWG. The Chairman of INWG will obtain approval from the chairman of TC6 for members added under this clause.

4. PURPOSE

INWG will study the problems associated with the interworking of computer networks planned or under construction in various countries. The ultimate goal is to define the technical characteristics of facilities and

operating procedures which will make it possible and attractive to interconnect computer networks.

In pursuit of this goal INWG will, from time to time, define and publish guidelines for the interconnection of participating networks. Where necessary and possible INWG should test the guidelines on experimental interconnections between those networks on which meaningful experiments can be done.

5. ORGANIZATION

Working Subgroups of a temporary nature may be created by the chairman of INWG to study specific problems related to INWG's ultimate goal INWG members may act on more than one subgroup. Two suggested subgroups are:

a) Communications System Requirements
b) Host-Host Protocol Requirements

6. MEETINGS

The meetings of INWG will be scheduled and publicized through the IFIP Bulletin. The chairman of TC6 will be advised of the calendar year's schedule and location of meetings, and of any budget requests by 15 September of the preceding year.

7. COMMUNICATIONS

The chairman of INWG will distribute to the INWG membership all reports, announcements and other material which he deems to be relevant.

8. CHANGES TO CHARTER

The Charter may be amended by majority consensus of INWG membership with the approval of the current TC6 Chairman.

9. DISSOLUTION

INWG will be dissolved by decision of the Chairman of TC6.

10. WORKSHOPS AND SYMPOSIA

INWG, through TC6, may sponsor any workshops and symposia on computer communications.

NETWORK DESIGN PHILOSOPHIES

Louis POUZIN
Director PROJ-PILOT
IRIA France

INTRODUCTION

George Feeney (128), talking about the future of computing utilities, made an interesting comparison between computing utilities now and the power plants 70 years ago in the United States. In 1905, about 50 000 local power plants owned by private companies in the US produced half the electric power, largely for their own internal needs; now there are about 200 large plants that produce about 95% of the whole electrical power in the US. Similarly, there are presently over 60 000 computers in the US, almost all of them private and providing a strictly local in-house service; but in the future there will be just a few behemoth centres.

In answer to a question about reliability problems with these computing utilities, Feeney said that they had a fully battery-protected power generator for their own computing centre. In the future, there will probably be a few networks providing nearly all the computing power; most of the power plants will probably have their own computers with full-scale memory protection, and also powered by a full-scale battery-protected power generator, for reliability.

TERMINAL NETWORKS

But first, what we should look at is what networks are. They started about 15 years ago with what we now call star networks of terminals. This is the type of network that usually contains a central computer, with a lot of terminals linked through private leased lines or switched lines, but at least one line per terminal, as

shown in Figure 1. That is the way in which most early time sharing

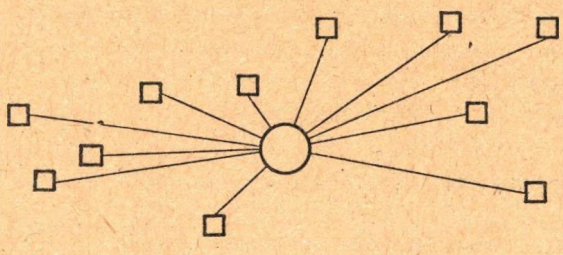


Figure 1: Star network

systems were built in the US and in Europe but it was very quickly discovered that these were not a very good solution if there were many terminals; since you need one line per terminal, this gives a very low usage of the line and also you need as many adaptors in the central computer as there are lines. The star arrangement takes more hardware and more software operating time.

Multi-drop lines

So one of the first improvements that went into networks was to have multi-drop lines, as in Figure 2, especially for those networks

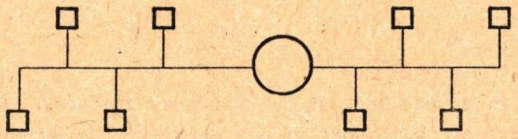


Figure 2: Multi-drop lines

where the terminal use is very low, and also where the terminal input is much more restricted than in a time-sharing system. It is

typically the way in which transaction systems operate. They have messages of certain lengths and the traffic is more or less predictable. So one improvement was to have several terminals on one line. That would provide a much better utilization of that line, and also less hardware, because you need only one input adaptor for several terminals.

One major factor about these improvements in topology was the cost of lines. Even though the cost of lines has dropped by a factor of 4 or 5 in the past 10 years, it is still considered an expensive item in all teleprocessing systems. So one of the major goals was to save on the communications cost; and that is what happened with the multi-drop lines.

Looped lines

But since the more terminals you put on one single line the more liable you are to be stuck with a broken line, a further improvement was to have some kind of loop as in Figure 3, so that, if a line

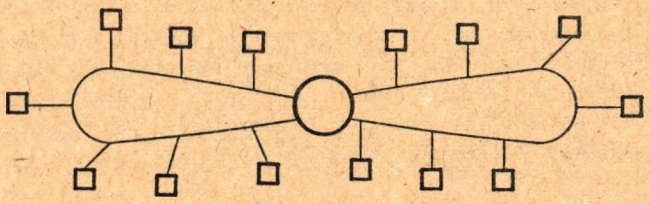


Figure 3: Loops

fails, the message can go round the other way and still get through to its destination. You may have more loops at the far end of these large loops, so that there is access to a particular terminal in the event of a line failure in that part of the system.

Most of these changes were intended to improve the communication services, by achieving more throughput, less cost, and better reliability. These improvements were necessary because the communication systems that were available on the market, namely the usual PTT

services, were not completely satisfactory for data transmission.

Multiplexing

After that, more improvements came; since many lines were going to the same direction, why not have some kind of multiplexing, as shown in Figure 4, just to save on the cost of the lines, by having

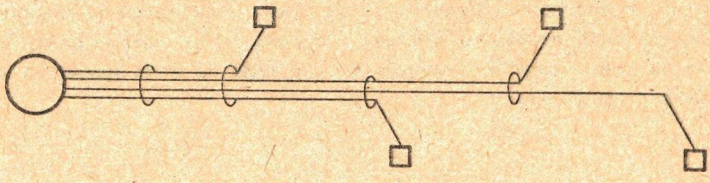


Figure 4: Principles of multiplexing

higher-speed lines at lower cost? So what people did was to develop multiplexing hardware, usually working on time division or frequency division techniques, just to channel several logical channels onto a single physical channel. This also saved on cost. It did not improve much on reliability, except that maybe the higher-speed lines had better treatment and were better maintained. But that was probably a marginal improvement; the main saving was in cost.

But it had some drawbacks because it is not possible to take one line out arbitrarily, not the way it is shown in Figure 4; usually you have to multiplex or demultiplex several lines at a time, which means that if there is only one terminal in a particular area, you have to demultiplex two or four lines, and then run separate single lines to other places where there are more terminals, as shown in Figure 5.

Programmed concentrators

With the advent of cheap minicomputers it became economically acceptable to replace hardware boxes with programmed boxes, to get more

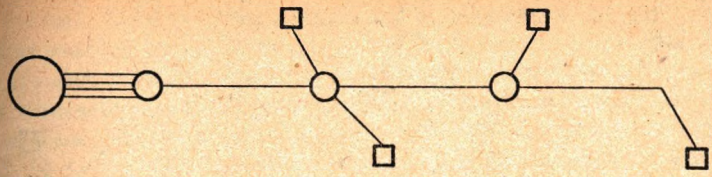


Figure 5: Hardware multiplexors

flexibility. One could have used the same schemes as were used with the multi-drop lines; what is represented as terminals in Figures 2 or 3 may be concentrators having various terminals attached to them, as shown in Figure 6. This changed the picture quite a lot, because in the scheme shown in Figure 5 the multiplexors were actually operating completely in hardware and there was no way to program the multiplexing. Actually, every single line had to be demultiplexed before entering the computer system, and it looked then from the computer system as though there were a lot of separate lines. But using programmed concentrators resulted in a lot of by-products, which I will cover afterwards.

The nodes could be programmed and, consequently, they could talk to the computer centre separately. In other words, the computer could have separate conversations with each one of the concentrators, as if they were separate terminals, although they were linked through a cascade of intermediate concentrators. That was the beginning of networks. There was some very simple message switching between them.

I should now like to describe briefly some of the advantages of concentrators.

Speed adjustment

Using just multiplexing techniques did not change the speed at both ends; the computer had to send or receive bits at the same speed as the terminals. Although they were multiplexed along the way, everything would occur as if the terminal had been hooked directly to the computer. But with concentrators that can store messages and re-transmit them, you could have a change of speed; in other words,

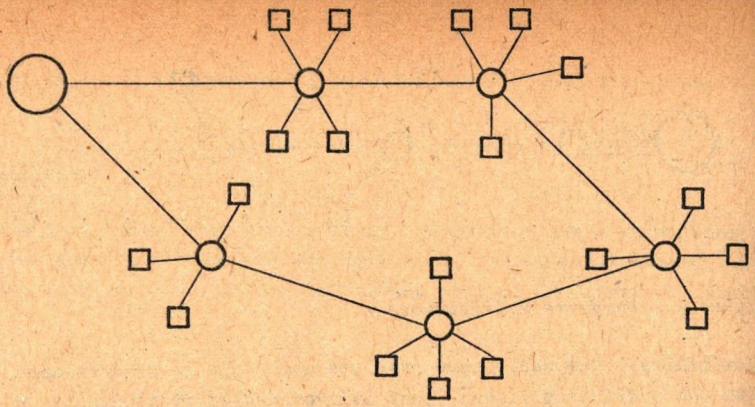


Figure 6: Programmed concentrators

you could communicate with the computer at high speed and with the terminal at low speed.

Load levelling

Since there was intermediate storage, even though it might be for a short time, this would bring some kind of load levelling. With multiplexing or multi-drop lines, the complete channel had to be available for each message to be transmitted. Even though they were multiplexing, each separate logical channel had to be free from end to end whenever there was a message to transmit. This is not true with concentration. A concentrator can multiplex in time the channels that are available. Sometimes it can delay messages in order to avoid high peaks of traffic. This produces a better utilization of the line, without losing messages.

Error detection

Also it could do things that were not usually done, at least not as well as we would have expected, in terms of error detection. Using multiplexing techniques on polled lines or low-speed terminals, error detection was very often limited to a parity bit, which in transmission is not a very good scheme, as you know, because if you lose several contiguous bits, there is one chance out of two that the parity bit will be good after the error has occurred. Using a concentrator, one could block characters or messages, several at a

time, and send that on the line in synchronous mode, with a cyclic redundancy code or checksum, which was a much better error detection scheme. This was considered a major advantage, not especially in time sharing where there is so much redundancy in messages that you can spot right away whether a message has been spoiled, but in data transaction systems, for example airline reservations or banking systems, where it is not obvious that the data has been corrupted. So it was very important to have very good error detection and protection in this environment, even though the lines were not good.

Message control

Also, you can have some kind of numbering scheme to control the succession of messages, just to make sure that none has been lost. This numbering has no meaning for the user, it is only a convention between concentrators and the central computer. If a message is missing in the series, it can be repeated because it is stored at the other end, which would not occur with a very simple terminal. With a very simple terminal, the message is stored in the human user's mind and if it gets lost, he has to re-type it. That has the advantage that he can change his mind.

But these checks were important because in the early systems no one had any idea about traffic that would occur, because nothing comparable had been designed before so traffic was mostly a mystery. With a hardware multiplexor, no-one could have any ideas about the traffic, the central computer had to take care of that. But most of the time the central computer comes with the manufacturer's software, which contains no statistics gathering, no accounting, nothing, because the manufacturer usually does not like the customer to have any idea about what is happening in the computer system. So very often statistics were not taken with early systems. With concentrators, however, since they were very small machines, anyone could program them and put counters in right places, and also have a real-time clock to put time stamps on messages, to know at which time the central computer went down and so forth, and so achieve better behaviour for the human users, such as putting out sensible messages.

Fail-soft operation

That also brought some better fail-soft procedures; in other words, whenever the central computer went down you could send the user a message telling him to stop typing because there was a temporary

breakdown. Of course, that could not be said by a broken-down computer.

Device handling

Another desirable feature that is difficult without concentrators is nice device handling; in other words, programming the terminal so that it takes into account all the varieties of terminal behaviour, such as the time it takes for a carriage return, the time it takes to bring the paper up to the first line of the page, to turn on or off some lights, to do a lot of nice things that usually are not done in central software. That brings a lot of comfort to the user.

Flexible configuration

Having concentrators was a way to decouple the terminal and the central computer. As you know, many manufacturers do not support every terminal, they support just a few of them, especially their own and a few 'foreign' terminals such as Teletypes. With a terminal from some small company, very often you have to take care of the software yourself. It is much easier to build that within the concentrator and, from the computer's viewpoint, it can be programmed to look like some other terminal that the computer knows. That is a way to change the configuration of terminals and lines, without major upheavals in the central software.

All these advantages can be carried over into computer networks because they also use small computers to carry messages.

COMPUTER NETWORKS

After these networks of terminals, we came to computer networks, as shown in Figure 7. I do not think that we can yet say that we know everything about computer networks; what we can say is that we have made computer networks. How they work is another story. With computer networks, the traffic was expected to be quite different since, instead of having real terminals, you have abstract terminals, which are processes, programs, or modules, or whatever you want to call them, and which are housed in big computers. They may also have real terminals, of course, but these terminals are appendages to

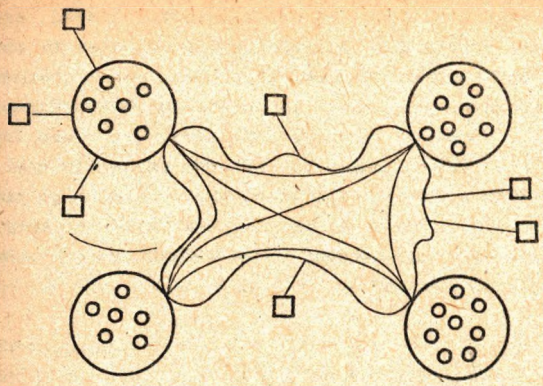


Figure 7: Computer networks

the active part of the abstract terminal, and this active part is usually a program. In this case, the traffic is quite different. It is much more unpredictable because usually terminals are served by people; if the terminals are a bit more automated, like card readers, their traffic pattern is predictable. But here it is quite unpredictable. All we can say is that the traffic is very bursty but we have no statistics. Consequently, the bandwidths that are required from the transmission system are very variable, very fluctuating.

Also it is not realistic to expect every computer in the network to have the same characteristics so heterogeneity is a built-in feature of computer networks, especially for the I/O system. There is no reason why an IBM machine should have the same procedure or the same code as a UNIVAC or an ICL machine. It can happen, of course, but it does not happen very often.

Since we are concerned with data processing, the error rate has to be very low. In data processing we would tend to consider an error rate of 10^{-6} as bad, 10^{-8} or 10^{-9} as quite acceptable, and probably 10^{-10} as quite good; beyond that it is almost impossible to distinguish human errors from computer errors. Usually quality improves over the years and we may be much more severe a few years from now. A bit error rate of less than 10^{-10} was what was required from the communications system.

A computer network also needs to be fail-soft. Indeed, access from terminals can be provided by means of a big computer to which they are attached, but if this computer goes down, users should be able to access other computers. That is one of the by-products of computer networks. Computer networks supposedly are there for providing services to a large community; if each person in the community has to go first to a central computer, why should he go to another one? So a good way is to go to the network first and then pick up one of the computers that happens to be available. So some kind of direct access is needed.

Conventional telephone system

What would we see if we were using the conventional telephone system to link computers?

- With dial-up lines, it takes from 10 seconds up to a few hours to get a call through.

- The bandwidth of conventional telephone lines is usually low; it is not necessarily low but very often it is limited to 4800 bauds. You can lease higher-speed lines, of course, but they are much more expensive and it takes more time to get them.

- The bandwidth is also fixed by design. All you have is a wire and you have to adapt the speed between the two ends. You cannot just hook two computers through a telephone line, you need some machinery to match the bit rate at each end.

- The error rate varies from 10^{-3} on bad switched lines up to 10^{-5} on reasonably good leased lines. This is below data processing standards.

- The only statistics you get is a bi-monthly bill with the amount owing but no detail. It is probably not enough for a computer system.

- There is nothing to prevent failure. If the line fails, you have to call the phone company, and it takes between a few hours and a few days to get it fixed.

Digital telephone system

There is at present no digital telephone system, except for a few experiments, and if we had such a system, it would have the following implications for networks:

- The system would probably be organized so that whenever you have a conversation you use a channel from end to end, because that is usually what voice requires. That would mean tying up a channel for the whole conversation, even though your traffic is bursty and does not use the whole bandwidth all the time.

- The voice transmission also uses periodic sampling. You are not going to sample your data, of course. I suppose that you could change that device and use some kind of register to put your data in the system. Periodic sampling means that the traffic is quite predictable because the bit rate is chosen by the telephone system, not by your system.

- In a voice system, it does not matter if you lose a few bits, it does not change your voice; if it happens to change it, then you repeat what you have said. So usually the bit error rate does not have to be very low.

- I do not know what the call delay on such a system would be, but it would probably be designed to be acceptable to human users, which means a few seconds. If this system were designed with traffic statistics in mind, they would probably be based on a call duration of 1 to 10 minutes, because that is typically how long telephone conversations last.

If you compare these characteristics with what data transmission would require for communication between computers, I do not think that any one of these objectives would fit a computer system.

Purpose-built data networks

If we had special data transmission systems designed for computers, that would probably be ideal. The only thing is that we do not have at the moment very definite plans for such systems. We know that there are some experiments going on. We know that the various PTT administrations have a lot of ideas, but there is no date, no deadline.

We know that it takes time to implement a national system, so in the meantime we have to do something.

MESSAGE SWITCHING

What people have done so far is to use some kind of message switching, using, as we shall see, computers and telephone lines. First, what is message switching? It is something that everybody has known for many years, perhaps without knowing that it is called message switching. You have a piece of data to send off and you put an address on it and some control information, as shown in Figure 8, since

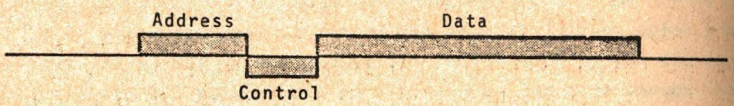


Figure 8: Message format

the system usually needs some kind of redundancy check or a few bits for various functions. You then give the message to the system, and it is carried rather like a letter, which goes into the mail box, then into a bag, then into a truck, and so on, until, finally, it surfaces in the destination mail box. In other words, it is stored and wherever it is stored it is taken out again and put in a different store, and so on, as shown in Figure 9. That is all there is to store and forward.

