International Institute for
Applied Systems Analysis
IIASA www.iiasa.ac.at

Evolution of Computer Networks: Theory and Experience. Proceedings of the Meeting, December 10-12, 1979

Petrenko, A.

IIASA Collaborative Paper
September 1981

EVOLUTION OF COMPUTER NETWORKS: THEORY AND EXPERIENCE

Proceedings of the Meeting,
December 10 - 12, 1979

A. Petrenko
Editor

September 1981
CP-81-27

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
A-2361 Laxenburg, Austria

PREFACE

This publication contains the papers presented at the meeting on "Evaluation of Computer Networks: Theory and Experience", organized by IIASA's Informatics Task in December, 1979.

In a sense, this meeting marked the conclusion to a series of regular conferences on the theoretical aspects of computer networking held by this group at IIASA. Not only has the Informatics Task now officially become IIASA's Computer Communication Services, but also the focus of IIASA's networking activities has shifted from pure research to practical usage of networking technology for the support of in-house research activities. The communication facilities provided by IIASA's gateway and communication center as a whole have been growing rapidly due to constant implementation and application of new software and hardware tools, and these practical developments are, naturally, based on the experience gained by this group in its research in the field of computer networks. IIASA's computer networking meetings served as the most appropriate forum for gathering and sharing such experiences.

Although the publication of the papers from the above mentioned meeting has been delayed due to certain technical and organizational problems, it is our belief that they will nonetheless be of great interest to those organizations cooperating with IIASA and to the scientific community as a whole.

CONTENTS

ARCHITECTURAL MODELLING OF DATA PROCESSING SYSTEMS

M. Bazewicz

ABSTRACT

The properties of a computer system's architecture can be examined from the point of view of data processing organization, i.e., centralized and distributed organization; utilization mode, i.e., batch and interactive; and communication function, i.e., number and types of protocols, etc. This paper presents a method of system architecture analysis characterized by a model of the user Job Handling Process (JHP). In order to compare the properties of data processing functions and mechanisms in systems with centralized and distributed architectures, JHP and Open Systems Architecture (OSA) models have been used. The typically applicational approach of the JHP model and the layer approach of the OSA model to the analysis of the system architecture permitted the author to propose an additional layer in the OSA mode. This layer defines the interfaces of "job preparation" mechanisms initiated by the open system user.

INTRODUCTION

Modern computer and communication technologies have made possible the interactive use of computer resources, independent of their degree of distribution. Interactive and batch mode can

be realized by the user in different ways depending on the organization of data processing and the architecture of the computer system. The following two basic methods are known to be used in interactive and batch users' job handling:

1. Centralization of resources and distribution of access by means of a network of terminals co-operating with a central multi-access processor - a system known as the centralized data processing system (CPS),
2. Decentralization of resources and distribution of access - a system known as the distributed data processing system (DPS), or as a network consisting of subscriber (HOST) computers and communication nodes.

The first of the above mentioned methods is characterized by the use of a time-sharing technique in the users' job service, and by resource management in the central processor. The handling of application jobs is the primary function of the system controlling both the communication between the user and the resources, and the service of the requested job stream.

The second method uses a time-sharing technique both in the resource management of the network's HOST computers and in the communication processes serving the distributed resources of the computers. In the analysis of the DPS, special importance is thus attached to communication problems.

The experience gained in servicing different application jobs, particularly in the existing heterogeneous computer networks (DPS), and the progress made in microprocessor technology (LSI), have made it necessary to adopt a new approach to the realization of these methods in system architectures. The CPS and DPS architectures will be discussed in terms of the time conditions of application job handling, as well as in terms of the multilevel layer structure of those models performing processing and communication functions. The job handling process has a virtual character: the user does not know by which mechanisms and in which part of the system's resources his job is being handled.

ARCHITECTURE OF THE DP SYSTEM

The main function of the DPS is to provide the network subscribers with access to different types of HOST computer resources such as:

-- program libraries and specialized databases,
-- problem-oriented programs and system simulators,
-- information about different fields of knowledge, with different degrees of selectivity in the data files (information retrieval).

Access to distributed computing facilities has not been considered here.

Thus the realization of the above-mentioned applications can be identified with those well-known methods used in job handling: distributed data processing and database distribution.

The users' requirements with regard to the job service mode, using simultaneous differentiation of the hardware and software properties, make it necessary to implement suitable mechanisms in the HOST computers and network NODES. By comparing the model properties of a centralized and a heterogeneous network, and on the basis of the application criteria, we can distinguish certain functions and mechanisms typical of both models. Users can only access the network resources if the following special functions of the DPS are implemented:

-- job handling by means of local and remote resources in arbitrary service modes: batch, interactive, information retrieval, with uniform methods of accessing resources,
-- transformation of the characters, control commands and command language of the local computer into the network control language, when necessary,
-- control of the user's cooperating processes in local and remote network computers under man-computer communication conditions (J/O),

-- preparation, initiation and closing of the job transfer between local and remote network resources. This transfer is related to packeting, network address representation and connection control,
-- communication control on the level of logical connections (e.g., routing, frame forming) and physical connections through a communication network.

Suitable hardware and software mechanisms, performing the processing on different levels of the system's hierarchy, are assigned to the above-mentioned functions. Job handling by means of these functions in the network structure depends on the different forms of communication between the separate levels. One of the methods of defining the functional properties of the system is to analyze the mechanisms which perform the processing and communication functions from the point of view of time requirements when handling application jobs in a network with a centralized and distributed structure.

FUNCTIONAL MODELS OF THE JOB HANDLING PROCESS

When analyzing job handling processes, we make the basic assumption that they can be divided into three components [2]:

-- the information processes of the users,
-- data processing in the computer,
-- the communication processes of the user-resources.

From a functional point of view, the following phases of the JHP in the system can be distinguished:

1. job preparation, consisting of defining data sets, algorithms and user features (user level),
2. input of the job and ouput of the results, depending on the cooperation between the user and the computer system (batch, dialogue, query), and the mode of the I/O data formatting (communication level),

3. job execution in the computer, depending on the service mode (strategy), by means of hardware resources (e.g., store, I/O) and logical resources (databases and programs - computer level).

The performance of the above-mentioned functions can be considered according to the following system states:

1. technically available and under repair,
2. busy and free,
3. overloaded and underloaded.

Certain propositional functions are assigned to the thus formulated model of the job handling process. When fulfilled, these functions enable the job being executed to pass from one phase to the next, or to the end of the job handling process. Each function is considered as a sequence of events in a limited time "t". The values of these functions depend, among other things, on the "a priori" fixed time of execution of the different phases of the job handling process and on the state of the computer (see Appendix).

It is possible to express the particular subsets of the propositional functions according to the following formula:

Function describing the preparation of the job of the i^{th} user:

$$F_i^{pr}(t) = \left\{ F_i^{zj}(t) \wedge F_i^{pd}(t) \wedge \left[(\tau_i^p + \tau_{max}^e) \leq \tau_{max} \right] \right\} .$$

where: F^{zj} determines the busy time of the i^{th} user,

$F^{pd}(t)$ determines the ability of the i^{th} user (user's features) to accept and prepare the job,

$\left[(\tau_i^p + \tau_{max}^e) \leq \tau_{max}\right]$ defines the requirement that the sum of the time necessary for preparing the job (τ_i^p) by the i^{th} user and the maximum admissible time for execution of the job in the system (τ_{max}^e), cannot exceed an assumed total time value (τ_{max}) for job handling.

Function describing the acceptance of the job by the computer system or network, and the output of the results to the user:

$$F^z(t) = F^t(t) \wedge F^c(t) \wedge F^d(t)$$

where:

F^t determines the influence of the terminal state on job acceptance. It is assumed that the terminal can be in one of the following states: technically available, under repair, and busy (job handling);

F^c determines the influence of the computer state on job acceptance from the terminal. It is assumed that the computer can be in one of the following states: technically available, under repair, and busy, i.e., with job acceptance or completion of the job being handled;

F^d determines the influence of the database sets and of the control and executive software. It is assumed that there are the following resource states: data file opening, filling and deleting states, and states of job acceptance and completion of job handling;

Function describing the job execution in the computer

$$F^{r}(t) = F^{cr}(t) \wedge F^{tw}(t) \wedge F^{op}(t)$$

where: F^{cr} determines the influence of the computer state on job execution. The following states of the computer are assumed: computer technically available, under repair, data file conformity state with the job algorithm, data file filling or deleting state, and states of job requesting or completion of job execution,

F^{tw} determines the influence of the terminal state on the result output under the condition defined for F^{t},

F^{op} determines the influence of users' decisions on the continuance of the job and the job result output. The following decision factors are assumed: errors observed, admissible job execution time exceeded, requirement for additional processing procedures.

Function describing the acceptance and transmission of the job being transferred between local and remote system resources by means of communication facilities:

$$F_m^k(t) = \left\{ F_m^{tk}(t) \wedge F_n^{lk}(t) \wedge F_q^{wk}(t) \wedge \left[(\tau^{tk} - \tau^{wk}) \leq \tau_{tr} \right] \right\} .$$

where: F_m^{tk} determines the influence of the preparation of the m^{th} job to be transferred by means of communication facilities. The following states are assumed: division of job state into transportation units (packets), state of network addressing conversion, state of communication connection setting,

F_q^{wk} determines the influence of the q^{th} communication node state on job acceptance for transference between the local and remote computer. The node is assumed to be in one of the following states: technically available, under repair, busy with communication setting-up, disconnection or job transfer,

F_n^{lk} determines the influence of the state of the n^{th} communication line between active nodes, this state being conditioned by the line capacity or overload as a result of job transfer,

$\left[(\tau^{tk} - \tau^{wk}) \leq \tau_{tr}\right]$ defines the requirement that the difference between the predicted time of job transfer (τ^{tk}) between two HOST computers and the real time for job preparation and transference (τ^{wk}) cannot exceed the maximum admissible service time of the communication (τ_{tr}) between two network computers.

In the simplified case, a JHP is considered where the functions $F_i^{pr}(t)$ for every user i = 1,2,3...,n are independent functions:

$$F^{pr}(t) = \bigcup_i \left\{ F_i^{pr}(t) \right\} ,$$

and the propositional functions $F_m^k(t)$ for every job to be transferred m = 1,2,3,...,s are also independent, as the communication between computers takes place in circuit switching mode or packet switching mode (transfer service time sharing) - only in the distributed data processing system (DPS).

$$F^k(t) = \bigcup_m \left\{ F_m^k(t) \right\} .$$

The functions $F^z(t)$ and $F^r(t)$ have the form of singular propositional functions - as a result of the assumption that the Job Handling Process is composed of one computer and one terminal - in the centralized data processing system (CPS). Appropriate conditions determining the influence of the state of a given system component and the dependence of the execution time of the requested jobs upon those states, are assigned to each function of this set.

The model of the "job handling process" is described by a set of propositional functions with logical values. For the purpose of formulating the propositional function $F(t)$, which determines the job handling in a period of time from the instant of job requesting (t_n^w) to the instant of completion of job handling and output of the results, the input request stream is taken into consideration. This stream can be described by the following propositional function:

$$F^w(t) = \left\{ 0 \leq t_n^w \leq t \right\}$$

Thus the condition which has to be satisfied in order to execute the job is formulated in the following way:

1. for the centralized data processing system:

$$F^{CP}(t) = F^w(t) \wedge \bigcup_i \left\{ F_i^{pr}(t) \right\} \wedge F^z(t) \wedge F^r(t) \quad .$$

2. for the distributed data processing stream:

$$F^{DP}(t) = F^w(t) \wedge \left\{ \bigcup_i F_i^{pr}(t) \right\} \wedge \left[\left(\bigcup_j F_j^z(t) \right) \wedge \left(\bigwedge_l F_l^r(t) \right) \wedge \left(\bigcup_m F_m^k(t) \right) \right]$$

where:

$F^{w}(t)$, $\left\{\bigcup_{i} F_{i}^{pr}(t)\right\}$,

$F^{z}(t)$, $F^{r}(t)$ are defined as above,

$F_{j}^{z}(t) \subset F_{G}^{z}(t)$ is a subset of propositional functions defining the states of active terminals when the stream of jobs waiting for processing has been determined,

$F_{l}^{r}(t) \subset F_{Q}^{r}(t)$ is a subset of propositional functions defining the states of active local and remote computers during job execution. It is assumed that $l \geq 2$, which means that at least two subscriber computers will be active,

$F_{m}^{k}(t) \subset F_{V}^{k}(t)$ is a subset of propositional functions defining the states of communications media during the transfer of the handled job stream.

In the case that application jobs are handled entirely with the use of the resources of a local computer (centralized service mode), it is assumed that the function $F_{m}^{k}(t)$ takes the value 1. The value of the function $F^{DP}(t)$ does not depend upon the state of the network communication devices. It is possible to model the JHP by a directed graph with different values on its arcs [3]. The following interpretation of this graph can be accepted: its nodes will represent the states of a computer system (X_i), whereas its arcs will represent the reasons for status changes expressed, for example, by the intensity of the job stream.

LAYERED ARCHITECTURAL MODEL OF THE SYSTEM

One convenient method of determining the characteristics of computer application systems, on the basis of their multi-criterion and multi-parameter analysis, is to represent the system architecture in a multi-level form. To each level a layer is assigned, in which mechanisms performing the characteristic job service functions for the given level are isolated [4,1]. The problem is one of selecting a sufficient number of layers to ensure further development or modification of the system without destroying the structure of the system.

The architecture of a system with a centralized and distributed service presented in Figure 1 permits us to determine the role and place of the processing control and communication functions. By comparing both these system models, it can easily be seen that, in the architectural model of the distributed processing system, some additional layers and modularization appear. Three supplementary layers (instead of one layer) cover the communication functions appearing in the cooperation of HOST computers. The following functions are concerned: functions of transformation of commands, data, computer system language into network control language; functions of managing the session and job transfer; and those of multiplexing the connections between resources and network addressing.

The topology of the computer network architecture in Polish universities is being developed according to the principles of the architectural model outlined above.

APPENDIX

VARIABLES AND STATES OF THE JHP

The following parameters of variable "t" are used to describe formally the static and dynamic properties of an informatics system in the job handling process (JHS):

-- instant of job request to the job handling system: $t^w_n = t^w_{n-1} + \ _n$, where $_n$ is the time interval between the request of the n^{th} job and that of the previous one,
-- instand of ending job preparation (the job is ready to be incorporated into the computer system,
-- instant of job request to the computer system,
-- instant of ending job handling, with result output,
-- instand of readiness of the i^{th} user to accept the job,
-- instant of job handling interrupt, caused by factors outside the computer system,
-- instant of interrupting the job preparation process by the i^{th} user for non-informatic reasons,
-- instant of interrupt completion in job preparation,
-- time at the i^{th} user's disposal,

-- maximum admissible time to solve job/problem,
-- maximum admissible time to handle/execute job in the computer system (in real-time systems),
-- predicted time of job preparation by the i^{th} user,
-- instant of terminal activation,
-- instant of disconnection of the terminal from the computer system,
-- instant of availability of the terminal to accept the job (permission sign for introduction),
-- maximum admissible time that job must wait for terminal availability,
-- instant of database-set opening for the job,
-- instant of filling the data file to be used for the job,
-- instant of data-file deleting,
-- instant of readiness of the i^{th} user's job to be transferred (division into blocks, packets, addressing and setting-up of connection),
-- instant of ending the transfer of job packets through communication nodes (packet-setting into blocks, setting-up of connection to HOST computer and communication disconnection),
-- predicted time of job transfer between two HOST computers,
-- actual time of job preparation and transference,
-- maximum admissible service time of the communication between two user processes in HOST computers,
-- set of the i^{th} user's knowledge (knowledge degree),
-- set of knowledge necessary to prepare the n^{th} job,
-- set of psycho-physiological features of the i^{th} user,
-- set of psycho-physiological features necessary for the preparation of the n^{th} job,
-- set of computer system procedures,
-- set of procedures necessary for handling the n^{th} job,
-- presence of data file in computer store (possibility of activation of data file),
-- data set corresponding to the algorithm of the n^{th} job,
-- set of errors observed by the user.