Messages

A human message is usually a few words or a few sentences. But in a computer a message is anything between one bit and millions of bits there is no sensible average between these two extremities. It can be a whole file or a whole data base. It can also be an event that

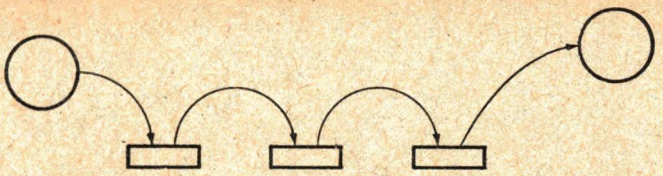


Figure 9: Store and forward

says "Yes, I'm working; that's my message".

Message switching systems

Let us take a look at the early systems that were designed for handling message switching. The first one appeared in about 1962, over 10 years ago. At that time, telephone lines were very expensive compared to computer costs; they were much more expensive than they are now. Also, we did not have the sophisticated modems that we have now, and the usual speed did not go much beyond 1200 bits per second. So the main objective was to save on communications cost and to use lines at the highest bandwidth possible. That is what the designers had in mind while designing the kind of system shown in Figure 10.

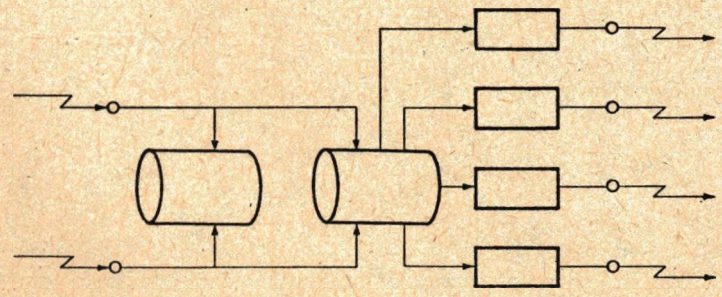


Figure 10: Early message-switching system

The only way to have lines at the highest possible utilization rate was to have queues. Wherever you want to have better bandwidth, you just build queues, as in supermarkets. If you want to optimize the cashier rather than the customer, then you build queues. Since the queues of messages were so large, they could not be stored within the central core of the computer, so they needed secondary storage. As soon as you have secondary storage, you have delay; messages would take between a few seconds and half an hour or more to get through the system. In this case, it is very unpleasant for the user to lose messages, because if he has to re-send a message that went off half an hour ago, it means a lot of management of messages at the user end. What the user wanted was to be able to forget about messages once he had delivered them to the transmission system. This implies a sophisticated operating system, with dual systems, back-up software, and all that jazz; it came to be very complicated and very expensive, just because there were low-speed lines. But that is the way early systems were designed. A lot of them are in existence now and I think they work pretty well.

Delay

But now let us take a look at why we have delays and what we could do to reduce them. The delays are made up of several components, as shown in Figure 11. First, there is a transmission delay, which is purely a characteristic of the electrical signal, which is governed by the line speed. When you get into a system, it may not be ready to handle the message right away, so there might possibly be an input queue. Then there is the processing of the message, which takes some time. Then there is another queue before it can get out; the line may be busy so it has to wait. Then there is the time to transmit it. If it has a lot of bits it takes time to get these bits out.

It can be seen from Figure 11 that the delay caused by the first queue and the processing of the message is roughly related to the number of messages. Usually the processing is about the same, regardless of the length of the message; it is just the fact that one message has to be handled. Since the processing time is usually related to the number of messages, queues are also related to that number; that is queueing theory. The second queue is different because the server here is the transmission line, and the service is dependent on the line speed and also on the message length.

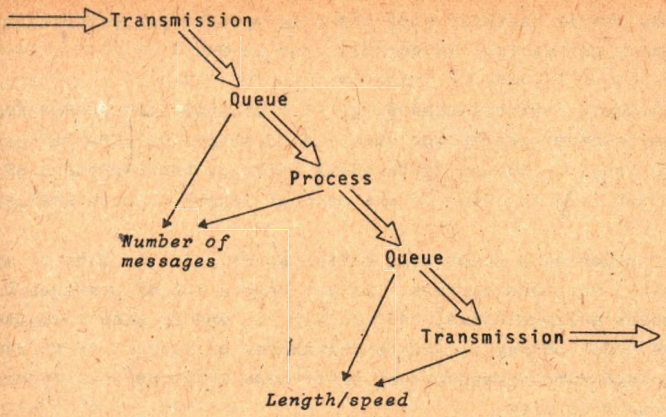


Figure 11: Elements in message delay

So the delay is not constant for each message. The queueing time increases with the message length, because the total number of bits to evacuate is larger. This is also in inverse relation with the line speed, of course, because if the line speed is twice as fast, then you have to wait half as long.

One way to reduce the total delay in a system is to have shorter messages; another is to have higher-speed lines because the delay is in inverse relation to the speed; finally, you can organize your system in such a way that there are almost no queues, because queues mean delay.

Fragmentation

One way to minimize the queues is to introduce some kind of fragmentation within the system, which means that if you have a bunch of bits to send off, you cut them into small pieces. Instead of having to carry the whole bunch of bits, store them, take them out again, put them into a different bag, and so on, you can have various processors or active components that take one piece at a time in a chain, as in an assembly plant. Each one takes a piece and as soon

as it is put in the next bag, there is another processor that takes the piece, and carries it forward, and so on.

In this way, the total time to get the message through the system is much shorter because the delay is no longer so affected by the message length. Though it is still related, the component of the delay that is a function of the message length is much smaller.

Fragmentation also means less buffer space. If you want to store something, you need buffers. If you have a lot of messages to carry, that needs a large store, and finally you end up with secondary storage. So the way to reduce the number of buffers is to have the message more spread out so that there is no need to store the whole bunch of bits.

Using fragmentation is also a way to squeeze in messages of higher priority. When you have a long train on a railroad system, there is no way to get cars ahead of it; you have to let the train go first. But if you fragment the train into small pieces and store each car at each intermediate station, then you can sneak in faster cars or faster trains. So usually there is some kind of priority needed for special messages, service messages, or perhaps some higher-priority traffic. This would not be possible if you were using very long messages that could not be broken.

Error recovery is also facilitated by fragmentation, because if the message gets corrupted at the other end, the only way to get it in its correct form is to re-transmit it. The overhead traffic associated with re-transmission is related to message length; if you have only small pieces, you have to re-transmit only small pieces, and on average there will be a gain in overhead traffic.

One drawback of fragmentation is the transmission overhead. By fragmenting information, of course, we have more pieces to manage. This means more addresses to look at, more bits to be included in addition to the information bits, and so on. That is the cost we have to pay for all the advantages of fragmentation.

PACKET SWITCHING

Packet switching is a term that was introduced into the literature

by Donald Davies about six or seven years ago. Packet switching is message switching, except that we call them packets. There is no relationship between the meaning of the packet and what a user would call a message. In early message-switching systems, the message was usually what the user would call a message, a Teletype line, for example; in other words, there was a very immediate relationship between user messages and system messages.

Packets

In packet switching, there is no such relationship, packets are just small strings of bits that are limited in length, and joined together they make up what a user would call a message, or what the outside system would call a message.

These packets are sent through a system, from computer to computer, from node to node, and they arrive at the other end, where, of course, they have to be put back together. The delay is partly a function of the number of nodes, because each node entails some storage and re-transmission, so one factor in the delay is proportionate to that; another factor is proportionate to the packet length divided by the line speed.

With a line speed of the order of 100 kilobits per second and with a very short packet, the main component of the delay will be the number of nodes. For example, it would take about 90 milliseconds to get 1000 bits through a network, having seven or eight nodes to go through. If you take a very long message that is broken up into a lot of small packets, the main component of the delay will be the packet length, not the number of nodes; in which case it may take something like a quarter of an hour to transmit a file of 10^8 bits. That is still acceptable. That is just an indication of the order of magnitude in these figures.

Network topology

Suppose you have a lot of nodes and a very large network, it would probably be a good thing to cater for two kinds of traffic: high-volume traffic and high-speed traffic. We already have this kind of arrangement for cars, trucks, and trains; so we could have some kind of highway, as shown in Figure 12, with a lot of small nodes

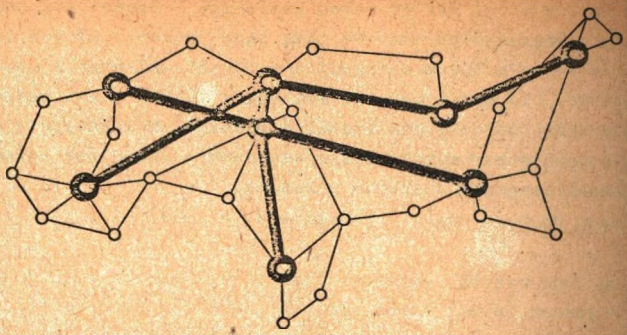


Figure 12: Hierarchical network design

with local traffic to cater for all local exchange; and a lot of big nodes to reduce the number of nodes to go through to get across the whole network; and also to have higher bandwidths between these big nodes. The design for large networks will probably involve some kind of hierarchy, not necessarily in terms of technology, but hierarchy in terms of routeing packets.

One of the main aspects of a network is its capacity, that is, how many bits it can get through from one place to another. Of course, it can vary, depending on the traffic. For example, if you have a long, narrow country, like Norway or Chile, the traffic will probably go mainly along the longer axis and not across it. One can therefore attempt to optimize the traffic in that direction rather than the other. Similarly, I should imagine that in the United States there is more traffic going coast to coast than north to south, although I am not sure about that.

Another main characteristic is the transit time, that is, how long it takes a packet to get through this network. Transit time is very important for traffic in conversational systems, or for computer-to-computer traffic. Reading a file within a different computer is quite reasonable if the transit time is in the order of a disk access, because you may consider the files on other computers like those on yours.

Cost and reliability are, of course, also important aspects. So far,

since most networks are experimental, cost and reliability are difficult to assess and also difficult to measure, because the users are not quite real users; they are research users.

Packet overheads

I said earlier that there was an overhead traffic associated with fragmentation. If you use packets, it means that the traffic has to go from node to node, and somewhere or other it has to be acknowledged. You cannot just send packets blindly; there must be some feedback scheme that tells the sender that things are going well and that he can go ahead. So there is a certain amount of acknowledgement traffic, which is related to the number of packets. The smaller the packets, the larger the overhead for acknowledgement; the higher the number of nodes, the higher the overhead traffic for acknowledgement,

Another possible drawback is that when you send packets in some order, 1, 2, 3, 4, they will probably go through in different ways or they may go ahead of one another, and so they may arrive in a different order. It depends on several factors, such as the software but there is no built-in machinery to put them in the right sequence unless it is done deliberately.

We also have some overhead associated with the word length. Let us say that we are sending messages from one computer, which is, say, a 36-bit machine, to a message processor at one of these nodes of the network, which has a 16-bit word. It takes two words and a part of the third one to store the 36-bits, and then it winds up in a Control Data machine, which has a 60-bit word. It does not fill up even one word. So what do we do with the rest of it? It is probably wasted. So that is an additional overhead associated with fragmentation that is not related to any user fragmentation.

Error control

Messages can arrive out of order; packets 1, 2, 3, 4, may arrive in the order 1, 3, 2, 4, What may happen also is that you never see packet 2; it gets lost somewhere and so it never arrives. Also you may have two copies of packet 3. Which one is right? They may be copies of each other, or it may be a wrong message 3. It is not

unlikely that in some place in the network a packet will be lost because the processor just went down; there is no way to recover what is inside the processor, so it is lost. Also, at some time a processor might have sent a packet in one direction, but it has never been acknowledged; the processor thinks that the packet has been lost, so it sends it in a different direction. But actually it has not been lost; it went to the destination. So there are two copies of this message arriving. That is the kind of thing that error control machinery has to take care of if we want things done in an orderly manner. So that is the problem of reassembly and sequencing of packets, so that they are put back into the right sequence and are combined into the original information.

Error control requires buffers, delays, and some kind of numbering scheme for reassembly and sequencing, sequencing being just reassembly at a higher level.

Adaptive and fixed routeing

The routeing associated with this kind of network is usually of two main types. The first is the fixed routeing type; in other words, when we want to send packets or messages, we pick up a route and we do not change it for the whole time the traffic will hold for a particular pair of users; secondly, we can use adaptive routeing, which means that packets go any way they can and it is up to them to find a way through the whole network.

These two methods have different implications. If you use fixed routeing, it takes some time to set up the route; there is no way to get round that. It requires path tables because someone has to remember all these routes. This information is either stored in a centralized system or it is scattered over the whole network, but you need a table somewhere. Because the route is fixed, it cannot take advantage of the meshing of the network, of the optimum route, so capacity would probably be limited, too, in some sense. If there is a failure, the route has to be reorganized. That is a process that will take some time and may entail some failures at each end.

On the other hand, it has some advantages. It is very easy to have sequential traffic on a fixed route. Just as in a railroad system, there is no easy way to get a car in front of one another. It will also supposedly be stable because all the parameters have been fixed.

We know the capacity and the route, so it is much easier to control. But it still needs routeing tables, because when you want to set up a route, it has to take care of some parameters that have to be chosen depending on the whole traffic within the network.

If you use adaptive routeing, the first two parameters, call set-up and path tables, are not needed because each packet has to find its way without taking care of the initial call or path table. They have to use the knowledge of the overall traffic for that. The capacity may be larger because they can use various routes, which may not be as loaded as in fixed routeing. Adaptive routeing is supposedly more resilient because if one line fails, the message can take another one. But the message is not naturally in sequence; it is naturally in disorder. The traffic can be stable, but that requires a lot of finely tuned design because, since the routeing is adaptive, it can run wild and so requires more care than fixed routeing. It also needs routeing tables, like fixed routeing.

THE CYCLADES NETWORK

We are presently building another computer network in France called CYCLADES. It is intended mainly for exchanging information between data processing centres and accessing data bases. It is not intended mainly for conversational traffic; it may be used for data base enquiry, which usually consists of just one question and one answer, instead of a long conversation. It probably would be useful for remote batch or time sharing in some local experiments.

The centres are mainly universities and research centres, and in the second phase there will be administration data processing centres. It contains various machines of different make and it uses packet switching with a small CII computer. Most centres are located in four areas in France, as shown in Figure 13, and we have a lightly meshed network with more than one route between nodes.

It is still an experimental prototype; it is not designed to be a general system or a national system. In terms of its characteristics, the packets will be in the order of 2000 bits and there will be no sequencing, which is a controversial point. We do not think now that sequencing is really required until someone brings a proof of the contrary; anyway it can be done in the host system or it can

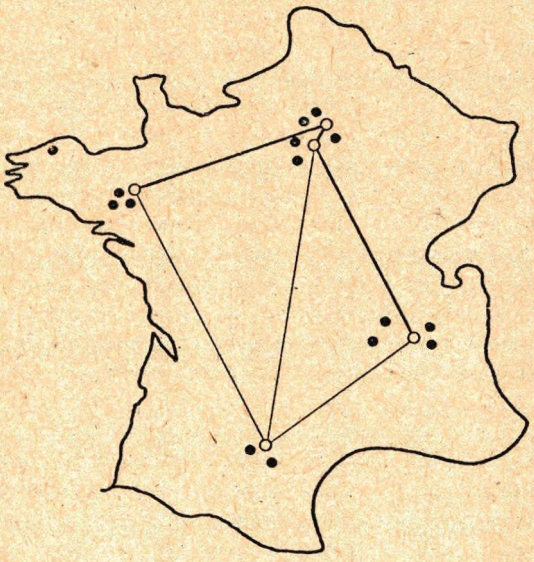


Figure 13: The CYCLADES network

be added as a service within the communications network.

The protocol, in other words the conventions used to communicate between the large computers, is based on a very simple scheme that sends letters and receives acknowledgements, but it can also be more sophisticated and provide virtual links, that is, sequential connections. But all that is implemented not at the communication network level, but at the host protocol level.

Finally, I should like to say a word about the timetable of this network. Experimental communication started in mid-1973 and we hope to have more nodes by the end of 1974; and to have real users being introduced by the beginning of 1975.

SOME PROBLEMS OF COMPUTER NETWORK OF POLISH SCIENTIFIC CENTRES

Mieczysław Bazewicz, Teodor Mika

Technical University of Wrocław

Abstract

The design of computer network established between scientific centres in Poland has put some problems about the computer cooperation. The paper attempts to precise the cooperation problems at three levels: user level, network procedure level and communication subnet level. A method of information flow evaluations for network design optimization was also considered.

1. NETWORK PURPOSES

An analysis of computing needs of universities and other scientific centres, made at the Technical University of Wrocław during few years' computer system exploitation has demonstrated, that some applications, as student training in computer science, may be realized in a local time sharing system. Other applications require a resource enlargement, which may be attained by providing to a user an access to resources, services and data bases of nonlocal systems. The resource growth allows to a user, to realize such computing jobs that cannot be executed in local system with own data sets and program library. The resources may be increased by linking local computer systems to a computer network.

There are three basic functions of a computer network of scientific centres to perform:

- load exchange; some scientific centre will realize its jobs in a greater computer than its own,
- special service exchange; the network has to provide to users an access to special languages and packages inaccessible in their own computer,
- access to data bases implemented in several host computers.