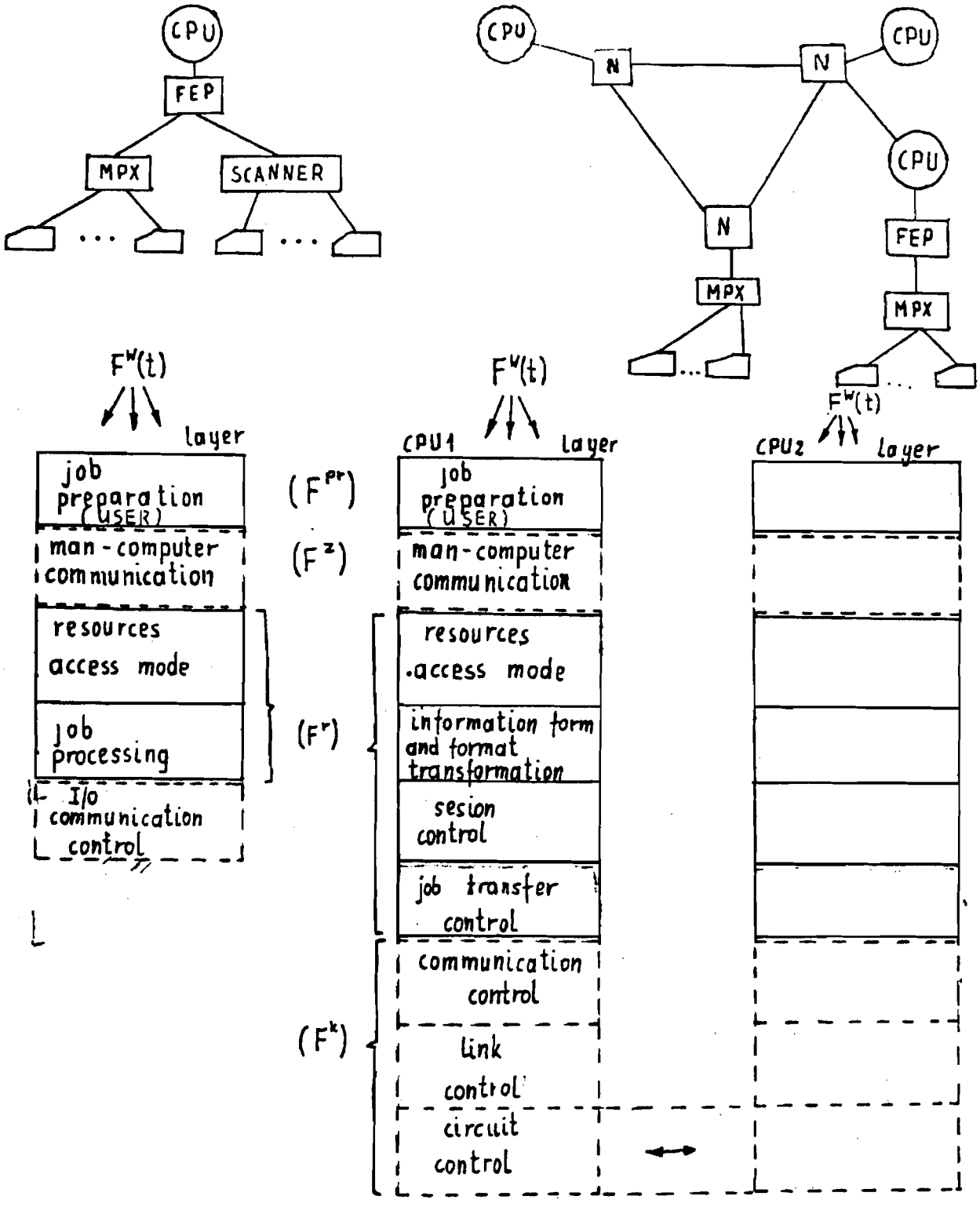
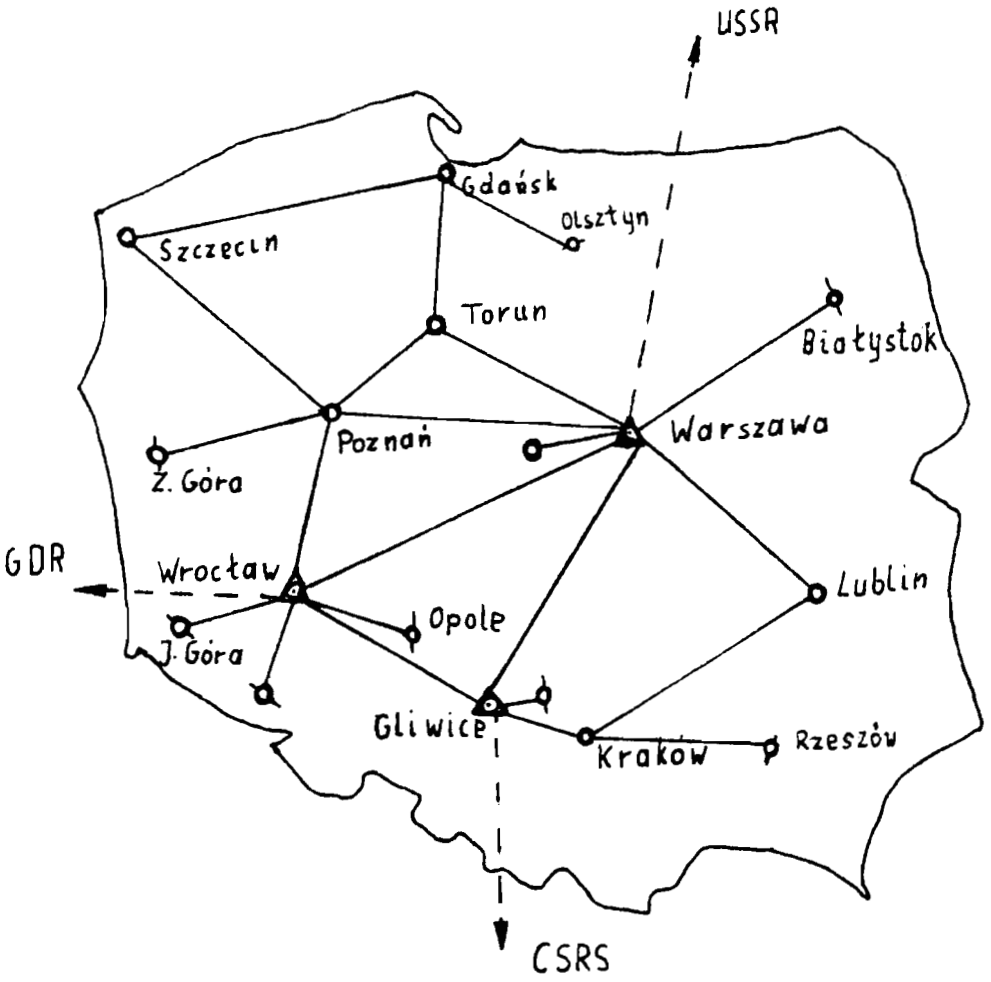


Figure 1. Phases of JHP in (1) a centralized and (2) a distributed data processing system.



—— MSK configuration

– – connections to the other networks

⊙ communication node

—o— concentrator

△ gateway

Figure 2. Configuration of the Inter-University Computer Network (MSK).

REFERENCES

[1] Bazewicz, M. (1979) Properties and Functions of Computer Networks. Part I: Protocols, Chapters I and VII. Wroclaw: Wyd. Politechniki Wroclawskiej, Seria WASC (in Polish).