2. USER-USER PROTOCOL

Some isolated time sharing system may be considered as a system, in which a set of users U exploit the hardware resources H, the special services S, and data from previously prepared data bases D. An isolated system can be represented by a structure of four sets:

$$T = \langle U, H, S, D \rangle$$

in which an exploitation relation φ between the user set and the another three ones was defined. The exploitation process of elements of these sets is named user process /pu/:

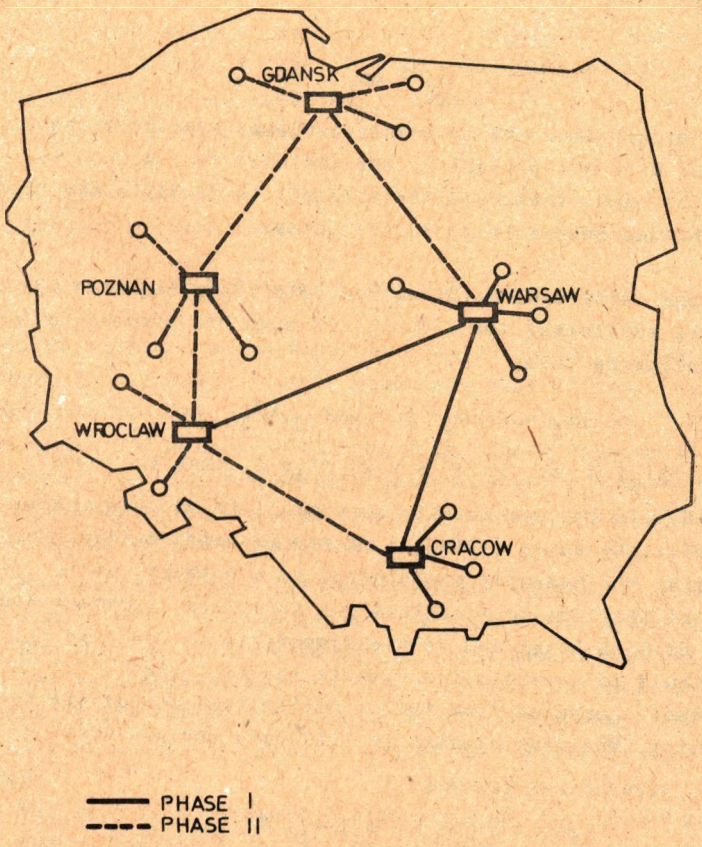


Fig.1. Computer Network of Polish Scientific Centres

$$pu_i = \varphi(u_i)(h_i, s_i, d_i)$$

where $u_i \in U$, $h_i \in H$, $s_i \in S$, $d_i \in D$.

The set of users of an isolated system is named local user set U_L :

$$U_L = \{ u_i : pu_i \in T \}$$

which means, that the necessary hardware resources, services and data are present in the isolated system, and so, that the exploitation of the elements is possible and the user has an access to isolated system.

When the resources, services and data of an isolated system are not sufficient for a user, and when his process is in the following form:

$$pu_j = \varphi(u_j)(h'_j, s'_j, d'_j)$$

where $h'_j \notin H$ or $s'_j \notin S$ or $d'_j \notin D$, then the user should have the possibility to exploit in another system the elements being deficient in the isolated system. For unifying the computer cooperation in a network, it is assumed that the user exploit the resources, services and data of both local and remote computers. It would be difficult to separate for example the exploitation of data in remote computer from the computing time use of the local computer. The user creates in this case two processes:

- the first local process:

$$pu_j = \varphi(u_j)(h_j, s_j, d_j)$$

where (h_j, s_j, d_j) are parts of the elements (h'_j, s'_j, d'_j) realized in local system,

- the second process:

$$pu_{jk} = \psi(u_j)(h_k, s_k, d_k)$$

is created in remote time sharing system:

$$T' = \langle U', H', S', D' \rangle$$

where $h_k \in H'$, $s_k \in S'$, $d_k \in D'$. An exploitation relation ψ between the user and the remote system T' was defined, and both systems T and T' have been linked by a network. The processes pu_j and pu_{jk} communicate each other transmitting some information over the communication subnet.

The set of users who create more than one user process in different network computer is the set of network users:

$$U_N = \left\{ u_j : pu_j \in T, \quad pu_{jk} \in T' / k=1,2,... \right\} .$$

The ψ relation means that, between processes of the same user, in different network computer, some information about hardware resource using, calling special program in remote computer and data exploitation from remote data bases is transmitted. The cooperation between these processes should be made according to a protocol /user-user protocol/ which should answer some questions:

- How a user process informs the local system T about the necessity to establish a link with a remote system T'.
- How to create a process pu_{jk} in the remote system T'.
- How to transmit some programs from pu_j to pu_{jk}, to exploit the hardware resources h_k in the system T', and how to transmit the program execution results in a return way.
- How to call a specialistic program s_k in remote system T'.
- How to exploit data d_k in system T' and how to transmit data sets between two systems.

The protocol should also resolve the problems of program and data structure incompatibility in different network computer.

3. END TO END PROTOCOL

For user process cooperation, a separate process named transport station is created in each computer of the network of scientific centres. The user processes communicate with the transport station on the base of interprocess communication facilities existing in operating systems. Next, the transport station transmits the information over a communication subnet to the transport station of the remote system T'. Transport stations with the communication subnet form the transport service /or post service/ of the computer network. A set of information E unified by transport stations flows over the post service P.

The computer network N may be considered as a system:

$$N = \langle U, H, S, D, P, E \rangle \text{ , where}$$

U - is the set of users of all network computers,
H - is the set of all network hardware resources,
S - is the set of special services of the network computers,
D - is the set of network data bases,
P - is the post service of the network,
E - is the set of information transmitted by the post service.

In the considered system the subset of local users is the set of network computer users, who do not transmit information by the post service:

$$U_L = \{ u_i : u_i \in U \text{ , } E(u_i) = \Lambda \}$$

The subset of nonlocal users or network users is the set of computer network users, who are transmitting information by the post service:

$$U_N = \{u_j : u_j \in U, e_{jk}(u_j) = (h_j, s_j, d_j)\ p_{jk}\ (h_k, s_k, d_k) / k = 1, 2, \ldots\}$$

where $e_{jk} \in E$ is the information transmitted between two processes pu_j and pu_{jk} of the same user u_j, over a logical way p_{jk} belonging to post service P. The system elements (h_j, s_j, d_j) and (h_k, s_k, d_k) belong to the local for the user u_j, system T and to the remote system T'.

The evaluation of the information $ej_k \in E$, such as greater of particular information block, block frequency and information flow intensity, is the introductory phase of the works on optimal network design solution, i.e. the communication subnet node distribution and configuration selection, traffic capacity, and network topology optimization. On the base of the information evaluation the discussed solution can be attained in way of simulation on a model /or alternative models/ of the network.

The logical ways p_{jk}, in general, do not cover the physical links between computers with communicating user processes. More than one logical way can exist between two transport stations, if more than one pair of user processes are communicating with both transport stations. For the evaluation it comes simply to difference both directions of transmission between two communicating processes, i.e. to decompose the logical way p_{jk} on two directions p'_{jk} and p'_{kj}. In this case the network user subset will be defined in the following form:

$$U_N = \{u_j : u_j \in U, e'_{jk}(u_j) = (h_j, s_j, d_j)\ p'_{jk}(h'_k, s_k, d_k)\ ,$$

$$e'_{kj}(u_j) = (h_k, s_k, d_k)\ p'_{kj}\ (h_j, s_j, d_j)\ /k = 1, 2, \ldots\}$$

where e'_{jk} and e'_{kj} are informations transmitted in particular directions, between two communicating user

processes, and their logical sum is the information e_{jk}. By the way, one can note, that the evaluation of values of e'_{jk} and e'_{kj} is simpler than the evaluation of e_{jk}.

The protocol of transport station cooperation named end to end protocol should include the rules of realization of following procedures:

- Establishment, i.e. initiation and activation of the direction p'_{jk} by an agreement with the transport station in remote system.
- Constitution and updating of a table of directions to send and receive some information /p'_{jk} and p'_{kj}/.
- Information receiving and sending in unified form from/to directions.
- Decomposition of the information received from user processes and information composition before its sending to a user process.
- Error and acknowledgment control.

4. COMMUNICATION SUBNET

A transport station is a process sending to the communication subnet the information arrived from the user process, and transfering to user processes the information received from the communication subnet. The transport station is linked to the nearest node of the communication subnet through a channel controlled by the transport station and an adapter for code wonversion.

The information coming from the transport station to the node is transferred in a packet switching mode, over several fully compatible nodes. Each node chooses a path for a block of information named packet or datagram according to the addresses of the source and destination user processes, included in the packet header formed by

the transport station. A node examines for path choice a routing table of active nodes and lines connecting them. A chosen path is, in general, the shortest one, but if the shortest path has a great traffic, it may be an alternative path which assures a shorter transit time. The routing table is permanently updated on the base of periodically incoming information /tt_i/ about the transit time, to enable the node to make the choice decision. And so in addition to the proper information some other information is transferred.

An another than the shortest path can be chosen if the line connecting two nodes or a node in the shortest path are inactive. The routing table is informed of the inactivity by the periodically incoming test information te_i .

Each information is acknowledged. The destination transport station after receiving of an information sends to the source transport station an acknowledgment. The acknowledgment ack_{ij} between transport stations i and j confirms a correct information transmission. The sended information is repeated /rep_i/ if its acknowledgment will not have arrived in a defined time out.

All the information traffic in a communication subnet can be distinguished into proper information, its acknowledgments, repeated information, transit time and test information:

$$T = \bigcup_{jk} e'_{jk} \cup \bigcup_{ij} ack_{ij} \cup \bigcup_{i} rep_i \cup \bigcup_{i} tt_i \cup \bigcup_{i} te_i \ .$$

The decisive for the information traffic evaluation is the character of the flow of the proper information e'_{jk} . The acknowledgments depend really of the proper information. The repeated information depends of the proper information and of the communication subnet fiability. The two last components of the information traffic are monotous, and enhance only monotously the traffic level.

The evaluation of the information traffic has a fundamental significance for the network structure and topology design, and for the choice of node and connect lines performance. The flow characters are used in a traffic simulation on a subnet model.

5. CONCLUSIONS

The computer network of scientific centres offers as standard three services: a form of load sharing, program sharing and data sharing. For performing these services the network should have optimal solutions of some problems about communication subnet performance and network topology. But most of the design time spent in the construction of computer networks is involved in the formulation and debugging of the operational protocols. The mentioned two areas of designing efforts are examined separately in the most of publications. However the areas influence one another. The proposition of problem formulation included in the paper, try to explain the type and form of information exchanged in the network and so a method of evaluation of information flows.

The computer network information flow may be decomposed into few components, and each of them may be evaluated separately to recognize its charakter.

The protocol may be considered at three levels, at the user level, transport station level and node level.

6. REFERENCES

1. D.J.Farber. Networks: An Introduction. Datamation 36-39. April 1972.

2. D.S.Jones. Data Transmission: The Future Network Software. NEDO-London 1972.

3. M.Bazewicz, T.Mika. Computer Network, as a Next Stage of Processing Need Satisfaction /Russian/. Z.f.R. - Informationen. Akademie der Wissenschafften der DDR. p.28-35. Potsdam. February 1976.

4. M.Bazewicz. Sieci komputerowe. Informatyka Nr 2 p.1-5. Warsaw. February 1977.

5. J.Bruno, L.Weinberg. Generalized Networks Embeded on a Matroid. Networks. V.6.N.3 p.303. July 1976.

RELAY ROUTING STRATEGY IN A COMPUTER COMMUNICATION SYSTEM

D A. Rutkowski
Technical University of Gdańsk
80-952 Gdańsk, Poland

ABSTRACT

In designing computer communication system one often has to take into account various paths over which messages may be routed to reach a given destination. In this paper adaptive routing algorithm based on the shortest paths of the first and second order between a given source and destination node is described. These shortest paths are calculated according to the delay tables that are periodically updated and the routing strategy is of a relay type with hysteresis. The influence of the hysteresis loop width and location on the average time delay is investigated. It is shown when such a routing strategy is profitable compared to the routing strategy based on the shortest path of the first order only.

1. INTRODUCTION

The effectiveness of a computer communication network depends to a high degree on a routing rule according to which messages may be routed from a given source to reach a specified destination. A number of dynamic routing algorithms have been described so far in the literature (see eg. [1],[2],[3]) that enable us better utilization of the network capabilities from a point of view of the average time delay than it is possible when the static routing rules are implemented. The most known algorithm is used in the ARPA network. In this case individual nodes are making their own decisions as to which outgoing links to associate with which destinations on the basis of messages exchanged periodically with their nearest neighbors. The implementation of the route chosen consists of setting up at each node along the path a routing table (delay table) that directs messages with a particular destination address to the appropriate outgoing link at that node. According to this table messages are routed over the shortest paths of the first order in the sense of a delay time. This algorithm doesn't take into account the shortest paths of a second order which, as it will turn out, give us a significant improvement and result in a decreasing of the average time delay in the network.

In this paper we shall consider an adaptive routing algorithm based on the shortest paths of the first and second order and we shall assume the routing rule to be of a relay type with hysteresis*)

*) Hysteresis introduced in the switching rule is for the sake of lowering the rate of switchings.

2. THE MODEL AND BASIC ASSUMPTIONS

Let us assume that:

A1. there are two incoming message flows S_1 and S_2 at a node p of a network (see Fig 1). They follow Poisson

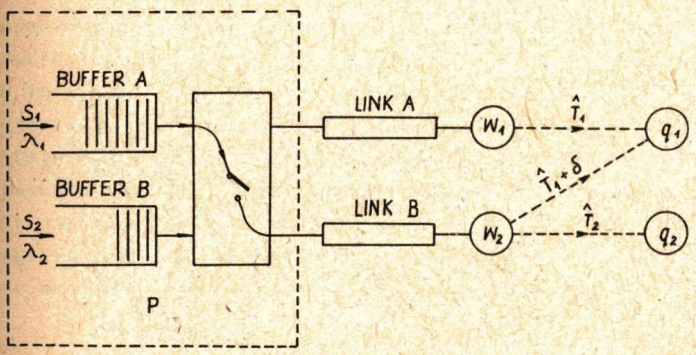


Fig 1 Block diagram of the system considered.
Denotations: w_1, w_2 - the nearest neighbors of a given node p; $\hat{T}_1$, $\hat{T}_1 + \delta$, $\hat{T}_2$ - the estimated time delays to reach the destination nodes q_1, q_2 from the nodes w_1, w_2

distribution with the average arrival rates equal to λ_1 and λ_2 respectively. The flow S_1 is destined to a node q_1 and the flow S_2 - to a node q_2. The probability density function of the message lengths follows an exponential function with average length $1/\mu$

A2. routing rule is decentralized and every node undertakes its own decision as to the outgoing link along which a given message is to be sent. The basis to this decision constitutes a delay table that comprises

estimated time delays from a given node to all destinations including waiting times in all outgoing buffers of that node.

A3 updating of the delay table is periodical

A4 routing rule is of a relay type with hysteresis as is shown in Fig 2 and can vary in a given updating interval i.e. when the buffer A is heavily occupied the

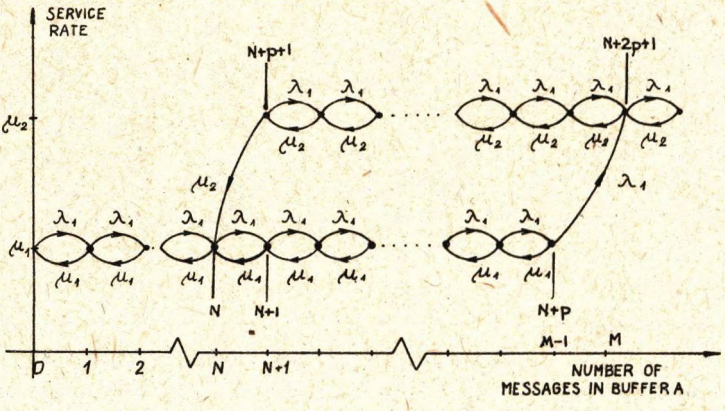


Fig 2. Service rate vs. number of messages in buffer A. Lines with arrows represent the state - transition - rate. The states of the service system are numbered differently from the queue length.

messages from the flow S_1 are directed over both outgoing links A and B and the departures of messages from the buffer B are temporarily suspended.

A5 messages from the buffer A are directed over the first accessible link while using both outgoing links A and B.

The problem is now to find such a routing rule that an average time delay for both message flows is minimum and leads to finding out a mode of link B assignment for the service of buffer A with taking into account the parameters of the hysteresis loop shown in Fig 2 i.e. the value N and the loop width p.

In our considerations we shall take advantage of the results obtained by Gebhard [4] concerning the M/M/1 queuing model with bilevel hysteretic service rate control. In this model the service rate control is dependent on the queue length as shown in Fig 2 where the state - transition - rate diagram is also given. For the arbitrary state the statistical equilibrium conditions are fulfilled. However the statistical equilibrium equations have different form depending on the set of states they are considered in. One can show [4] that the state probabilities are given by

$$P_n = \varrho^n P_o \quad , \qquad o \leq n \leq N \tag{1a}$$

$$P'_{N+i} = \frac{P_N}{1-\varrho^{p+1}} \left(\varrho^i - \varrho^{p+1}\right) \quad , \quad 1 \leq i \leq p \tag{1b}$$

$$P'_{N+p+1} = \varrho_1 P'_{N+p} \tag{1c}$$

$$P'_{N+p+j} = \frac{P'_{N+p+1}}{1-\varrho_1} \left(1 - \varrho_1^j\right) \quad , \quad 1 \leq j \leq p+1 \tag{1d}$$

$$P_M = P'_{N+2p+1} \tag{1e}$$

$$P_{M+k} = P_M \varrho_1^k \quad , \qquad k \geq 1 \tag{1f}$$

where $$P_o = \left[\frac{1}{1-\varrho} - \frac{(p+1)\varrho^{N+p}(\varrho - \varrho_1)}{(1-\varrho^{p+1})(1-\varrho_1)}\right]^{-1}$$

$$\varrho = \frac{\lambda}{\mu_1} \quad , \quad \varrho_1 = \frac{\lambda}{\mu_2} \quad , \quad p = M - N - 1$$

It results from the above formulae that the distribution of the state probabilities is in the case of variable service rate more concentrated around small values of n than in the case when fixed service rate μ_1 is used. This is illustrated in Fig 3. It is important from the practical

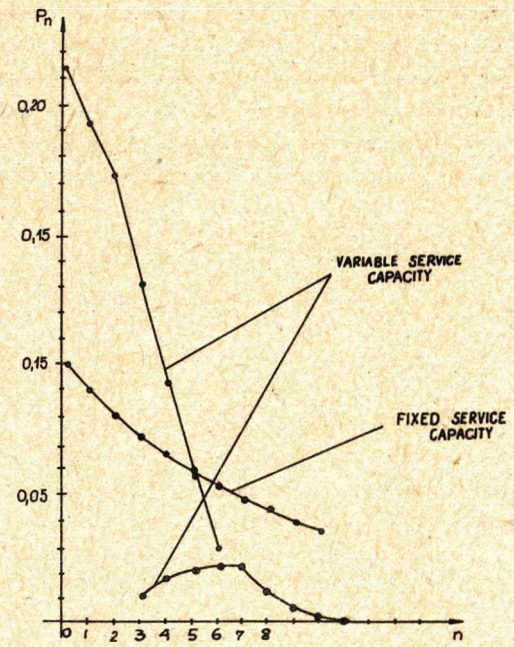


Fig.3. State probabilities in the case of fixed and variable service rate for N = 2, M = 7, p = 4.

point of view to find the expected value of queue length in both cases with fixed and variable service rate.

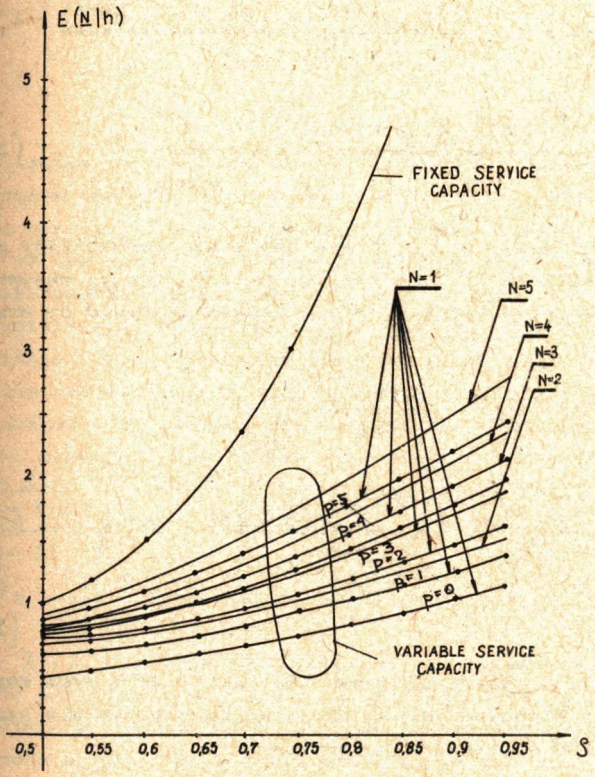


Fig. 4. The expected values of queue lengths vs ϱ ; $\varrho_1 = \frac{\varrho}{N}$

The state probability generating function G (n,z) is the sum

$$G(n,z) = G_1(n,z) + G_2(n,z) =$$

$$= \frac{P_o}{1-\varrho z}\left[1 - \frac{\varrho^P(\varrho z)^{N+1}(1-\varrho)(1-z^{p+1})}{(1-\varrho^{p+1})(1-z)}\right] +$$

$$+ P_o \frac{\varrho^P(\varrho z)^N \varrho_1 z (1-\varrho)(1-z^{p+1})}{(1-\varrho^{p+1})(1-\varrho_1 z)(1-z)} \qquad (2)$$

where $G_1(n,z)$, $G_2(n,z)$ are the state probability generating functions for the service rate μ_1 and μ_2 resppctively

The expected value of queue length is found by evaluating

$$E(\underline{N}) = \left.\frac{\partial G(n,z)}{\partial z}\right|_{z=1} =$$

$$= P_o\left\{\frac{\varrho}{(1-\varrho)^2} - \frac{\varrho^{N+p}(p+1)(\varrho-\varrho_1)}{(1-\varrho^{p+1})(1-\varrho_1)}\left[N + \frac{p}{2} + \frac{1-\varrho\varrho_1}{(1-\varrho)(1-\varrho_1)}\right]\right\}$$

(3)

In Fig 4 there is shown the family of curves representing the expected values of queue lengths in the system with fixed and variable service rate.

3. THE ANALYSIS OF MESSAGE DELAY FOR THE FLOW S_1

The results of the mentioned above considerations concerning the analysis of service with variable rat-e can be used to control the flow of messages from a given source node to destination node of a computer communicat-ion system via two independent routes: the shortest route of

the I and II order respectively. We shall consider the case when for the service of more, on the average, occupied buffer A one can use the link B. From Little's theorem we have the average waiting time $\bar{T}_w^{(h)}$ in buffer A to be

$$\bar{T}_w^{(h)} = \frac{E(N_1 \mid h)}{\lambda_1} \qquad (4)$$

where letter h denotes that we calculate average waiting time under the condition that service rate is variable. Now we can find the average delay for the flow S_1. We can split up the flow S_1 destined to node q_1 into two different routes. On the average buffer A is being served per time unit within α sec. with high rate $\mu_2 = \mu(C_1 + C_2)$ and within $1 - \alpha$ sec. with low rate $\mu_1 = \mu C_1$. It results from the assumption A5. that messages are directed over both outgoing links proportionally to their throughput i.e. $\frac{C_1}{C_1 + C_2}$ messages from buffer A are being sent over link A and $\frac{C_2}{C_1 + C_2}$ messages are being sent over link B. Thus the average time delay for the flow S_1 is given by

$$\bar{T}_1^{(h)} = \frac{\lambda_{11}}{\lambda_1}\left(\bar{T}_{w_1}^{(h)} + \hat{T}_1\right) + \frac{\lambda_{12}}{\lambda_1}\left(\bar{T}_{w_1}^{(h)} + \hat{T}_1 + \delta\right) \qquad (5)$$

where λ_{11}, λ_{12} are the fractions of the arrival rate λ_1 that is directed over link A and B respectively, $\hat{T}_1$, $\hat{T}_1 + \delta$ - the estimates of the time delay while sending mes-sages from node w_1 and w_2 to the destination q_1. From the expression (5) we can easily get

$$\bar{T}_1^{(h)} = \bar{T}_{w_1}^{(h)} + \hat{T}_1 + \alpha \frac{\mu C_2}{\lambda_1}\delta \qquad (6)$$

In the case when the shortest path of the I order only is used for messages waiting in buffer A the average time

delay is expressed by

$$\bar{T}_1 = \bar{T}_{w_1} + \hat{T}_1 \qquad (7)$$

where $\bar{T}_{w_1} = \dfrac{E(N_1)}{\lambda_1}$

4. THE ANALYSIS OF MESSAGE DELAY FOR THE FLOW S_2. COMPARISON OF THE SYSTEM WITH AND WITHOUT RELAY ROUTING STRATEGY

In order to find the average message delay for the flow S_2 we shall assume that breaks in the service of buffer B decrease its rate of service proportionally to the time link B is used for the service of buffer A i.e. the effective rate of service for the buffer B by link B is $\mu(1 - \alpha)C_2$. Thus the average waiting time in buffer B is given by

$$\bar{T}_{w_2}(h) = \frac{E(N_2 \mid h)}{\lambda_2} = \frac{1}{\mu(1 - \alpha)C_2 - \lambda_2} \qquad (8)$$

and the average time delay for the flow S_2 along the path from source node p to destination node q_2 while using relay routing strategy is as follows

$$\bar{T}_2(h) = \bar{T}_{w_2}(h) + \hat{T}_2 \qquad (9)$$

where $\hat{T}_2$ is an estimated time delay while sending messages from node w_2 to node q_2.

In the case when link B is used for the transmission of messages residing in buffer B only the average time delay takes the form

$$\bar{T}_2 = \frac{E(N_2)}{\lambda_2} + \hat{T}_2 \qquad (10)$$

In what follows we shall be interested in finding out the average queue length in buffer B that depends on the average break time in the service of buffer B and the average time between breaks. As it results from the considerations in Section 2 the average rate of link B switchings is given by

$$\eta = \lambda_1 P'_{N+p} = \lambda_1 \frac{\varrho^{N+p}(1-\varrho)}{1-\varrho^{p+1}} P_o \qquad (11)$$

and the average time interval between two consecutive switchings is $\frac{1}{\eta}$ sec. Thus the average break time in the service of buffer B is $\frac{\alpha}{\eta}$ sec. and the average time between breaks is equal to $\frac{1-\alpha}{\eta}$ sec. In order to find the average queue length in buffer B averaged with respect to the effects resulting from the switching of the link B we shall implement the approximate analysis to the curve shown in Fig 5.

Within the break in the service of buffer B its queue length will increase on the average by $\lambda_2 \frac{\alpha}{\eta}$ messages and, in normal circumstances its length should drop to the mean value $E(N_2)$ before the next break. We can approximate the section of the curve representing the queue length within the time interval $\frac{1-\alpha}{\eta}$ by the straight line section with a slope of $-(\mu C_2 - \lambda_2)$. Obviously in normal circumstances the following condition must be satisfied

$$\frac{\alpha}{\eta} \frac{\varrho_2}{1-\varrho_2} < \frac{1-\alpha}{\eta} \qquad (12)$$

where $\varrho_2 = \frac{\lambda_2}{\mu C_2}$

and this leads to the inequality

$$\alpha < 1 - \varrho_2 \qquad (13)$$

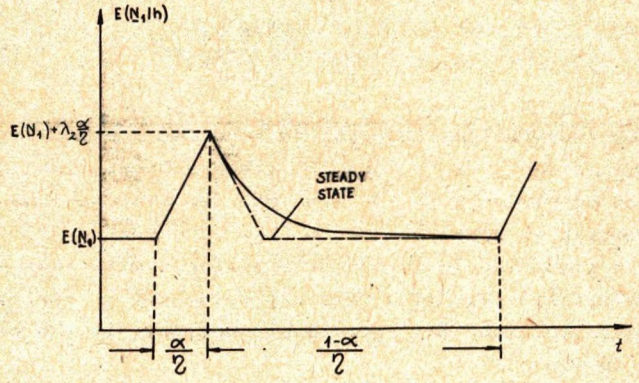


Fig. 5. Average queue length in buffer B.

Now we can comapare the average time delay for messages with and without relay routing strategy. As a criterion of the comparison we can assume the difference between time delays in both cases

$$\bar{T} - \bar{T}^{(h)} = \frac{\lambda_1}{\lambda_1 + \lambda_2}\left(\bar{T}_1 - \bar{T}_1^{(h)}\right) + \frac{\lambda_2}{\lambda_1 + \lambda_2}\left(\bar{T}_2 - \bar{T}_2^{(h)}\right) \qquad (14)$$

Taking into account the formulae (6) and (9) we can convert (14) to the following form

$$(\lambda_1 + \lambda_2)\left(\bar{T} - \bar{T}^{(h)}\right) = E(\underline{N}_1) - E(\underline{N}_1 | h) - \alpha\delta\mu c_2 - \left[E(\underline{N}_2 | h) - E(\underline{N}_2)\right] \quad (15)$$

Prectically we are interested in choosing such N and p for which $\bar{T} - \bar{T}^{(h)}$ is maximum.

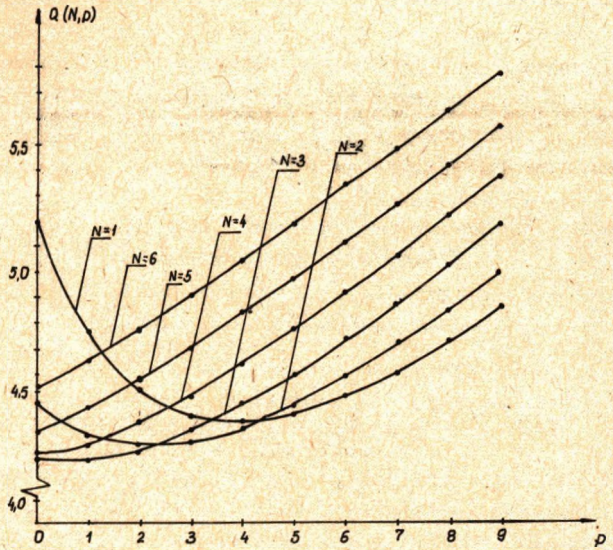


Fig 6 a. $Q(N,p)$ vs. p ; δ = 200 msec., λ_1 = 45/sec., λ_2 = 20/sec., μC = 50/sec., ϱ = 0.9, ϱ_1 = 0.45, ϱ_2 = 0.4

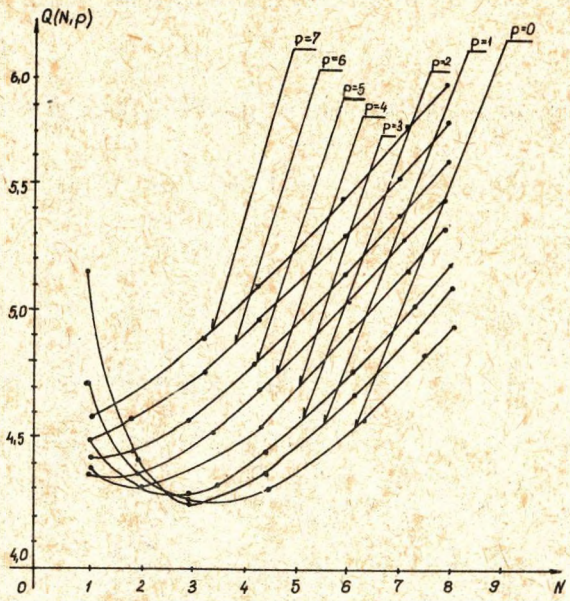


Fig. 6b. $Q(N,p)$ vs. N ; δ = 300 msec., λ_1 = 45/sec., λ_2 = 20/sec., μC =50/sec., ϱ = 0.9 , ϱ_1 = 0.45, ϱ_2 = 0.4

In Fig 6 a,b there are shown 2 families of curves that illustrate the behaviour of the sum

$$Q(N,p) = E(\underline{N}_1|h) + E(\underline{N}_2|h) + \alpha\delta\mu C_2 \qquad (16)$$

It is easy to notice that following condition is fulfilled

$$\max_{N,p}\left(\overline{T} - \overline{T}^{(h)}\right) = \min_{N,p} Q(N,p) \qquad (17)$$

It is worth mentioning that for the typical experimental data encountered in the ARPA network one can reduce the average time delay by about 10-30% while using relay routing strategy together with the strategy based on the periodically updated delay tables.

5. CONCLUSIONS

The concept of a relay routing strategy has been introduced and a model for a relay routing has been developed. The expected message delays with and without relay routing have been presented. It has been demonstrated that the relay routing strategy is an interesting supplement of the routing strategy based on the delay tables.

REFERENCES

1. Brown C.W. - Adaptive Routing in Centralized Computer Schwartz M. Communications Networks with an Application to Processor Allocation, ICC Conference Proc., San Francisco, June, 1975

2. Haisch H.F. - Adaptive Message Switching Networks, Boorstyn R.R Ph.D. Dissertation, Polytechnic Institute of New York, 1976

3. Gallager R.G. - An Optimal Routing Algorithm Using Distributed Computation, Proceedings of the IEEE, January, 1976, pp.73-85

4. Gebhard R.F. - A Queueing Process with Bilevel Hysteretic Service - Rate Control, Nav. Res. log. Quart., Vol. 14, March, 19 67, pp. 55-67

THE DESIGN OF DATA TRANSMISSION SYSTEMS IN THE CZECHOSLOVAK CONDITIONS

Joseph F. Pužman

Federal Ministry of Technology and Investments

Abstract

In the first part of the paper the general method for the data transmission system design is given by means of certain parameters and the procedures yielding the optimal or at least quasioptimal solution. The notion of primary (outer) variables is introduced in order to describe the system from the user's point of view. The data transmission system itself is defined by a set of secondary (inner) variables. The design is then the process of choosing values of secondary variables so that the primary ones might satisfy user's demands.

The second part is devoted to the applications of such a method for the specific Czechoslovak conditions. The properties of telecommunication network, the set of available devices, the P.T.T. tariffs and legal constraints, those are the substantial restrictions the system analyst must take under consideration.

1. INTRODUCTION

This paper discusses the problem of data transmission system design which at present plays an important role in the computer-communication area. Although many approaches have been already proposed and used in real conditions with more or less success the author feel that many white places still exist in that field. Therefore another attempt is suggested which seems to fit the works being done till now.

The purpose of the paper is to outline the main ideas of data transmission design procedure but the scope does not enable to deal more deeply with the methods. On the other hand, however, the design procedure is illustrated by some considerations about the influence of specific Czechoslovak conditions.

In order to give the treatment as comprehensive as possible we start with more general design problem.

From the system theory point of view the problem of analysis and design can be formulated as follows:

Let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of variables taking their values a_i from the finite or infinite sets A_i, $i = 1,2,..,N$. The Cartesian product $\bigtimes_{i=1}^{N} A_i$ is then the set of all vectors $(a_1, a_2, \ldots, a_N)$, $a_i \in A_i$, $i = 1, 2, \ldots, N$. If each system element is described by any set A of the properties of such an element then the vector $(a_1, a_2, \ldots, a_N)$ represents a system S consisting of N elements with properties a_1 through a_N. We shall call X the set of secondary (inner) variables of a system.

From the side of user or observer a system behaves according some rules which can be defined by another set of

variables, say $Y = \{Y_1, Y_2, \ldots, Y_M\}$ depending on X ($Y_j = Y_j(X_1, X_2, \ldots, X_N)$, $j = 1, 2, \ldots, M$). Suppose each Y_j takes its values b_j from the set B_j (finite or infinite). Since there exists a correspondence (in general not one-to-one) between Y and X, the system is represented also by a vector $(b_1, b_2, \ldots, b_M) \in \bigtimes_{j=1}^{M} B_j$. Y can be called the set of primary (outer) variables.

If $S = (a_1, a_2, \ldots, a_N)$ is given the analysis of S is to find the vector $(b_1, b_2, \ldots, b_M)$. On the other hand the designer has to find the vector $(a_1, a_2, \ldots, a_N)$ in order to attain the system properties $(b_1, b_2, \ldots, b_M)$.