[2] Bazewicz, M. (1979) Computer Application Systems in Universities. Warsaw: PWN (in Polish).

[3] Maczulin, W.W., and A.P. Piatibratow (1972) Effectiveness of Data Processing Systems. Moscow: Izdatielstvo Sovietskoje Radio (in Russian).

[4] Zimmerman, H. (1979) Open Systems Architecture. Page 865, Proceedings of EUROCOMP '78 Conference. London: On-line.

STANDARD NETWORK ARCHITECTURES

M. Bozzetti

INTRODUCTION

The trend towards distributed computing and the need to access geographically distributed (physical and logical) resources are causing significant changes in the traditional computing scene, which until quite recently was dominated by "self-contained" systems: such changes are marked by major efforts in computer communication networks, i.e., modular configurations with different elements of processing mechanisms assigned to perform distinct functions. In the last decade, networking has become important in the research and business areas, involving EDP manufacturers, public administrations, universities and research bodies. All of these organizations are planning and/or developing commercial, public and experimental networks. Tables 1, 2 and 3 list some commercial, public and research networks which have been developed in recent years.

Considering that the basic goal of a network is to maximize the communication capabilities of the network users, while minimizing the cost, a strict (top-down) architectural approach has been adopted for all the networks.

THE ARCHITECTURAL APPROACH

It is essential to analyze the reasons for such an approach.

Economic Motivations: Changing Cost of Communication and Computing

The cost of hardware is generally dropping very rapidly due to the introduction of VLSE technology. On the other hand, the cost of software is decreasing and communication costs are very slowly decreasing (see Figure 1). For these economic reasons, the design philosophy and priorities of ten years ago, when computer hardware costs were most predominant, must be changed.

Functional Motivations

-- high system performance,
-- high availability,
-- high reliability,
-- high flexibility,
-- high throughput,
-- fail soft (graceful degradation),
-- easy expansion.

The designer of a network architecture must consider:

-- the requirements of the end-user environment,
-- the requirements of the involved EDP systems,
-- the current evolution of the hardware/software technology,
-- the current evolution of the communication technology.

Why Network Architectures?

Each of the network architectures is structured in layers: Figures 2, 3 and 4, showing the structures of the ARPA, EIN and ETHERNET networks, provide examples of this. Keeping in mind

the motivations listed above, it is important to clarify the reasons and needs for a layered structure.

The described functional motivations require the organization of a set of functions, necessary for providing meaningful interaction. These functions should be partitioned in a suitable way, i.e., they should be structured in layers. A structure based on hierarchic layers must insure:

-- independence of activity between layers,
-- sharing of common services,
-- hiding of information about a specific implementation of each layer.

Many techniques may be used as criteria for the subdivision of the functions into layers:

-- division on the basis of physical and logical boundaries within systems,
-- division where there is a change of address or multiplex,
-- division according to a commonality of functions,
-- division on the basis of available services and interfaces.

An architecture is defined as a collection of layers organized in such a way that each layer offers services to the layer above it and uses the services of the one below it. Figure 5 shows the typical elements to be found in any network. From the diagram it can be seen that:

-- different layers within the same unit communicate and cooperate by means of interfaces,
-- the same entities of different units communicate and cooperate by means of protocols.

Useful interaction can occur if all the involved entities observe the same set of rules governing the transfer, structure and meaning of the data. This set of rules is called a protocol, that is, a formal set of conventions governing the format and

the relative timing of message exchanges between two communicating entities. There is a correspondence between layers and protocols. Figure 6 shows the general structure of a generic network architecture.

STANDARDS IN NETWORK ARCHITECTURES

In response to the increasing pressure of teleinformatics, several international standards organizations have undertaken studies to define a set of standards oriented towards facilitating multivendor nets. The most important bodies involved in such activities are:

- -- CCITT (Commissions VII and XVII) for public administrations (PTT's),
- -- ISO (TC 97, SC 16 and SC 6),
- -- ECMA (TC 23 and TC 9) for European Manufacturers,
- -- IFIP (TC 1 and TC 6) for research bodies.

All of these bodies are converging in order to accept and improve the Reference Model of a Network Architecture first proposed by the ISO and called a Reference Model of Open Systems Interconnection. The basic aim of the OSI architecture is to guide the development of standards that will make possible the configuration of a wide variety of computer and data processing networks.

The ISO Reference Model

The ISO Reference Model of Open Systems Architecture represents an effort to standardize the structure for interconnecting computer systems and transferring information among them. The OSI structure comprises seven layers (Figure 7):

- -- physical layer,
- -- link layer,
- -- network layer,
- -- transport layer,
- -- session layer.

-- presentation layer,
-- application layer.

The "frame of reference" introduces the following basic concepts (Figure 8):

-- open working: ability of a user (human being or application program) of any computer to communicate with a user of any other computer,
-- object: any unit within the frame of reference,
-- open object: an application,
-- relationship: any association between objects involving the exchange of information,
-- session: relationship between open objects,
-- interconnection mechanism: any means for provision of a relationship.

The application layer provides protocols and/or the distributed information (system) service appropriate to an application, to its management and to system management. Applications cooperate and intercommunicate by means of and according to application layer protocols. The other layers exist only to support this layer.

The presentation layer provides a set of services which may be selected by the application layer to enable it to interpret the meaning of the data exchanged. The presentation layer is site independent.

The session layer supports the interactions between cooperating application entities by means of two services:

-- the session administration service, which binds two application entities into a relationship and unbinds them,
-- the session data transfer control service, which controls the data exchange between two application entities.

The transport layer provides transparent, reliable and cost effective transfer of data, optimizing the use of the available communication resources.

The network layer provides functional and procedural means of transport layer independence from routing switching considerations, including the case where several communication resources are used in tandem. Network functions are based upon the use of telecommunications facilities.

The link layer provides the functional and procedural means to establish, maintain and release one or more data links between two or more network units (hosts, front-ends, nodes, etc.)

The physical layer provides mechanical, electrical, functional and procedural characteristics to establish, maintain and release data circuits.

COMPANY	NETWORK NAME
Burroughs	DNS
Control Data	DNS
Data Point	ARC
Comten	CNA
Digital E.C.	DECNET
Fujitsu	FNA
Honeywell	DSE
HP	DSN
IBM	SNA
ITT	CNA
Mod Comp	MAXNET
NCR	DNA
Nixdorf	NCN
Oki	DONA
Olivetti	ONE
Philips	COMSYS
Siemens	TRANSDATA
Univac	DCA

Table 1. Some commercial nets

COUNTRY	PUBLIC NETWORK NAME
Austria	CUDN
Belgium	RTT
Canada	DATAPAC
	INFOSWITCH
ECC	EURONET
France	TRANSPAC
Germany	EDS
Japan	DCNA
	VENUS
	DDX
UK	EPSS
Scandinavia	DATANET
Spain	CTNE
South Africa	SAPONET
USA	TELENET
	TYMNET
	BDN
	TNS
	TELPAK
	COMPAK
	GRTHNET
	CALCOMP

Table 2. Some public data networks

NETWORK NAME	COUNTRY
ARPA	USA
EIN - COST 11	ECC
RESOURCE CYCLADES	France
GMD	Germany
NPL - NET	UK
Berlin - NET	Germany
RPCNET	Italy
ETHERNET	USA
MIT - NET	USA