2. DEFINITION OF DATA TRANSMISSION SYSTEM

It has been proved by our experience and practice eight classes of primary variables of data transmission system are sufficient [1] :

a) accessibility of hardware and software components, means, specialists, operational room (such variables are discrete as the corresponding sets B are finite);

b) node layout, functions and equipment (e.g. the type of processor in the computer centre, the type of jobs and tasks processed, the structure of data and media);

c) the amount of data emitted from and received in the nodes, and their time distributions (peak hours);

d) readiness measured by the response time or by the delivery time (the mean value, the variance, the x-percentile, etc);

e) accuracy giving mostly by the error rate and secrecy (safeguarding data against a strange interference and

misapplications);

f) reliability (degree of resistence to failures) measured e.g. by the MTBF (mean time between failures) and the MTTR (mean time to repair), by the availability factor (MTBF/(MTBF + MTTR)), by the probability that two network nodes remain connected after a failure occurence, etc.;

g) efficiency of system elements, subsystems and the whole system;

h) total cost which, at present, differs from the costs of local computing systems due to large charges of telecommunication lines and devices.

The list of primary variables shows their heterogenity: some of them acquire the finite number of values (they are discrete), some of them are continuous, the response time, the error rate and the MTBF (MTTR) are random variables in contradistinction to the others which are deterministic.

In the literature the slightly different sets of primary variables has been found [2, 3] but their detailed discussion is out of the scope of our paper.

The number of secondary variables depends on the level of investigations and it is often very large. It is convenient to arrange secondary variables in certain classes describing the parts of system and their properties. Here is an attempt of such a decomposition:

- general secondary variables which involve e.g. the type of system (on-line, vs. off-line, interactive vs. noninteractive, time-sharing, in-plant vs. out-plant, etc.), the transmission mode (SX, HDX, FDX), the total number of network nodes and their layout, the structure of links

(point-to-point, multipoint, ring), the structure of network;

- data communication secondary variables such as the type of links (telegraph, telephone, wide-band, etc., switched vs. leased, private vs. common-carrier, 2-wire vs. 4-wire), the data signalling rate, the data transmission rate, the type of transmission (serial vs. parallel), the interface between data communication equipments and lines, the length of line, the line capacities, their purchase and operational costs;

- communication control secondary variables, e.g. the type of control (centralized vs. decentralized), the switching mode (line, message, packet), alphabets and codes used for transmission, the length of blocks and packets and their formats, methods of error control, flow control, recovery control, capacity assignment control, routeing, adressing, monitoring, the control levels, the distribution of control functions in a network, demands of attended or unattended operation;

- secondary variables characterizing hardware and software components which can be further divide in two groups: common (characteristics for installation such as dimensions, weights, voltage, power, temperature and humidity ranges for technical devices) and specialized concerning specifications, functions and features including the compatibility. As an example see the most comprehensive list of such specifications in Auerbach Data Communications Reports.

Note that some secondary variables may depend on others. Thus, the careful selection of them is necessary in order to obtain a set of independent variables which is prerequisite for further work.

3. DESIGN PROCEDURE

Obviously, there does not exist, in general, a system (a set of values of secondary variables) satisfying the set of values of primary variables. For example, one cannot determine the amount of data to be reliably transferred between two nodes during the prescribed time interval with the fixed cost being less than the necessary cost for such a system. Therefore, additional requirements on primary variables are needed.

In practice, one $Y_i\ (x_1, x_2, \ldots, x_N) \in Y$ is chosen as an objective function subject to the procedure of minimization or maximization. As to the others Y_j, $j \neq i$, the following cases may occur:

a) Y_j takes an arbitrary value from B_j (the influence of Y_j is neglected);

b) Y_j takes an arbitrary value only from the proper subset of B_j, specially $Y_j = y_j \in B_j$ (Y_j is a constraint).

Thus, the objective function may be choosen among the readiness, the accuracy, the reliability, the efficiency and the cost, while the node layout is given by its one value according to the structure of computation or control system. Other examples are discussed in detail in [4].

The analytical solution seems to tend to the methods of mathematical programming. Unfortunately, they cannot be directly applied for the large number of variables and for the fact that some secondary variables are discrete. Moreover, some primary variables are not convex (concave)functions of secondary ones so that even the convex integer programming fails. The only way leading out of the impasse is to decompose the whole system in a proper manner. We try

to sketch such a decomposition.

Suppose the cost is choosen as an objective function (it must be minimized) and the other primary variables become constraints.

The first step consists in the estimation (rather than the evaluation for the lack of necessary knowledge of probabilistic parameters) of data flows between all nodes. It is a routine which can be sometimes excluded from the design procedure if data flows are already learnt from the user's foundations. The good estimation requires only the treatment the results of queueing theory or the application of sophisticated tables [2], graphs and nomographs (see an attempt in [5]).

Knowing all data flows, node layout, accessible telecommunication lines with their cost functions the structure (the topology) of data transmission system may be optimized. For the topology design good heuristic algorithms, giving appropriate results, are proposed. Nevertheless, as we shall see in the next section, some problems remain still unsolved.

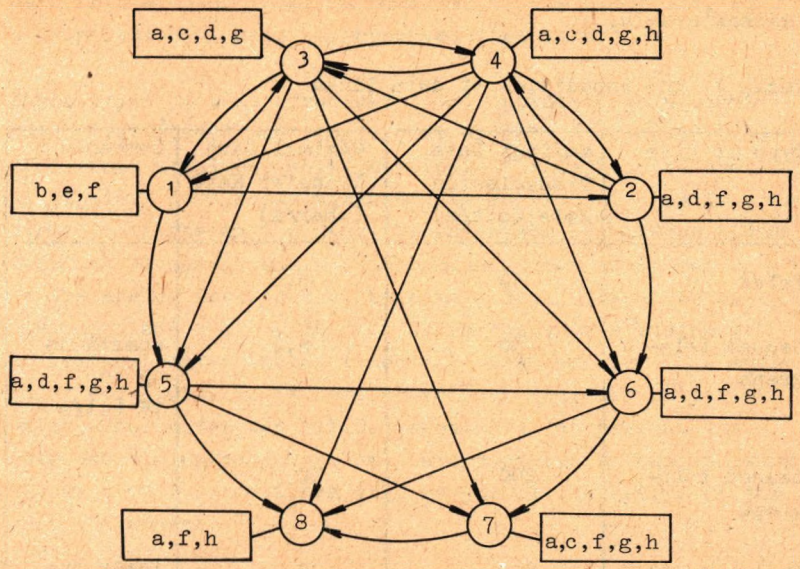
The third step deals with the design of network control. This step is the most complex due to many secondary variables and the lack of explicit dependence between primary and secondary variables. In this field little work has been done in spite of many efforts in the standardization area. A certain attempt has been outlined in [6] where the main functions of control procedures and protocols have been defined but it is unsufficient for the purpose of deeper and more serious design.

The first three steps of proposed design procedure gives us the logical and functional framework of the data transmis-

sion system. The last five steps belong to the technical project and may be solved by degrees. If the additivity assumption holds (the cost is an objective function) the method of dynamic programming seems to be a very useful tool [7]. Besides, the searching procedure, the "man-computer" heuristic programming, the adaptive routeing, those are other methods which fit to the most of practical problems.

For the sake of brevity we only enumerate the last steps without any comments. We begin with the choice of communication means (lines and data sets), then we deal with terminals and communication control units (multiplexors, concentrators, communication controllers, FEP, switching units, etc.) and the choice ends with the telecommunication oriented software. In the last step the total cost has to be evaluate.

The design steps are not only coupled in the forward direction but also they influence backwards, and this fact makes the procedure difficult. If we denote the design steps by numbers 1 through 8 (1 - data flow, 2 - topology, 3 - control, 4 - communication means, 5 - terminals, 6 - control units, 7 - software, 8 - cost) the directed graph with eight vertices could serve as the telling model of main relations between design steps (Fig. 1). For the completeness each node of a graph is weighted by the letters indicating the influence of primary variables on particular steps (see Section 2).



4. DISCUSSION OF DESIGN IN REAL CONDITIONS

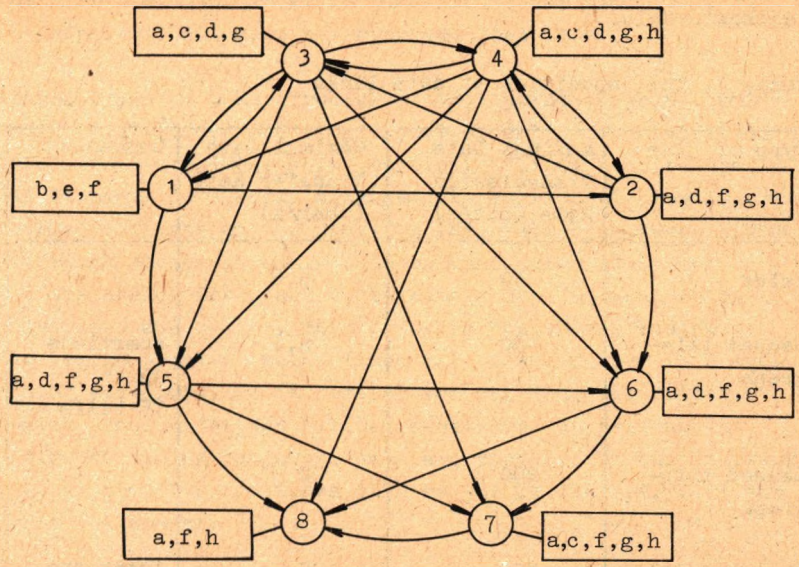
In this section we try to confrontate the design procedure mentioned above with real conditions, in particular, in the Czechoslovakia. We shall pass through design steps and explain the problems arising due to the limited accessibility.

The determination of data flows is out of any constraints. Nevertheless, it is desirable to take under consideration the possibilities of telecommunication network in order to amend the user´s demands (particularly on readiness). The accessibility of data transmission rates which has no doubt been mentioned at this step (see Fig. 1) plays, however, an important role after the transition to the next step. Table 1 summarizes the possibilities of data transmission in the

sion system. The last five steps belong to the technical project and may be solved by degrees. If the additivity assumption holds (the cost is an objective function) the method of dynamic programming seems to be a very useful tool [7]. Besides, the searching procedure, the "man-computer" heuristic programming, the adaptive routeing, those are other methods which fit to the most of practical problems.

For the sake of brevity we only enumerate the last steps without any comments. We begin with the choice of communication means (lines and data sets), then we deal with terminals and communication control units (multiplexors, concentrators, communication controllers, FEP, switching units, etc.) and the choice ends with the telecommunication oriented software. In the last step the total cost has to be evaluate.

The design steps are not only coupled in the forward direction but also they influence backwards, and this fact makes the procedure difficult. If we denote the design steps by numbers 1 through 8 (1 - data flow, 2 - topology, 3 - control, 4 - communication means, 5 - terminals, 6 - control units, 7 - software, 8 - cost) the directed graph with eight vertices could serve as the telling model of main relations between design steps (Fig. 1). For the completeness each node of a graph is weighted by the letters indicating the influence of primary variables on particular steps (see Section 2).



4. DISCUSSION OF DESIGN IN REAL CONDITIONS

In this section we try to confrontate the design procedure mentioned above with real conditions, in particular, in the Czechoslovakia. We shall pass through design steps and explain the problems arising due to the limited accessibility.

The determination of data flows is out of any constraints. Nevertheless, it is desirable to take under consideration the possibilities of telecommunication network in order to amend the user's demands (particularly on readiness). The accessibility of data transmission rates which has no doubt been mentioned at this step (see Fig. 1) plays, however, an important role after the transition to the next step. Table 1 summarizes the possibilities of data transmission in the

Czechoslovakia.

Table 1. The capacities of data links

Type of Line	Maximal Data Transmission Rate (bit/s)	Maximal Data Transfer Rate (char/s)	Comments
Telex	50	6,6	
Leased telegraph	50	6,6	sometimes up to 75 bit/s
Leased telegraph	200	20	
Switched telephone	600	25	lower quality
Switched telephone	80	20	parallel transmission
Switched telephone	1200	150	standard
Leased telephone	9600	1000	higher quality (conditioned)
Leased wideband	72000	9000	
Metalic	9600	1000	short (in-plant, local)

The policy of the Czechoslovak P.T.T. allows, however, all data transmission rates recommended by the C.C.I.T.T. if appropriate data sets are approved (see later).

Three types of line structure exist: point-to-point, multipoint (multidrop) and ring circuits. Point-to-point lines are always available while multipoint lines require special terminal or circuit arrangement. Multipoint line is constructed either on the discrete interface I_2 between a data terminal equipment and a data communication equipment or on the analog interface I_1 between a data communication equipment and a line. The former is the user's business: he leases point-to-point lines and arranges them in a multipoint structure. It is less economical than the latter which comes under the P.T.T. competence; at present the corresponding arrangement sets are prepared by our industry and such multipoint lines will be put by the P.T.T. at the user's disposal.

Ring circuits in spite of their efficiency, are limited to special purposes such as in-plant computer and terminal networks. In addition to it, the lack of convenient data sets precludes their wider applications.

Data networks (except of ring type) are based either on point-to-point or multipoint lines. The star network, the multilevel star network with multiplexors, concentrators and terminal control units at lower level centers, and distributed networks - those are the networks belonging to the point-to-point type. The general tree topology terminal networks form another structure. Both types are realizable if all lines needed between certain nodes are available.

The topology design is immediately followed by the design of network control. First let us touch the error control function.

The design of error control includes the determination of block lengths, format, error correction and/or detection code, recovery, etc. As an example Table 2 shows two solutions used in the batch off-line terminals ZPD 200 and ZPD 1200 produced by our industry.

Table 2. Comparison of two error detecting codes

Equipment	Generating Polynomial over GF(2)	Burst error detection capability
ZPD 200	$X^{12}+ X^{7}+ X + 1$	all bursts of the length 12, all double bursts of the total length 7, 99,951% of the length 13, 99,976% longer bursts
ZPD 1200	$X^{16}+ X^{12}+ X^{5}+ 1$	all bursts of the length 16, 99,9968% of the length 17, 99,9984% longer bursts

Note that while ZPD 1200 uses the well known detecting scheme (see the C.C.I.T.T. recommendation V 41 or ISO standard 3309), ZPD 200 is designated for the data transmission over noisy switched lines with a great occurrence of error bursts so that the block length has to be shortened and more efficient code has to be found.

The control on the data link level is practicaly reduced to BSC and HDLC procedures (perhaps with slight modifications). At present, no terminal device is on up-to-date HDLC basis. The newest VDU terminal EC 7925 being developed this year is BSC oriented because of the EC telecommunication software. HDLC seems to be more attractive in the connection with computer networking.

Common carriers mostly serve for data transmission purposes even their primary assignment has been in the field of telegraph and telephone message transfer. Private lines, in spite of their looser rules of usage form a negligible portion, and common carriers are, and will remain, the main communication mean. This fact considerably restricts the freedom of system analysts who must comply with the P.T.T.'s policy.

In the Czechoslovakia the following rules of P.T.T. must be taken under the consideration. If a user wish to transmit data over the public network (telex or switched telephone) he gets a data link (including data sets). If higher transmission rate is necessary (Table 1) the P.T.T. leases either data links (if appropriate data sets are available) or permits to connect to lines user's data sets being approved in advance. In the future, however, only data links will be leased.

A survey of data sets produced by our industry is in Table 3. For the sake of completeness note that all devices work in the temperature range 5 - 40°C and all use 220V + 10% - 15%/50 Hz ± 2% (except of VPM 020 which is supplied from a telephone line).

If other equipments (terminals, concentrators, control units) are directly coupled with data sets being in possession of the P.T.T. their interface I_2 also is under the control of P.T.T.

At the end of this section we mention the cost function. Almost all design procedures assume that the cost of lines as a function of distance and time of employment is linear. The tariff policy, however, tends to stepwise functions which sometimes prefers some devices to another ones. Higher efficiency of data transmission comparing with telegraph

Table 3. Technical characteristics of data sets manufactured in the Czechoslovakia

Model	Transmission rate	Transmission	Mode	Line	Recommend. C.C.I.T.T.	Dimensions (mm)	Weight (kg)	Power (VA)
MDS 200 (EC 8002)	up to 200 bit/s	serial	SX,HDX FDX	telephone	V21,V24 V28	430 x 275 x 162	11,0	15
MDS 1200/ C,D (EC 8006)	up to 600/ 1200 bit/s	serial	SX	telephone	V23,V24 V28	430 x 275 x 162	10,5	29
MDS 1200/ F (EC 8006)	up to 600/ 1200 bit/s	serial	SX,HDX FDX	telephone	V23,V24 V28	430 x 275 x 162	10,5	29
TMS 200 (EC 8032)	up to 200 bit/s	serial	SX,HDX FDX	telegraph	S15,S16 V24,V28	430 x 275 x 162	10,4	50
VPM 020 (EC 8025)	up to 20 char/s	parallel	SX	telephone	V19	312 x 268 x 116	2,0	0,5
PPM 020 (EC 8025)	up to 20 char/s	parallel	SX	telephone	V19	430 x 275 x 162	10,5	22

and telephone transmissions will definitely lead to changes in this field.

5. CONCLUSIONS

If we compare the preceding section with the theory of data transmission system design touched in first sections we may see the problems which arise in the confrontation of the theory and the practice. While the theory gives us clear results, the practical applications require more simple tools even to the detriment of less exactness. In fact, if the cost of system being designed intuitively is higher by one per cent than the system which we can obtain by means of exact but more complex procedure, then one prefers the former. This is true only for simple data transmission systems (e.g. for the star network with several point-to-point data links). The future belongs, however, to larger terminal and computer networks where the one per cent saving may reach hunderds crowns a month and then the exact design procedure is justifiable.

Nevertheless, system analysts and users ask for a handbook without any formulas and theoretical considerations. They are not interested in the validity of assumptions, they need tables and nomographs for the quick and simple estimation of necessary values. For such purposes Martin's book [2] may excelently serve.

Different conditions in individual countries force to have specific methods which can stem from the unique methodology but which should respect the accessibility of devices and the P.T.T. policy [8].

6. REFERENCES

[1] Pužman, J. (1975), Planning of Communication Network for Data Transmission (in Russian), Proc. of the Symp. SVYAZ-75, Moscow, May 22 - June 5

[2] Martin, J. (1972), System Analysis for Data Transmission, Englewood Cliffs, Prentice-Hall

[3] Rét, A. (1977), On the Design of Remote Data Processing System, Proc. Natl. Conf., Riga, April 25 - 29

[4] Pořízek, R., Pužman, J. (1974), Variables and Constraints in Data Communication Systems Design, Proc. IIASA Conf. on Comp. Comm. Networks, Laxenburg, pp. 167 - 171

[5] Thananitayaudom, T. (1977), Communication Systems - Analysis and Design Using Nomographs, Comp. Networks 1, pp. 147 - 154

[6] Pužman, J. (1977), Functions of Procedures and Protocols, Working Paper of the First Winter School, Karpacz January 18 - 21

[7] Pozin, I. L., Shcherbo, V. K. (1976), Remote Data Processing in Automated Systems (in Russian), Statistika, Moscow

[8] Hellmann, Z., Pužman, J. (1977), The Data Transmission System Design (in Czech), appears as the enclosure of Automatizace starting from January

OPTIMISATION OF DYNAMIC MULTICOMMODITY FLOWS IN COMPUTER NETWORK

Wojciech Molisz

Technical University of Gdańsk, Poland

ABSTRACT

A model is proposed for dynamic multicommodity flows in store-and-forward computer network in which the topology and channel capacities are fixed. Expected flow requirements between all pairs of nodes which maybe time varying are assumed to be known. Two kinds of functionals playing the role of the optimisation criteria are considered. One represents the total cost of network operation and includes memory cost and transmission cost. The other one reflects the loss of value of information, which is assumed an increasing function of the delay.

For given cost functions the dynamic multicommodity flow problem is formulated. Several approaches for solving this problem, including the Pontryagin's minimum principle and Bellman's dynamic programming are considered.

1. INTRODUCTION.

The concept of multicommodity flows in the context of store-and-forward /s-f/ networks is not new, but the previous papers were devoted only to the static cases. Recently Segall [1] introduced the model of dynamic flows in data-communication networks. However some basic difficulties in constructing a realistic model of the optimal control of s-f communication network still exist. It seems that successively improved dynamic flow models will finally lead to the well formalized stochastic optimal control models.

In this paper we have in general the same point of view as in [1], in details. however, there are some differences. The basic assumptions made in the paper are the following. Given:

1/ The reliable, fixed topology network of m nodes and n links /simplex channels/.

2/ An m.m matrix, called the requirement matrix, whose entries $r_{pq}(t)$ are nonnegative functions of time over the period $<t_0,t_1>$.

3/ Finite capacities of links and buffes in nodes /details in Section 2/.

The assumption (2) corresponds to one out of the two situations. In the first one we suppose that the future flow requirements $r_{pq}(t)$ are known in advance /eg. they are scheduled in some way/. In the second, the statistically expected flows in the time period $<t_0,t_1>$ are assumed to be given.

Our task here is to construct and solve the dynamic multicommodity optimal control flow problem. In Section 2 we introduce the dynamic flow models, in Section 3 we state the optimisation problems, while in Section 4 we discuss the solution methods.

2. THE DYNAMIC FLOW MODELS.

The node structure is shown in Fig.1. Each node works in the well known manner, typical for s-f networks and we'll only sketch it. All messages coming from other nodes and originating in the node are directed to the processor buffer /we are not interested in the traffic between nodes and user terminals/.

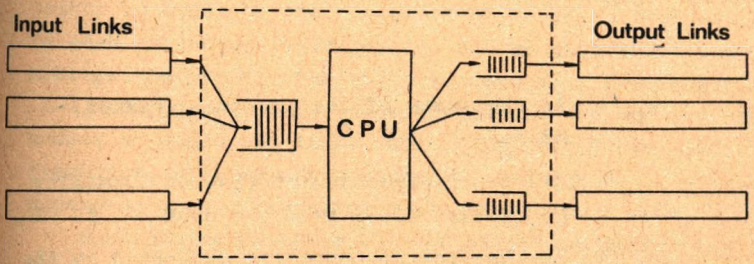


Fig.1. The node structure.

After its header processing the message is passed to the suitable output link buffer. The duration of this transfer is negligible compared to the transmission time outside the nodes. Since the transfers of messages between buffers within the node are random in nature it will be convenient to introduce the equivalent continuons streams of bits giving the same result as the real transfers.

We shall write now the flow conservation equations at nodes and then formulate the state equations.

2.1. The continuous time state equations.

The process of processor buffer loading is described

by the equation:

$$x_{lo}(t) = x_{lo}(t_o) + \int_{t_o}^{t} \Big[\sum_{i=1}^{m} \sum_{\substack{p=1 \\ p\neq l}}^{m} \sum_{\substack{q=1 \\ q\neq p}}^{m} u_{pq}(i,l;\tau) + \sum_{\substack{q=1 \\ q\neq l}}^{m} r_{lq}(\tau) \Big] d\tau \qquad (1)$$

where: $t_o \leqslant t \leqslant t_1$

$l = 1,\ldots,m$

$x_{lo}(t)$ is the state of the processor buffer of node l,

$u_{pq}(i,l;t)$ is the fraction of flow of commodity (p,q) that flows from the node i to l,

$r_{lq}(t)$ is the flow of commodity generated in the node l with destination to node q.

Let us denote by $\mu_{pq}(l,j;t)$ the value of an equivalent stream of bits of commodity (p,q) that flows from the processor buffer to the j-th output link buffer of the l-th node during the period of time $\langle t_o, t\rangle$. The number of bits removed from the processor buffer since the moment t_o is given by the integral:

$$\int_{t_o}^{t} \sum_{j=1}^{m} \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq l}}^{m} \mu_{pq}(l,j,\tau)\, d\tau \qquad (2)$$

Subtracting (2) from (1) we obtain the flow conservation equations for processor buffers.

Similarly as above we can obtain the flow conservation equations for output link buffers:

$$x_{1j}(t) = x_{1j}(t_o) - \int_{t_o}^{t} \left\{ \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq 1}}^{m} \left[u_{pq}(1,j;\tau) + \right.\right.$$

$$\left.\left. - \mathcal{U}_{pq}(1,j;\tau) \right] \right\} d\tau \qquad (3)$$

As it is usually done in the control theory we write down the state equations. The number of components of the state vector is equal to the number of all buffers. The value of each component describes the actual content of the apriopriate buffer. The state vector $\vec{x}(t)$ has exactly $(n+m)$ components. That value corresponds to the sum of number of nodes and links. Multicommodity flows, and their fractions, will be named controls, because their values depending on our decisions will change the states.

Subtracting (2) from (1) and differentiating the result we obtain the state equations for processor buffers:

$$\dot{x}_{1o}(t) = \sum_{i=1}^{m} \sum_{\substack{p=1 \\ p\neq 1}}^{m} \sum_{\substack{q=1 \\ q\neq p}}^{m} u_{pq}(i,1;t) +$$

$$- \sum_{j=1}^{m} \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq 1}}^{m} \mathcal{U}_{pq}(1,j;t) + \sum_{\substack{q=1 \\ q\neq 1}}^{m} r_{1q}(t) \qquad (4)$$

$$1 = 1,\ldots,m$$

Similarly, differentiating (3) we obtain the state equations for output link buffers:

$$\dot{x}_{lj}(t) = -\sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq l}}^{m} u_{pq}(l,j;t) + \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq l}}^{m} \mu_{pq}(l,j;t) \quad (5)$$

$$j = 1,\ldots,m \; ; \quad l = 1,\ldots,m \; ; \quad j \neq l$$

Now we can observe that the rates of change of the state vector components do not depend on the states. This is evident as buffers contents depend only on loading and unloading processes i.e. controls.

Two conditions must be considered: finite capacity condition:

$$x_{lj}(t) \leqslant d_{lj} \quad (6)$$

$$l = 1,\ldots,m \; ; \; j = 0,\ldots,m \; ; \; t_o \leqslant t \leqslant t_1 ; \; j \neq l$$

and initial state condition:

$$x_{lj}(t_o) = a_{lj} \quad (7)$$

$$l = 1,\ldots,m \; ; \; j = 0,\ldots,m \; ; \quad j \neq l$$

where a_{lj} is the initial content of buffer.

In addition to this nonnegativity conditions for all states and controls and finite link capacity conditions are required:

$$\sum_{p=1}^{m} \sum_{\substack{q=1 \\ q=p}}^{m} u_{pq}(i,j;t) \leqslant c_{ij} \quad (8)$$

for all links (i,j) and time $t \in \langle t_o, t_1 \rangle$

2.2. The discrete time state equations.

On the basis of equations (4) and (5) we can write down the state equations for discrete time case. Let us divide the time interval into K periods of length T. Then:

$$x_{lo}(k+1) = x_{lo}(k) + T \sum_{i=1}^{m} \sum_{\substack{p=1 \\ p\neq l}}^{m} \sum_{\substack{q=1 \\ q\neq p}}^{m} u_{pq}(i,l;k) +$$

$$+ \sum_{\substack{q=1 \\ q\neq l}}^{m} r_{lq}(k) - \sum_{j=1}^{m} \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq l}}^{m} \mu_{pq}(l,j;k) \qquad (9)$$

$$l = 1,\ldots,m \; ; \quad k = 0,\ldots,K-1$$

$$x_{lj}(k+1) = x_{lj}(k) - T \sum_{\substack{p=1 \\ p\neq q}}^{m} \sum_{\substack{q=1 \\ q\neq l}}^{m} \left[u_{pq}(l,j;k) + \right.$$

$$\left. - \mu_{pq}(l,j;k) \right] \qquad (10)$$

$$j = ,\ldots,m \; ; \quad l = 1,\ldots,m \; ; \quad j \neq l \; ;$$

$$k = 0,\ldots,K-1$$

In all contraints (6),(8) one must replace t by k, where k is the number of the actual period. Of course, nonegativity conditions must hold. The initial conditions now become:

$$x_{1j}(0) = a_{1j} \; ; \quad j = 0,\ldots,m \; ; \; 1 = 1,\ldots,m \; ; \; j \neq 1 \quad (11)$$

2.3. Final states.

Final states can be fixed or not. Fixed final state conditions exist in one of the following forms:

$$x_{1j}(t_1) \equiv 0 \quad (12)$$

for all 1,j

$$x_{1j}(t_1) \leq \varepsilon \quad (13)$$

where ε is a given small number.

Condition (12) means that all flow requirements are satisfied, while condition (13) allows a little fraction of traffic to be stored in intermediate nodes.

3. THE FLOW CONTROL OPTIMISATION PROBLEMS.

Before formulating the optimisation problems we shall introduce the optimisation creteria. Delay is one of the most important factors in computer network optimisation and the practical optimisation creterion must take it into account.

3.1. The optimisation criteria.

One of the simplest criterion is the cost functional. If α denotes the memory utilization cost per bit and unit of time /in the j-th buffer of the 1-th node/ and γ_{ij} - the transmission cost of one bit through the link /i,j/ then the total network utilisation costs during the period of time $\langle t_0, t_1 \rangle$ are:

$$J\left[\vec{x}(\cdot),\ \vec{u}(\cdot)\right] = \int_{t_o}^{t_1} \left\{ \sum_{l=1}^{m} \sum_{j=0}^{m} \alpha_{lj} x_{lj}(\tau) + \right.$$

$$\left. + \sum_{i=j}^{m} \sum_{j=1}^{m} \gamma_{ij} \left[\sum_{p=1}^{m} \sum_{q=1}^{m} u_{pq}(i,j;\tau) \right] \right\} d\tau \qquad (14)$$

The functional (14) reflects the operational costs explicitly. But there are delays considered implicitly too. If there are b bits during the time interval $<t_o, t>$ in the buffer B, then the integral:

$$\int_{t_o}^{t} b\, d\tau$$

is proportional to the time spent by the message consisting of b bits in the buffer. Minimisation of the first component of (14) will lead to the minimisation of delay suffered by messages. The second component in (14) will help to avoid closed loop flows.

The other criterion deals with "the concept of value of information". We assume that the nonincreasing functions of the value of information depending on delays are given. Two typical examples of such functions are shown in Fig.2. The loss of the value of information we define as the difference between its maximal and actual value. In this case the optimisation criterion plays the role of the total losses and is defined as follows:

$$J\left[\vec{x}(\cdot)\right] = \int_{t_o}^{t_1} \left[\sum_{l=1}^{m} \sum_{j=0}^{m} \psi_{lj}(\tau)\ x_{lj}(\tau) \right] d\tau \qquad (15)$$

where $\Psi_{1j}(t) = \varphi_{1j}(0) - \varphi_{1j}(t)$ (16)

$$1 = 1,\ldots,m \; ; \; j = 0,\ldots,m \; ; \; j \neq 1$$

Minimisation of (15) also leads to delay minimisation.

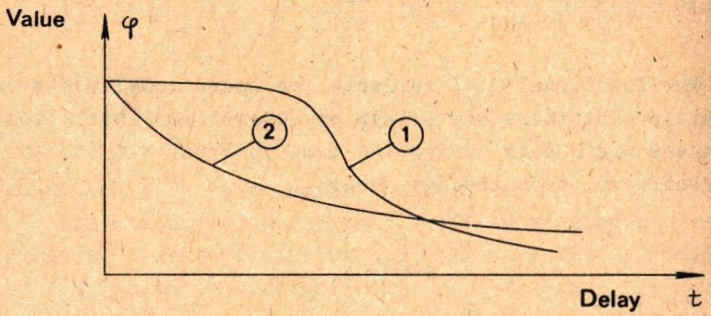


Fig.2. The value of information as a function of delay

Another criterion, is the interval of time during which all requirements are satisfied. Assuming that for some time $t' < t_1$ all source flow intensities become zero we must send all waiting messages to their destinations at the shortest time. The time t_1 cannot be fixed and all states must satisfy final conditions (12). Now we shall rearrange the state equations and constraints to make the notation more clear and to obtain the expressions in a compact form. In the vector-matrix version the state equations (4),(5), become:

$$\dot{\vec{x}}(t) = B\vec{u}(t) + R\vec{r}(t) \qquad (17)$$

where the components of the vector of controls $\vec{u}(t)$ represent all fractions of all commodities flowing in existing links as well as all inter-buffer transfers:

$$\vec{u}^{t}(t) = \left[u_{12}(1,j;t), \ldots, u_{m-1,m}(i,m;t), \mu_{12}(1,j;t), \ldots, \mu_{m-1,m}(i,m;t) \right]$$

and $\mu_{pq}(1,j;t)$ denotes an equivalent stream of bits of commodity (p,q) that flows from the processor buffer to the j-th link buffer in the 1-th node during the period $\langle t_o, t \rangle$, due to the transfers of messages just processed.

Equation (17) describes the dynamics of flows. We clearly have the capacity constraints:

$$\vec{x}(t) \leq \vec{d} \tag{18}$$

and :

$$\Gamma \vec{u}(t) \leq \vec{c} \tag{19}$$

where Γ is the operator that integrates the link flows; and nonnegativity constraints:

$$\vec{u}(t) \geqslant \vec{0} \; ; \quad \vec{x}(t) \geqslant \vec{0} \; ; \quad \vec{r}(t) \geqslant \vec{0} \tag{20}$$

Finally we must take initial and final state conditions into account:

$$\vec{x}(t_o) = \vec{a} \; ; \tag{21}$$

$$\vec{x}(t) = \vec{0} \; ; \tag{22}$$

3.2. <u>The optimisation problems.</u>

The first optimisation problem that can be stated here is the minimum cost control:

Find the set of controls $\vec{u}(t)$, for $t \in \langle t_o, t_1 \rangle$ that minimizes the functional (14) subject to the state equations (17) and control and state constraints (18)÷(21).

We can easily obtain the goal control problem, from the above one by simply adding the final state condition (22)

to the previous set of constraints.

In the optimisation problem of the third kind we shall try to minimise the total losses of value of information. It means that the functional (15) will play the role of the optimisation criterion, while flow dynamics and control and state constraints will be the same as in the first problem.

Finally we shall pose the time-optimal control problem. In this case we shall, have to find the set of controls $\vec{u}(\tau)$; $\tau \in \langle t_o, t \rangle$ that will bring the state from $\vec{x}(t_o) = \vec{a}$ to $\vec{x}(t) = \vec{0}$ while minimizing the duration of this operation, subject to (17) ÷ (22). From the users point of view it means that all messages will be sent from sources to destinations possibly in the shortest period of time $(t - t_o)$.

4. SOLUTION METHODS.

There are various ways in which the control problems stated in Section 3 can be attacked. Size limit of this paper does not allow to discuss them all in details, and we shall restrict the analysis below to the selected problems.

Consider first the time optimal control problem. As an example of a first possible approach we shall refer to the Pontryagin's minimum principle.

4.1. The minimum principle.

We introduce the new variable:

$$x_{m+n+1}(t) = t - t_o \tag{23}$$

$$\dot{x}_{m+n+1} = 1 \tag{24}$$

Then we write Hamiltonian:

$$H(\vec{x},\vec{u},\vec{p},\vec{\lambda},\vec{\xi}) = 1 + \vec{\lambda}^t(t)\left[\vec{d}-\vec{x}(t)\right] + \vec{\xi}^t(t)\vec{x}(t) +$$

$$+ \vec{p}^t(t)\left[B\vec{u}(t) + R\vec{r}(t)\right] \qquad (25)$$

Where $\vec{p}$ is the vector of costates and $\vec{\lambda}, \vec{\xi}$ are vector-function Lagrange multipliers, superscript t denotes transposition.

Optimal controls must satisfy the canonical equations:

$$\left.\begin{aligned} \dot{\vec{x}}(t) &= B\vec{u}(t) + R\vec{r}(t) \\ \dot{\vec{p}}(t) &= \vec{\lambda}(t) - \vec{\xi}(t) \end{aligned}\right\} \qquad (26)$$

The remaining necessary conditions due to control and state constraints are:

$$\left.\begin{aligned} \vec{\lambda}^t(t)\left[\vec{d} - \vec{x}(t)\right] &= 0 \\ \vec{\lambda}(t) &\leq \vec{0} \\ \vec{\xi}^t(t)\vec{x}(t) &= 0 \\ \vec{\xi}(t) &\leq \vec{0} \end{aligned}\right\} \qquad (27)$$

Note that the final values of costates cannot be equal to zero. Aditionally we have constraints (18)÷(22). As well known /see e.g.[2]/, the optimal control is of bang-bang type. Because of the nonnegativity conditions the optimal controls take the form:

$$u^*_{pq}(i,j;t) = \frac{1}{2}\left[1 - \operatorname{sgn}\left(\Theta^*_{ij}(t)\right)\right] \qquad (28)$$

where $\Theta_{ij}(t)$ is the switching function.