Table 3. Some of the most well known research nets

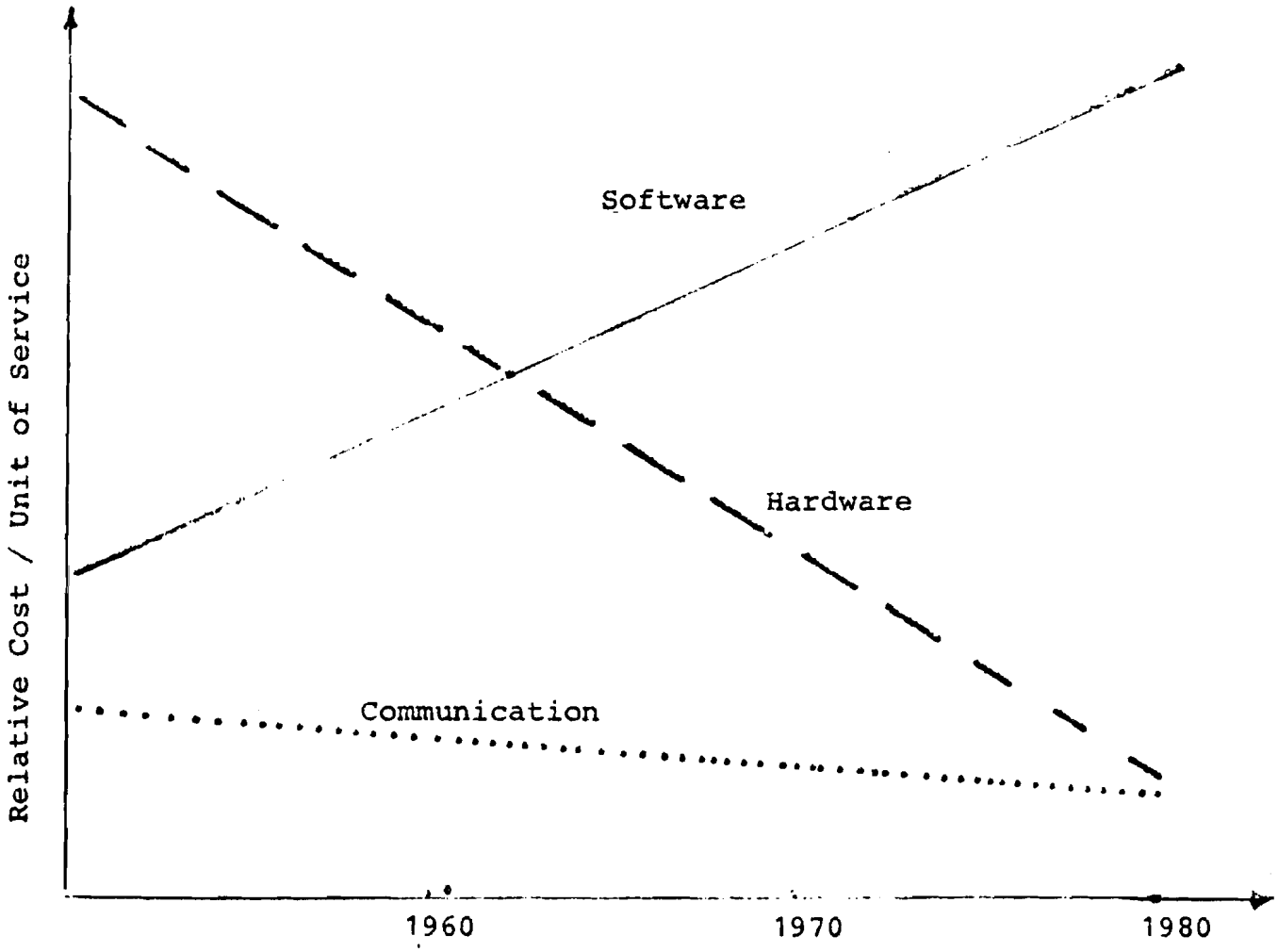


Figure 1. The changing costs of EDP Systems

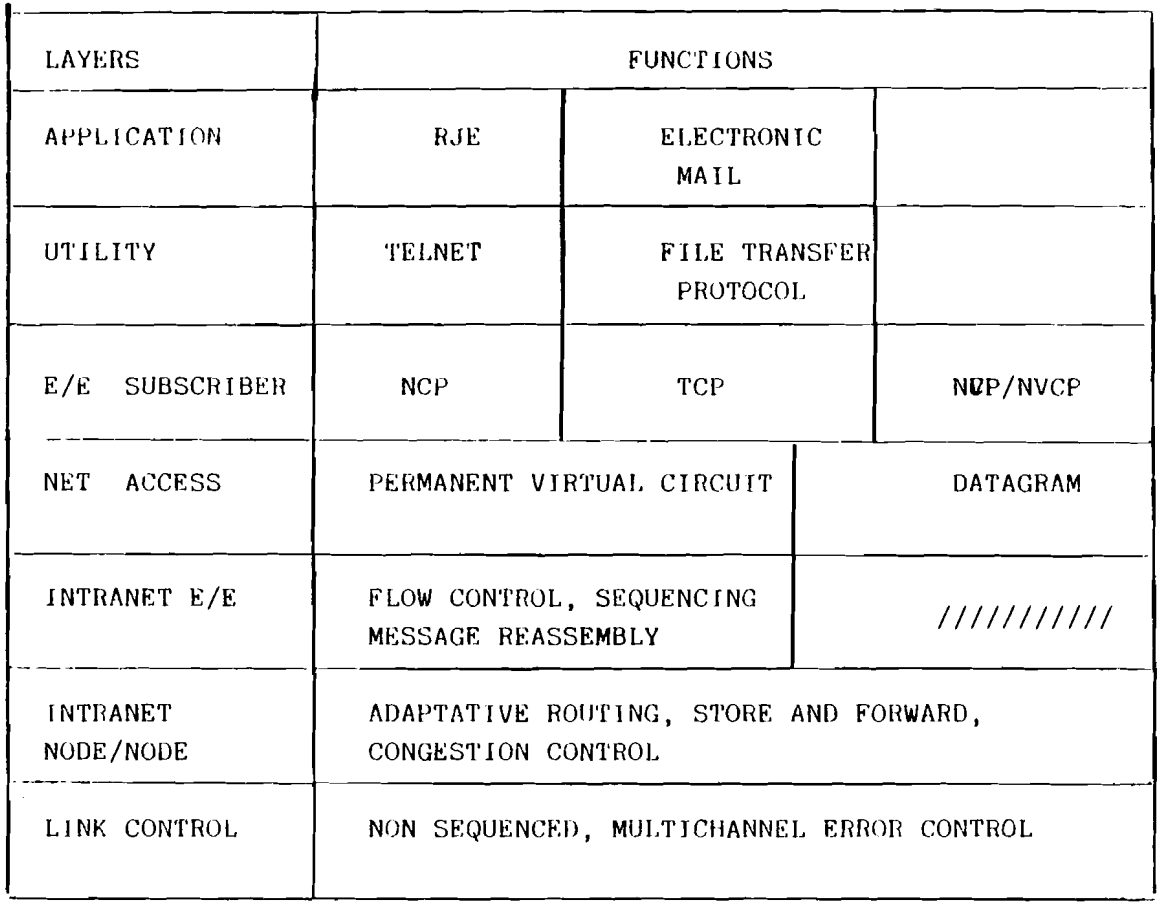


Figure 2. ARPANET layered architecture

LAYERS	FUNCTIONS		
APPLICATION	RJE (BS)	ELECTRONIC MAIL	REMOTE DB ACCESS
UTILITY	BULK TRANSFER	VIRTUAL TERMINAL	
E/E SUBSCRIBER	TRANSPORT SERVICE WITH LIAISON AND LETTERGRAM		
NET ACCESS	SC - NSC PROTOCOL (DATAGRAM)		
INTRANET E/E	PACKET ORDERING AND SEQUENCING		
INTRANET NODE to NODE	ADAPTATIVE ROUTING, STORE AND FORWARD CONGESTION CONTROL		
LINK CONTROL	HDLC LIKE WITH CHANNEL FLOW AND ERROR CONTROL		

Figure 3. European Informatics Network (EIN) - layered architecture as implemented by CREI