In each moment only one commodity can flow through a link, using its total capacity.

Consider now the first optimisation problem stated in paragraph 3.2. Another solution possibility is described below.

4.2. The dynamic programming.

The discrete, vector-matrix version of the state equations is:

$$\vec{x}(k+1) = \vec{x}(k) + \hat{B}\vec{u}(k) + \hat{R}\vec{r}(k) \tag{29}$$

while the corresponding quality criterion becomes:

$$\hat{J}(\vec{x},\vec{u}) = \sum_{k=0}^{K-1} \left[\vec{\alpha}^t \vec{x}(k+1) + \vec{\gamma}^t \Gamma \vec{u}(k) \right] \tag{30}$$

Let us denote by $F\left[\vec{x}(0)\right]$ the optimal value of the quality criterion. Then:

$$F\left[\vec{x}(0)\right] = \min_{\vec{u}(0),\ldots,\vec{u}(K-1)} \hat{J}(\vec{x},\vec{u})$$

We will apply the dynamic programming approach to solve the optimal control problem. First we find the last sample of the optimal control vector $\vec{u}(K-1)$ that minimizes the cost function at the last stage. Denoting the $(K-1)$-th stage minimum cost function by $f_{K-1}\left[\vec{x}(K-1)\right]$ we can formulate the equatuon:

$$f_{K-1}\left[\vec{x}(K-1)\right] = \min_{\vec{u}(K-1) \in U_{K-1}} \left[\vec{\alpha}^t \vec{x}(K) + \vec{\gamma}^t \Gamma \vec{u}(K-1) \right] =$$

$$= \min_{\vec{u}(K-1) \in U_{K-1}} \left\{ \vec{\alpha}^t \left[\vec{x}(K-1) + \hat{B}\vec{u}(K-1) + \hat{R}\vec{r}(K-1) \right] + \right.$$

$$\left. + \vec{\gamma}^t \Gamma \vec{u}(K-1) \right\} \tag{31}$$

Where U_{K-1} represents nonnegativity conditions, finite buffer and link capacity constraints.

To find the optimal value of $\vec{u}^*(K-1)$ we must solve the parametric linear programming problem, since in (31) we have linear function of controls and unknown content of buffers at the moment (K-1). The constraints represented by U_{K-1} are linear as well.

Equation (31) is the initial one for the recurrence relation. In general we shall find the optimal semple of the control vector at the beginning of (K-k)stage from the following recurrence relation:

$$f_{K-k}\left[x(K-k)\right] = \min_{u(K-k) \in U_{K-k}} \left\{ \vec{\alpha}^t \vec{x}(K-k) + (\vec{\alpha}^t B + \vec{\gamma}^t \Gamma)\vec{u}(K-k) + \right.$$

$$\left. + \hat{R}\vec{r}(K-k) + f_{K-k+1}\left[\vec{x}(K-k) + \hat{B}\vec{u}(K-k) + \hat{R}\vec{r}(K-k)\right] \right\} \quad (32)$$

with $k = 2,\ldots,K$

In this backward solution we shall obtain the set of controls $\left\{ \vec{u}^*[\vec{x}(0)],\ \vec{u}^*[\vec{x}(1)],\ldots,\ \vec{u}^*[\vec{x}(K-1)] \right\}$ being functions of states: $\vec{x}(0),\ldots,\vec{x}(K-1)$. Since the state $\vec{x}(0)$ is given we shall obtain the numerical values of $\vec{u}(k)$, using the state equation (29) (K-1) times. Optimisation problems were formulated in such way that all individual commodity flows in all links could be distinguished. This permits to construct the time-tables for dispatching all messages at the bit level. Sometimes it may be necessary to form packets from messages if actual values of streams do not allow to transmit them undivided.

REFERENCES..

[1] A.Segall (1977) - The Modeling of Adaptive Routing in Data Communication Betworks, IEEE Trans.COM-25, 85-95.

[2] M.Athans, P.L.Falb (1966) - Optimal Control,McGraw-Hill.

ACCESS TO COMMON CHANNEL IN A PACKET SWITCHING SYSTEM

Krzysztof Pawlikowski

Technical University of Gdańsk, Poland

Abstract

The access to a common channel used for data transmission from a set of terminals to the CPU is analyzed. It is assumed that the terminals are localized at different places along this channel. The packet lengths are fixed and external priorities of packets are introduced. These priorities correspond to some relative ranking of the terminals. Packets are sent step by step from one terminal to another in the CPU direction. The packet is queued at the intermediate terminal if more privileged packets are waiting there for transmission. In this case the packet having the highest priority is sent out from this terminal. The common channel works rhythmically. This rhythm is determined by the flow of the time slots.
The grade of service of a terminal is measured by the number of slots able to provide the CPU with packets originated at this terminal. The stationary probability distribution of that number of slots is calculated and some numerical results are presented.

1. INTRODUCTION

Channels of computer communication are usually common shared by many users. In this paper a communication system consisting of N buffered input terminals $T_1, T_2, \ldots, T_N$ linked by a common time-multiplexing channel to a CPU is analyzed, Fig.1. It is assumed the packet switching technique is applied. The fixed-size packets are sent rhythmically. The channel rhythm is determined by a stream of time slots being generated at the end of channel and moving down from terminal to terminal in the CPU direction, Fig.2. One slot is capable of transmitting a single packet.

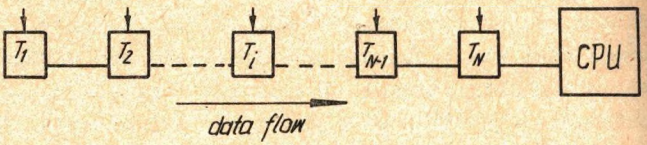


Fig.1. The system analyzed.

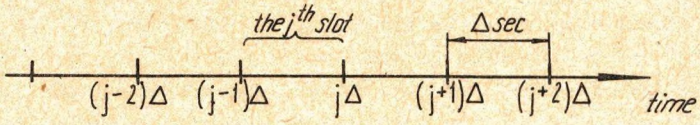


Fig.2. Time sequence of slots observed at a point of the common channel.

We introduce some relative ranking of the terminals. The packets entering the system at the same terminal belong to the same priority class. The packet transmission is interrupted at an intermediate terminal if more privileged packets are waiting there for transmission. The effect of such relative ranking of the terminals on the grade of their service is investigated. We consider the number of slots available for the packets originating from the i^{th} terminal, from the point of view of an observer at the input of the CPU.

We take into account also the case if terminals are linked by two parallel channels added to improve the system's reliability. The presented results can be directly applied to systems with separate communication from terminals to computer and vice versa. Recently the above manner of servicing of terminals has been proposed for loop communication systems by Katz and Konheim [1]. In multidrop networks our analysis admits to investigate a relative ranking of lateral channels along the main channel.

2. MODEL DESCRIPTION

The general model for the analyzed system is shown in Fig.1. As it has been above-mentioned the packet switching technique is used and external priorities of packets are introduced. For simplicity the packet length will be treated as the length unit of data and the slot duration, Δ seconds, will be the time unit. Each slot is capable of transferring a single packet. At each terminal the service operation can take place every Δ seconds and may start only at the beginning of each incomming slot. If two parallel channels are used, two slots arrive at each terminal every Δ seconds.

Let the i^{th} terminal have priority $\Pi(i)$. All packets entering the system at this terminal have the same priority.

Let the packets belonging to the priority class $\Pi(i_1)$ be more privileged than the packets belonging to the priority class $\Pi(i_2)$ if $\Pi(i_1) < \Pi(i_2)$. We assume that the packet occupying the slot will be buffered at intermediate terminals if more privileged packets are waiting there for transmission. The slot is then claimed by the packet having the highest priority. Thus the transmission of a packet from its terminal of entry to the CPU may require several slots.

The arrivals of data for transmission are assumed to be determined by stationary random processes, i.e., the probability of arriving of n packets in the time interval (t_1, t_2) depends only on the difference $(t_2 - t_1)$ and not on the absolute value of t_1. The number of packets which enter the system at the i^{th} terminal during the j^{th} slot will be donoted by $X_i(j)$ and its generating function by

$$G_i(z) = E\left[z^{X_i(j)} \right] \qquad (1)$$

The mean number of packets arriving at terminal T_i during a slot duration will be denoted by EX_i.

Arrival processes at different terminals are considered to be mutually independent. Note that the variables $X_1(j)$, $X_2(j), \ldots, X_N(j)$ actually refer to different time intervals, the difference attributable to the transit time of the slot from terminal to terminal. In what follows all processes at each terminal are analyzed according to the local time of this terminal, i.e., at each terminal the time has been counted since the instant when the first slot arrived.

The numbers of packets buffered at the terminal will be observed only at the instants just before the arrival of a slot. Let $L_{i,k}(j)$ be the number of packets which have entered the system at terminal T_i and wait for transmission at terminal T_k just before the start of the $(j+1)^{st}$ slot, i.e., at the instant $t = j\Delta - 0$ (according to the

local time of terminal T_k). Of course packets originating from terminal T_i can be buffered at terminal T_k if $i < k \leq N$ and $\Pi(k) < \Pi(i)$.

For analitical tractability we assume that terminals have infinite buffering capacities. Our analysis is adequate for terminals with limited buffering capacity also, but only for the low overflow-probability region.

We will consider the number of slots (per time unit) accessible for packets originating from terminal T_i, from the point of view of an observer at the input of the CPU. It is equivalent to the number of slots (per time unit) unoccupied by packets originating from more privileged terminals than T_i. For simplification we will analyze the reciprocal variable - the inaccessible channel period for packets from T_i determined by the sequence of slots occupied by packets having higher priorities than $\Pi(i)$.

3. THEORETICAL ANALYSIS

Let relative ranking of the terminals be described by the sequence $\Pi(1)$, $\Pi(2),\ldots,\Pi(N)$ being a permutation of the integers $1,2,\ldots,N$. The most privileged packets originate from terminal T_{i_1} having the priority $\Pi(i_1) = 1$, afterwards from terminal T_{i_2} having priority $\Pi(i_2) = 2$ and so on. Let us note that if the terminal priorities correspond to their positions along the common channel, i.e., if $\Pi(i) = i$, then the packet occupying a slot is delivered to the CPU without any obstacles. There are no interruptions of transmission. Such priority structure is the simplest one and therefore is frequently used e.g. in the loop communication systems.

As mentioned the state of the common channel is observed at its end, i.e., at the input of the CPU. Let $L_{hi}(j)$ be the total number of packets originating from terminals more privileged than terminal T_i and waiting in the system

for transmission at the instant $t = j\Delta - 0$ (according to the local times of terminals), i.e.,

$$L_{hi}(j) = \sum_{p \in \mathcal{P}_i} \sum_{k=1}^{N} L_{pk}(j) \quad , \tag{2}$$

where $$\mathcal{P}_i = \left\{ p : \Pi(p) < \Pi(i) \right\} \tag{3}$$

is the set of integers indicating the terminals having priorities higher than $\Pi(i)$. Similarly let $X_{hi}(j)$ be the total number of packets arriving during the j^{th} slot at more privileged terminals than T_i, i.e.,

$$X_{hi}(j) = \sum_{k \in \mathcal{P}_i} X_k(j) \tag{4}$$

If terminal T_i has the highest priority, i.e., $\Pi(i) = 1$, then each slot can be claimed by a packet originating at this terminal. In this case $\mathcal{P}_i = \phi$ and $L_{hi}(j) = 0$, $X_{hi}(j) = 0$ for each j. In the sequel we assume that the priority of the ahalyzed terminal is not equal to 1.

The $(j_o + 1)^{st}$ slot is accessible for a packet originating from T_i, i.e., this slot can deliver such packet to the CPU, if

$$L_{hi}(j_o) = 0 \tag{5}$$

For simplicity we assume that $j_o > 0$. If M denotes the number of parallel common channels, then

$$L_{hi}(j_o + 1) = X_{hi}(j_o + 1)$$

$$L_{hi}(j_o + 2) = X_{hi}(j_o + 1) + X_{hi}(j_o + 2) - M$$

$$L_{hi}(j_o + \vartheta) = X_{hi}(j_o + 1) + X_{hi}(j_o + 2) + \ldots + X_{hi}(j_o + \vartheta) - (\vartheta - 1) \cdot M \qquad (6)$$

for $1 \leq \vartheta \leq \tau_i$, where

$$\tau_i = \min_{\vartheta} \left\{ \vartheta : L_{hi}(j_o + \vartheta) < M \right\} \qquad (7)$$

Variable τ_i is the inaccessible channel period for packets originating from T_i, from the point of view of the input of the CPU. The probability distribution of τ_i will be calculated assuming that $E\left[X_{hi}(j)\right] < M$, then $\Pr\left[\tau_i = \infty\right] = 0$. This expresses the natural requirement that packets should not arrive at a higher rate than they can be removed from the system.

Last equation of (6) may be rewritten as

$$L_{hi}(j_o + \vartheta) = L_{hi}(j_o + \vartheta - 1) - \vartheta \cdot M \qquad (8)$$

for $1 \leq \vartheta \leq \tau_i$, under the fictitious initial condition that

$$\Pr\left[L_{hi}(j_o) = M \right] = 1 \qquad (9)$$

Thus τ_i can be interpreted as the first passage time to the states lower than state M in Markov chain developed according to (8). Distributions of τ_i, for any M, have been given in [2].

SINGLE COMMON CHANNEL, M = 1

The generating function of the first passage time to state 0 in Markov chain developed according to (8) has the form

$$E\left[v^{\tau_i}\right] = R_i(v) \qquad (10)$$

where $z = R_i(v)$ is the unique solution of equation

$$z - v \prod_{k \in \mathcal{P}_i} G_k(z) = 0 \qquad (11)$$

which is of modulus less than one for each v, $|v| < 1$. Functions $G_k(z)$ are determined by (1), set $\mathcal{P}_i$ is given by (3). After differentiating of (10) we obtain

$$E[\tau_i] = \left(1 - \sum_{k \in \mathcal{P}_i} EX_k\right)^{-1} \qquad (12)$$

where EX_k denotes $E[X_k(j)]$.

DOUBLE COMMON CHANNEL, M = 2 .

The generating function of the first passage time to state 0 or 1 in Markov chain developed according to (8) has the form

$$E[v^{\tau_i}] = R_{1i}(v) + R_{2i}(v) - R_{1i}(v) \cdot R_{2i}(v) \qquad (13)$$

where $z_1 = R_{1i}(v)$ and $z_2 = R_{2i}(v)$ are the only two roots of equation

$$z^2 - v \prod_{k \in \mathcal{P}_i} G_k(z) = 0 \qquad (14)$$

which are of modulus less than one, $R_{1i}(v) \neq R_{2i}(v)$ for each v, $0 < |v| < 1$, and $R_{2i}(1) < R_{1i}(1) = 1$. The first derivative of (13) at $v=1$ equals

$$E[\tau_i] = \left(2 - \sum_{k \in \mathcal{P}_i} EX_k\right)^{-1} \left(1 - R_{2i}(1)\right) \qquad (15)$$

$E[\tau_i]$ determined by (12) or (15) is the mean inaccessible channel period for packets originating from terminal T_i, i.e., having priority $\Pi(i)$. By Blackwell's theorem [3] the reciprocal of this parameter is equal to the expected number of slots (per time unit) available for packets belonging to the i^{th} priority class, from the point of view of the input of the CPU. Konheim in [4] has proposed to call this expected number of slots the slot availability. We shall denote this parameter by $\bar{S}_i$. Thus

$$\bar{S}_i = \left(E[\tau_i]\right)^{-1} \tag{16}$$

It measures the percentage of slots available for terminal T_i from the point of view of the input of the CPU. According to (12) and (15):

$$\bar{S}_i = \begin{cases} \left(1 - \sum_{k \in \mathcal{P}_i} EX_k\right), & \text{for } M = 1, \\ \left(2 - \sum_{k \in \mathcal{P}_i} EX_k\right)\left(1 - R_{2i}(1)\right)^{-1}, & \text{for } M = 2, \end{cases} \tag{17}$$

where set $\mathcal{P}_i$ is determined by (3).

4. INTERPRETATION OF RESULTS

Introducing the relative ranking of terminals on one hand we must pay a price in more elaborated terminals but on the other hand we are able to improve the performance of the overall system. On the basis of the obtained results we can optimize the system performance with respect to the slot availability.

It can be proved that the sequence of $\{\bar{S}_k\}$ is nonincreasing together with priority decreasing, for every priority structure of terminals. Thus to maximize $\min_k \bar{S}_k$ for fixed

for each i. For this input process the times at which messages arrive are determined by a standard Poisson process with rate λ (messages/slot duration) while the lengths of the messages are geometrically distributed with mean message length $1/(1-q)$ packets, $0 \leq q < 1$. The choice of this particular arrival process was motivated by some measurements of data traffic [5]. All results are given for systems with ten terminals.

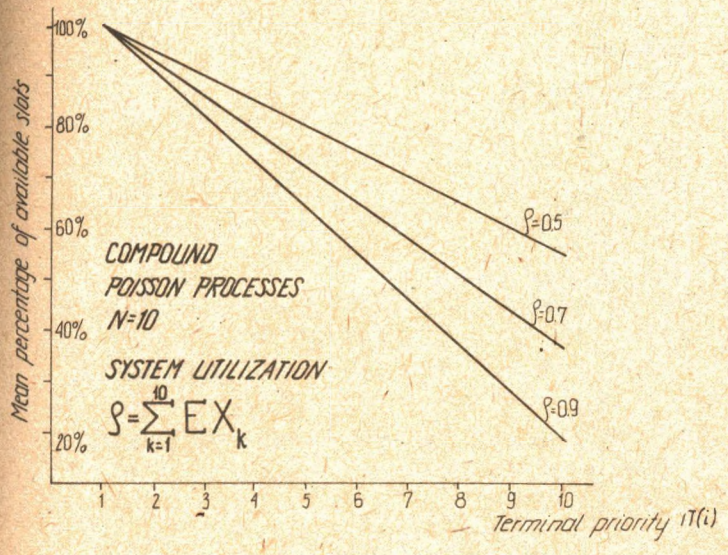


Fig.3. Slot availability as a function of terminal priority. Single common channel, M = 1.

Figure 4 shows that the slot availability for M = 2 depends on the mean message length too. In this case for the same system utilization the slot availability decreases if the mean message length increases. It can be such a strong degradation of service in some priority classes that the slot availability is less than for higher system utilization.

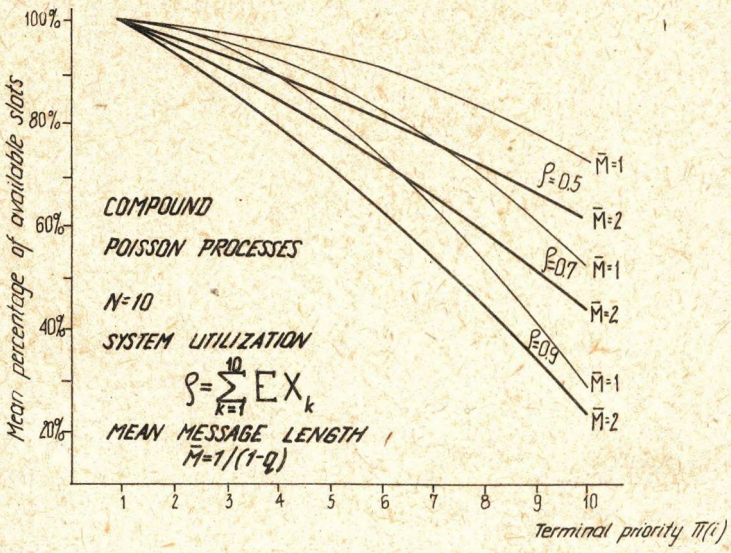


Fig.4. Slot availability as a function of terminal priority. Double common channel, M = 2.

REFERENCES.

1. S.Katz, A.G.Konheim, "Priority disciplines in a loop system", J.ACM, vol.21, nr 2,April 1974, pp.340-349.

2. K.Pawlikowski, "Multiple channel occupation time in packet switching system", Bull.Acad.Pol.Sci.,Ser.Sci. Tech., will be published.

3. L.Takacs, "Introduction to the theory of queues", Oxford University Press, Inc., New York, 1962.

4. A.G.Konheim, "Service epochs in a loop system", Proc. 22 nd Int.Symp.Computer-Communication Networks and Teletraffic, Polytech.Inst.Brooklyn, Brooklyn, N.Y., April 1972, pp.125-143.

5. A.L.Dudick, E.Fuchs, P.E.Jackson, "Data measurements for inquire-response computer communication systems", Proc.IFIP Congress 71, vol.1, Ljubljana, Yugoslavia, 1971, pp.634-641.

THREE YEARS EXPERIENCES IN USING A TELETYPE TERMINAL

Dr. Katalin HOVÁNYI

Petroleum Engineering Research Laboratory of the Hungarian Academy of Sciences

Abstract

The large majority of the problems in practical engineering may be solved using computers. In fact often it is the only possibility. As Computer Science is growing in Hungary, more and more people in Engineering get in touch with a remote processing network.

This paper is concerned with running a remote station of the Central Site of the Hungarian Academy of Sciences where a CDC 3300 type computer has been in service since 1971. The paper deals with the problems of training the users, mostly research workers, with the problems concerning organisation and with providing adequate computing capacity. It gives a short description of how engineering programs with large memory requirements may be handled through the terminal, and relates the accumulated experiences.

1. A short review of the terminal system of CDC 3300 computer

1.1. The structure of the terminal system /Fig.1./

The CDC 3300 Central Computer of Hungarian Academy of Sciences was installed in Budapest early in 1971. It's main task is the support of the scientific research work. The terminal system has been functioning since 1974.
Our Teletype terminal was installed three years ago. This terminal consist of a teletype modem connected to the special DATEX line and a teletype. The transmission speed of this DATEX line is 110 bit/sec.

1.2. RESPOND, PUDDING and FLOWER subsystems and their practicability

The subsystems operate under the CDC 3300 MASTER /Multiple Access Shared Time Executive Routine/ operating system.

1.2.1. RESPOND

RESPOND-EXPORT/IMPORT - REI - is a multiple access subsystem. After linking a Teletype terminal to REI, the user can submit up to 16 commands that provide the capabilities to:
- allocate and open files,
- release unneeded files,
- enter data records into a file,
- modify a file by adding, deleting or replacing records,
- submit a JOB for batch processing.

This was the first working subsystem. It has many problems. For example:
The system has no JOB restart. The work area from the last edited record may be rewritten in an opened file at a system abort. The overhead time and the memory requirements are too large.

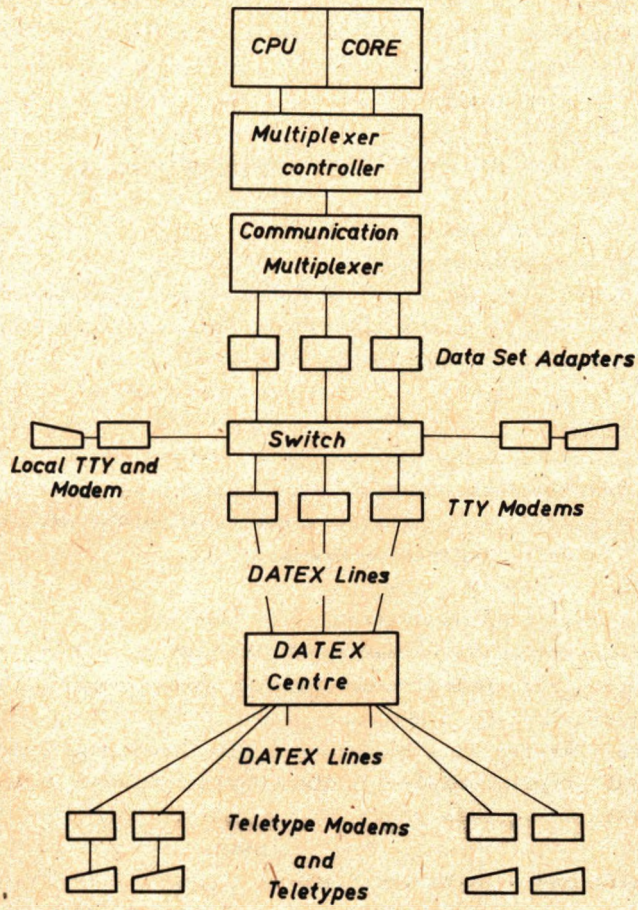


Fig. 1. The Structure of Teletype Terminal System

1.2.2. PUDDING

The new interactive subsystem PUDDING / Program Update Debugging and Developing Interactive Game / has many possibilities.
During a session the users can:
- write one or more records on PUDDING input file,
- modify the records on the input file or another opened
- user file,
- activate MASTER tasks,
- ask the used compute time,
- communicate with the operator of the Central Computer,
- print the output on the teletype or the central printer.

The RESPOND will be replaced by this system.

1.2.3. FLOWER

FLOWER /FORTRAN-like Language for Outstation Work with Error Recovery/ is a little interactive subsystem for training to write FORTRAN programs.
It provides:
- program writing line by line,
- free read and write format to the terminal,
- automatic error recovery during the translation and execution.

We use this system as a calculator. First we used it to teach FORTRAN programming but The PUDDING system is much better for this purpose.

2. Training and working experiences

2.1. Users' training

At a little terminal site the users work directly with the computer. The theoretical and the manual parts of the training are inseparable. We can class the users into types on the basis of their computer knowledge.

For example the user is:

- perfect in programming, knows well the CDC 3300 operating system,
- perfect in programming but has worked on another computer,
- not too pefect in programming, he knows a program language but he hasn't used it yet.

2.1.1. Our training system

At the first type of users the steps of the work:

- short review of the terminal system,
- detailed description of one subsystem,
- 4-6 short practical applications of it,
- training of the terminal handling and activating,
- work with the computer,
- talking over the problems of work,
- writing and running user programs through the known subsystem,
- learning other subsystems and their commands,
- comparing the subsystems with one other.

The second case:

- short explanation of the CDC 3300 operating system,
- parameters of the most important JOB control cards,
- using these cards, 15-20 little examples,
- having a talk with users about their programs and problems,
- learning the terminal system according to the preceding method.

The third type of method:

- practice of terminal handling and activating,
- compilate and execute some little source language programs written by the users through one subsystem,
- learning the used JOB control cards and subsystem sommands,
- writing a full program with the teacher, using the learned cards and commands,

- running this program in little groups - two users make it,
- speaking about the error messages sent by the computer during the sesson,
- learning the remaining subsystem and running the jointly written program through these ones,
- compare discussion.

2.1.2. Our observations

The optimal number of groups is 6-12.
More than 12 people can't work on the same level, one part of group talls away.
The sample programs and user guides must be on hand at the beginning of the course of lectures.
The term-time of the first method is 2 days, at the next two types it is 3 days at least.
According to our experience the manual part of the training was most successful when the users work in pairs on the terminal. More than two people disturb each other in the work. In the course of terminal training every user must deal with manual work.
The new users have to work with the computer two or more months in order to get a good knowledge of the whole system. The users must be informed about this fact because many of them leave off using the system by reason of "great" problems.

2.2. Organization of terminal network

There are 10 Teletype terminals now. These terminals share 3 lines connected to the system multiplexer. The RESPOND subsystem is available from 3 to 5 o'clock p.m.
In first two years we made a weekly program indicating the computer line numbers, the working hours and the stations each days. The experiences of that time help us to make a

permanent time-table. Every station has got a copy from it. To prevent the terminal overloading the Central Site the following restrictions are made:
At the same time one terminal has only 3 JOBs in the computer. If a JOB needs disc or magnetic tape the Central Site operator should be informed by a message in time. The upper limit of compute time allowed, memory available to a JOB is changed dynamically by the operator depending on the workload of the Central Site.

2.3. The organisation of work at our terminal

The JOBs run on our terminal are submitted by our own research institute and three other research groups of the Academy and the University.
On the average we have to run the JOBs of four to eight users within a limited period making as many runs of each Job as possible. This may be mean 15 to 25 program runs at a session. This compared to the workload of a large computing centre means modest performance. A Teletype terminal with its transmission speed of 110 bit/sec makes the input of data and sampling of the output very slow. In order to queue the JOBs and to set them in optimal sequence it is essential for the operator of the terminal to prepare for each session very carefully.
We have to:
- provide data preprocessing,
- run the JOBs,
- keep records of the work done on the terminal,
- give access to the terminal for those users who wish to handle the terminal themselves to correct their JOBs or to lookat their results,
- provide an operator for those who do not want to handle the terminal themselves,
- give software assistance to those who need it.

These tasks are done by two members of our group. To make

data processing easier, our users are free to punch their JOBs themselves if the JOB is not too long. We collect the JOBs to be run before the terminal session starts and establish the running order.
The sequence is set as follows:
Correction JOBs that use little computing time, require no special discs will run first. JOBs that require more resources are submitted next, While these are running the results of the correction JOBs may be looked at and perhaps further modified. Long JOBs are transferred to system files in the end, as transferring 100 records on line takes about 20 minutes from a Teletype station. These JOBs are submitted to be run in most cases only next day.

To ease documentation we use double copy paperrolls. The original copy is given to the user and we keep the second copy for two months. At the end of each session the operator makes an entry for each JOB run in terminal log book and hands over the list to the users. The log book contains the name of the JOB, its owner, the identifier of the input and output system file, the computing time, the number of runs and their sequences. It is the operator who transfers the JOBs to the system files and submits them for running. If requested the users may look at the results in the outfile and perform the appropriate corrections themselves.
The software assistance includes:
- the explanation of the error messages,
- giving information concerning packages of the program library,
- giving courses on new software subsystems.

I would like to mention that our experienced users may use such subsystems as FLOWER or PUDDING even if the operator is not present as these subsystems may be used all day without time contraint.

3. The use of the remote station to do calculations technical nature

3.1. The characteristics of the programs and interconnected programs run from the station

The scope, size and computing power requirement of the JOBs vary as our users come from a wide circle. The large majority of the programs written in FORTRAN and SIMULA need 30-40 KWord of core memory / 24 bit/word /, 4 to 6 thousand lines and 10 to 60 minutes computing time. The fields of research requiring such computing are manifold. Just some examples based on the information given by one of the research groups:

- determine the state of strain and stresses in plates and shells of revolution using finite element method;
- calculation of the natural frequencies of the systems of finite and infinite degree of freedom;
- solution of some symmetric problems of the theory of thermoplasticity;
- calculation of states of strain and stresses of a cylindrical shell in the case of unsymmetrical load.

Our own institute, the Petroleum Engineering Reseach Laboratory of the Hungarian Academy of Sciences use computers to solve problems related to the following subject:

- locating hidrocarbon reserves on the basis of geochemical parameters in space;
- modelling the variation of geological and mining parameters in space;
- modelling problems that arise while producing oil from an oilfield.

I feel that even this short list of the fields of applications demonstrates the enormous possibilities of assisting research work by the use terminals.

The terminal may, of course, be used to run programs for

evaluating measurements data, for editing existing data or program fields or as a handy "desktop" calculator though its primary use should be the solution of problems that require computing power that only centralised large computer systems can offer.

3.2. Experiences in using small editing correction JOBs that handle large programs

Due to the short ON LINE time, the low transmission speed and the large number of users per teletype stations, it is almost impossible to transfer a program longer then 200 lines to Central Site.
The package developed by the software department of the MTA SZTAKI /Institute of Automation and Computer Science of the Hungarian Academy of Sciences/ called DAISY /Debug Aimed Interpretive System/ allows the simple and effective handling of large JOBs.

The package has two main functions:
It provides the users access to their programs or data that were keyed directly to magnetic tape from coding forms and later transferred to a "correction file" allocated on system discs. /See. Fig.2./
It allows the user to change to correct the "correction file" by using the commands of DAISY. /See Fig.3./

DAISY is used as follows:
The programs and data in source language on coding form are posted to the Central Site. When the material is already on the "correction file" the user gets a listing of the file. On it the user marks the records or characters to be corrected. Using the correction language, the corrections are performed. The corrected file is transferred to the user's own permanent file and the "correction file" is released. For further corrections the

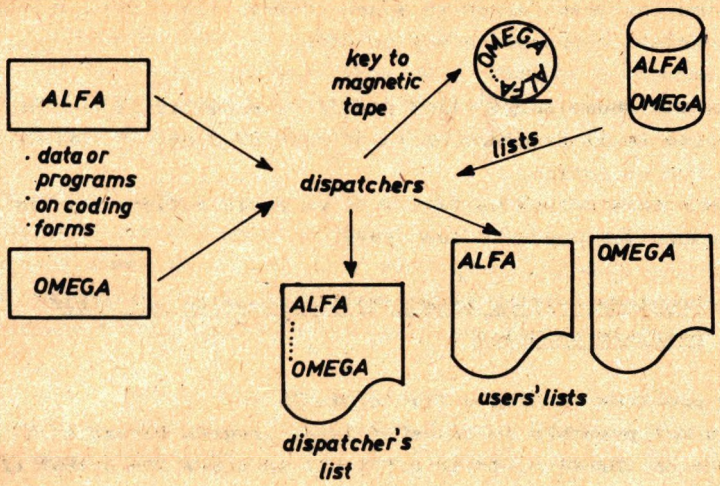


Fig. 2. First function of DAISY

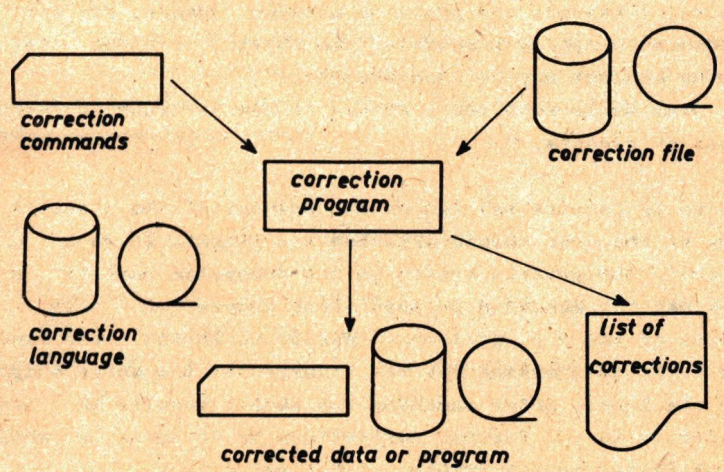


Fig. 3. Second function of DAISY

file may be transferred to a new "correction file" and the appropriate corrections may be done.

To handle these large files from a terminal it is enough to write small programs of 15-30 records long to submit them for execution.
Through this method we feel that we could satisfy most of the computing needs of our users.

3.3. Problems arising from remote processing and their possible solution

Main problems running a TTY terminal:
It is not possible to transfer large program blocks at TTY speeds on the line. It is difficult to check the output of programs producing a large volume of results. If the postal line or the teletype breaks down, the time lost may be considerable. The files stored on system files of the terminal subsystem are prone to accident damage. For the newcomers to remote processing the initial problems they come up against may be considerable.
Part 3.2. deals with the problems of how to transfer large volume of results.

We have to results printed on the printer of the Central Site, or the user should organise his program in such a way that, the program writes on the subsystem outfile a few lines for checking purpose after completing each section of the JOB. It is much easier to locate these lines from a TTY then asking for the transfer of the whole output over the lines. After checking the sample results the user may decide to add further corrections or to have the whole output listed on the printer of Central Site.

Breakdowns may cause several problems, being on a dial-up system, the remote sites are in touch with each other. It often happens that they do little favours to each others either by lending machine time or by running each others' JOBs if it is urgently needed. They even exchange programs or programming tricks through the paper tape reader.

If during data transfer or correction some files are open they are easily erased or damaged if the Central Site breaks down. Therefore it is worth while to make copies of important files. It is much easier to up-date yesterday "standby" copy than to start again from scratch.

It is not practicable to train all sites to recognise rare error messages or to teach every little trick to those new to remote processing. It is much more profitable to ask for advice from the more experienced sites as the problems arise one by one. Sometimes getting together and talking over things in a "user's club" may be very profitable.

I am convinced that all those working in such a network have very much in common and they have many experiences worth exchanging even if some of them work in Medicine while others in Astronomy.

Bibliography

/1/ A.Gyárfás, G. Jónás: DAISY 1.0, Felhasználói ismertetők 8. MTA SZTAKI Budapest, 1974.
/2/ MTA. CDC User Journal 7. MTA SZTAKI Budapest, 1974.