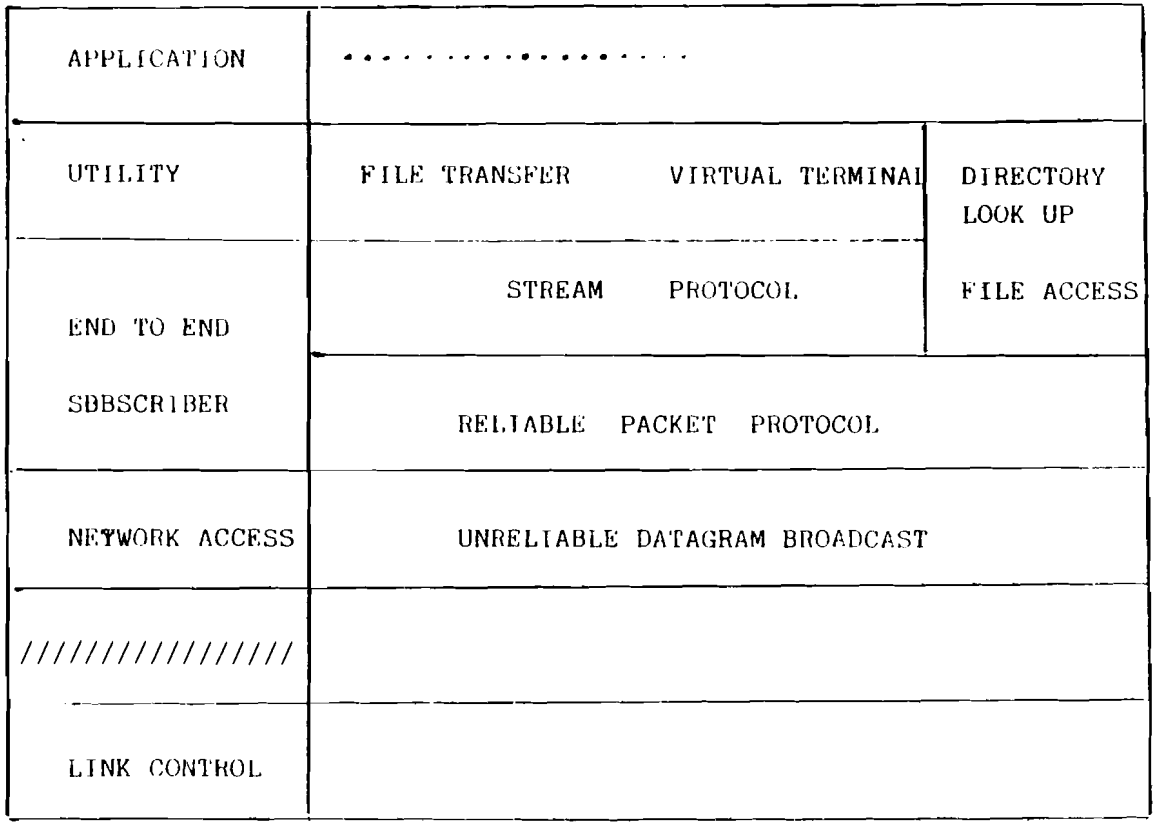


Figure 4. ETHERNET layered architecture

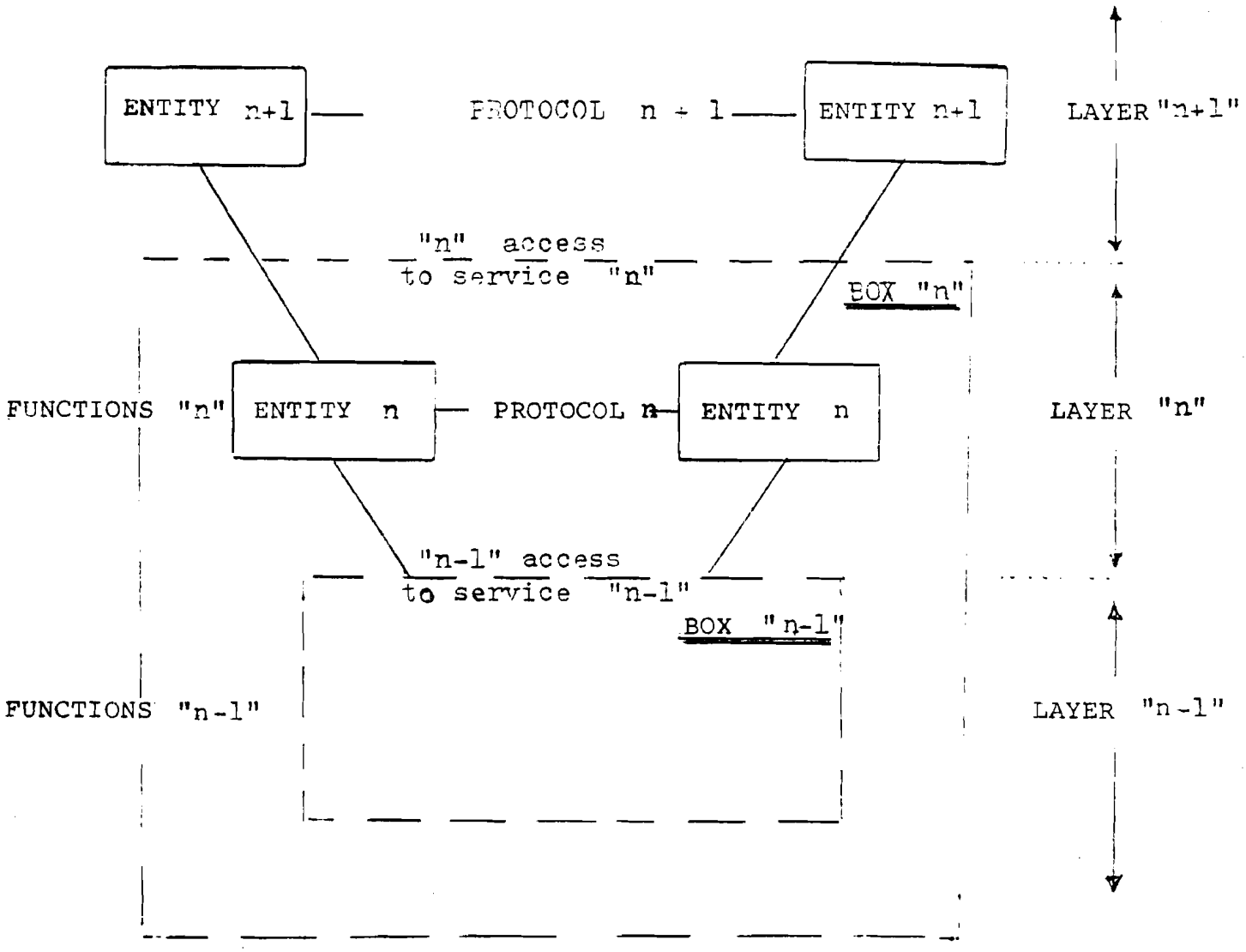


Figure 5. Basic elements of layered structure

LAYERS	FUNCTIONS
APPLICATION	FUNDS TRANSFER, INFORMATION RETRIEVAL, ELECTRONIC MAIL, TEXT EDITING, TRANSACTION HANDLER, etc.
NET UTILITY	VIRTUAL FILE (FILE TRANSFER) AND VIRTUAL TERMINAL SUPPORT
END/END SUBSCRIBER	INTERPROCESS COMMUNICATION (VIRTUAL CIRCUIT, DATAGRAM, BROADCAST)
NET ACCESS	NET ACCESS SERVICES
INTRANET END/END	FLOW CONTROL, SEQUENCING
INTRANET NODE TO NODE	ROUTING, CONGESTION CONTROL
LINK CONTROL	ERROR HANDLING, LINK FLOW CONTROL

Figure 6. General protocol layers

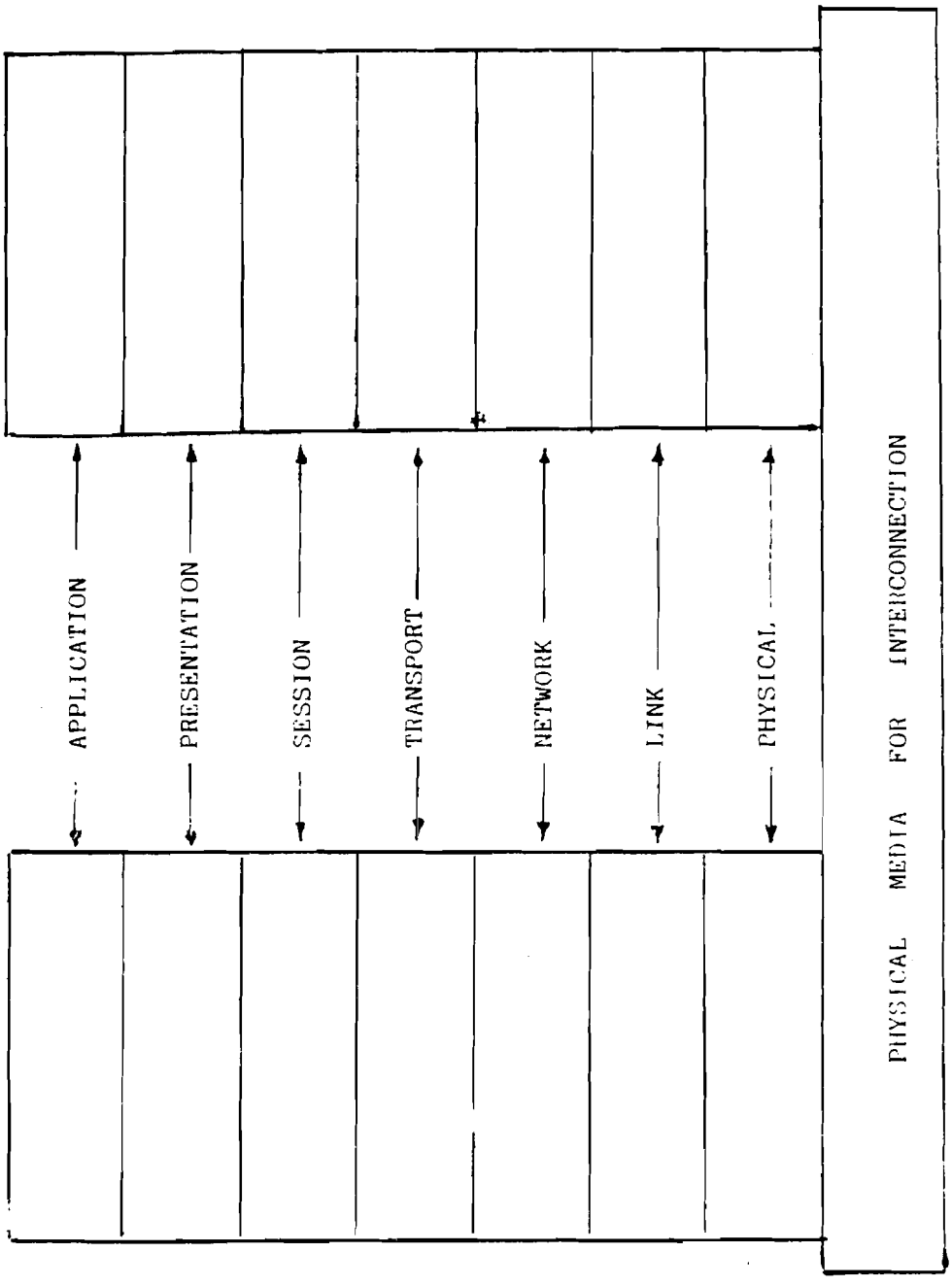


Figure 7. The ISO seven layers structure

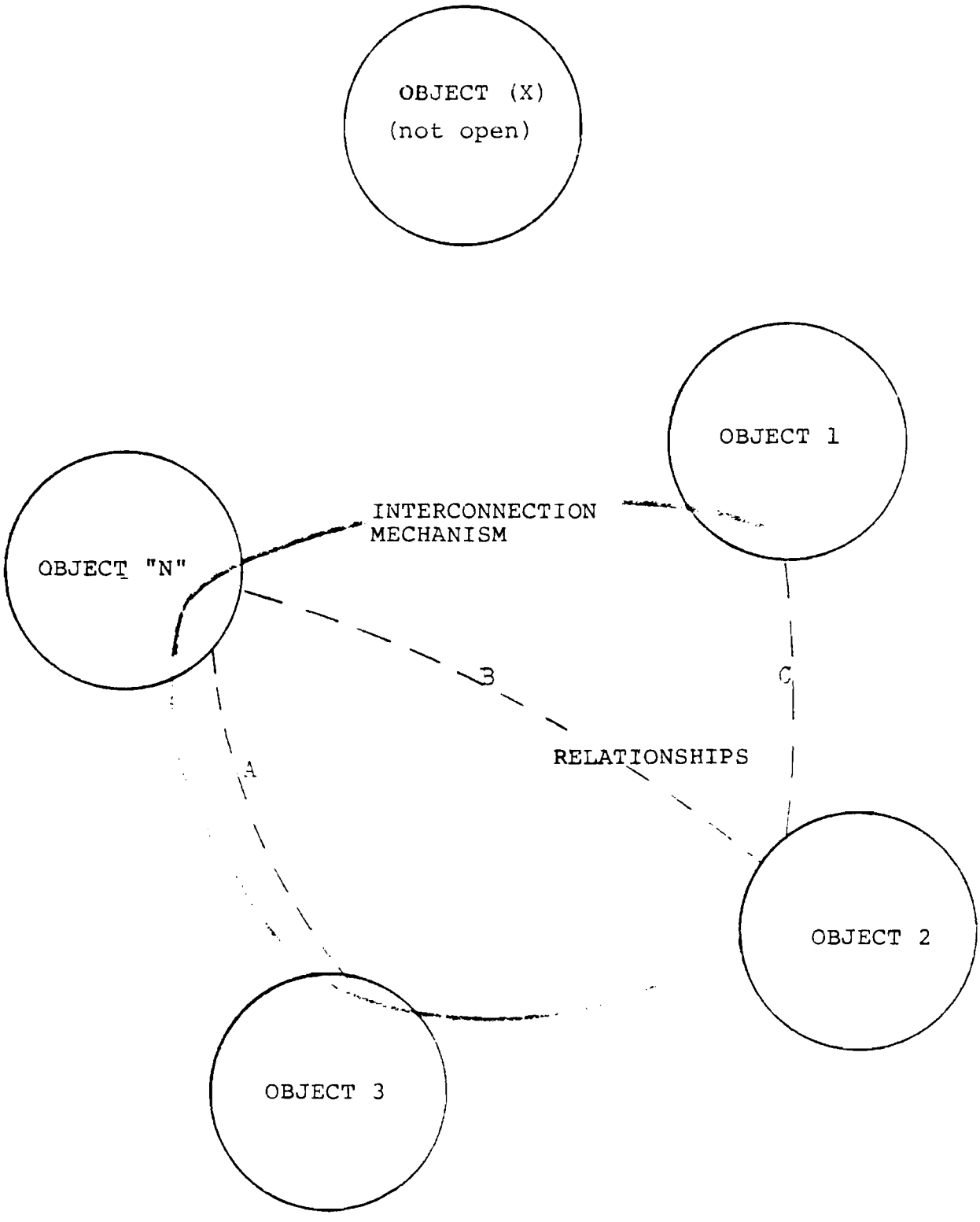


Figure 8. ISO Reference Model: frame of reference

M-T INTERCONNECTION OF INTERLOCUTORS IN COMPUTER NETWORKS

S. Alfonzetti, S. Casale and A. Faro

INTRODUCTION

In a packet-switched computer network, one of the fundamental problems is that of the interaction between two user processes running on remote computers. This interaction is hierarchically organized in communication levels (see Figure 1), each of which is realized by means of a couple of communication processes performing given functions, such as flow and error controls, fragmentation, and link opening and clearing. This communication structure has several advantages, among which are greater system modularity and the possibility that a process of level "l" can be utilized by several processes of level "l + 1" (multiplexing).

The set of rules governing the message exchange between two remote communication processes of a given level is called a "protocol". Because of their importance, communication protocols have been studied intensively and in particular, formalization methods have been developed for protocol specification and validation [1]. Nevertheless, in computer networks, in addition to the interaction between two remote processes of the same level, the interaction between two processes of adjacent and different levels needs to be considered (see Figure 1).

The aim of this paper is to study the latter type of interaction. This study will be made by modelling each communication process as an "interlocutor", according to the "theory of colloquies" [2]. In particular, by adopting a formal description of the interlocutor by means of a Mealy automaton, the necessary and sufficient conditions are given in order that two interlocutors of adjacent levels may lead to a composite interlocutor. Finally, a first algorithm verifying the above conditions and, in this case, a second algorithm giving a formal description of the composite interlocutor starting from the descriptions of the two interlocutors , are given.

THE INTERLOCUTOR AND ITS INTERCONNECTIONS

Because it is possible to describe a protocol as a colloquy between two entities called "interlocutors" [2], a computer network can be studied usefully by considering a model comprising a set of interlocutors variously interconnected by means of communication channels. The interlocutor is essentially a device with three types of inputs (messages μ, commands η, texts τ), three types of outputs (messages m, commands c, texts t) and a set of internal states (Figure 2). The ports relative to the messages m and μ are called M-ports; those relative to the commands η and c, H-ports; and those relative to the texts t and τ, T-ports.

Two interlocutors can be interconnected in one of the three following meaningful ways [3]:

-- M-M interconnection (Figure 3) in which the output messages of one are the input messages of the other, and vice versa. Such an interconnection is typical of two remote interlocutors of the same level performing a colloquy protocol; it gives rise to a device with four text-ports and four command-ports called a "Communication Device" (CD),

-- T-T interconnection (Figure 4) in which the output texts and commands of one are, respectively, the input texts and commands of the other, and vice versa. This interconnection is typically local and gives rise to a device called a "Procedure Adaptor" (PA). Examples of a PA are in the nodes of a network and in the gateways between two networks,
-- M-T interconnection (Figure 5) in which the messages and commands of one are, respectively, the texts and commands of the other. This interconnection is typically local and in general gives rise to a new interlocutor called a "Composite Interlocutor" (CI). Examples of a CI are in the host-computers of a network.

Figure 6 shows the protocol structure of the European Informatics Network (EIN) with the three types of interconnections already described.

In the study of an interlocutor in relation to its two adjacent interlocutors, it is still convenient to distinguish the commands exchanged between either one or the other. Using the subscripts T, M and L, respectively, for the commands relative to the adjacent interlocutor connected by means of the T-ports, to the adjacent interlocutor connected by means of the M-ports and to the Local Controller*, the following definition can be given:

Def. 1: The interlocutor is a device with five inputs $(\mu,\eta_T,\eta_M,\eta_L,\tau)$, five outputs (m,c_T,c_M,c_L,t) and a set of internal states (see Figure 7).

The internal structure of an interlocutor is shown in Figure 8; it differs from the one usually used in [2],[4] and [5], due to the absence of the buffers relative to the ports T and H.

*The Local Controller is a device which allows the user to interact with an interlocutor of a given level without involving the intermediate levels.

This structure is particularly convenient in the study of the M-T interconnection for the realization of a composite interlocutor due to the absence of asynchronism between the two interlocutors.

The interlocutor works in the following way:

-- at the arrival of an input message μ, the input unit separates the envelope from the possible text*; the first is forwarded to the procedure unit (PU) which, according to the present state of the interlocutor, emits one or more quadruples $(\underline{m},c_T,c_M,c_L)$; the dashed line labeled with c_{IU} in Figure 8 points out the possible authorization given to the IU in order to transmit the text; it is mutually exclusive with a command c_T,
-- at the arrival of an input command η (η_T or η_M or η_L) the PU, according to the present state, emits one or more quadruples $(\underline{m},c_t,c_M,c_L)$.

Finally the output unit (OU) has the task of assembling the envelope $\underline{m}$ and the possible text τ in the output message m. From the above, the following may be deduced:

Def. 2: The PU is a device with four inputs $(\underline{\mu},\eta_T,\eta_M,\eta_L)$, four outputs $(\underline{m},c_T,c_M,c_L)$ and a set of internal states**.

Supposing that the PU processes only one input at a time, and can emit more outputs for one input, the following

---

*The envelope is that part of the message containing information of interest to the PU; the text is the remaining part, to which the PU is transparent.

**It should be noted that the internal states of the PU coincide with those of the interlocutor.

definitions can be given:

Def. 3: The input vocabulary of an interlocutor is the union of the input sets of each channel:

$$J = J_{\mu} \cup J_{\eta_T} \cup J_{\eta_M} \cup J_{\eta_L} \quad .$$

Def. 4: The output vocabulary of an interlocutor is the Cartesian product of the output sets of each channel:

$$U = U_m \times U_{c_T} \times U_{c_M} \times U_{c_L} \quad .$$

Consequently the PU can be represented by a Mealy machine having:

input set	J
output set	$U^{(n)}$
internal state set	S
function of the new state	$J \times S \rightarrow S$
output function	$J \times S \rightarrow U^{(n)}$

where $U^{(n)}$ is the Cartesian product*:

$$\underbrace{U \times U \times \ldots \times U}_{n \text{ times}}$$

Def. 5: The integer n is called the order of the interlocutor.

In computer networks the presence of interlocutors of order greater than one is typical. Examples of these interlocutors are encountered on the levels which operate fragmentations (from files to letters, from letters to packets).

---

*The PU can also be considered as a machine which, at the reception of an input $i \in J$, produces:

-- an ordered n-tuple of output envelopes $\underline{m}$
-- an ordered n-tuple of output commands $\underline{c}_T$
-- an ordered n-tuple of output commands $\underline{c}_M$
-- an ordered n-tuple of output commands $\underline{c}_L$

From what has been said, it can be seen that the behavior of the interlocutor is completely specified if the PU is formally described. Several formal description techniques have been used in the literature to represent the PU, such as those based on logical matrices, on Petri networks, on variable structure sequential machines or on Mealy automata. This last formalization technique will be utilized later on to study the M-T interconnection of two interlocutors, as it is the most favorable one for this type of study.

THE M-T INTERCONNECTION

The M-T interconnection schema of two interlocutors A and B is shown in Figure 9. The device inside the dashed line has the same inputs and outputs as a single interlocutor and we can therefore consider it also to be an interlocutor. In order to verify this, it is necessary to determine the input and output sets and the internal states. From Figure 9 it can be seen that the possible composite interlocutor must have:

$$J_{\eta_T} = J^A_{\eta_T} \quad , \qquad J_{\eta_M} = J^B_{\eta_M} \quad ,$$

$$U_{c_T} = U^A_{c_T} \quad , \qquad U_{c_M} = U^B_{c_M}$$

In addition, as a consequence of the stated hypothesis that the PU processes only one input at a time, it is necessary to assume that the possible composite interlocutor receives only one command at a time from the local controller; hence:

$$J_{\eta_L} = J^A_{\eta_L} \cup J^B_{\eta_L} \quad .$$

In order to determine the other three sets U_m, J_μ, and U_{c_L}, we consider Figure 10 which indicates the IU, the PU and the OU of the possible composite interlocutor. The resultant PU is activated by a couple of envelopes $(\underline{\mu}^A, \underline{\mu}^B)$ and produces

one or more couples of envelopes $(\underline{m}^A, \underline{m}^B)$ and commands to the local controller (c_L^A, c_L^B). From the above:

$$J_\eta \subseteq J_\eta^A \times J_\eta^B \quad ,$$

$$U_m \subseteq U_m^A \times U_m^B \quad ,$$

$$U_{c_L} \subseteq U_{c_L}^A \times U_{c_L}^B \quad ;$$

the inclusion signs are used because generally not all the above couples are possible. Figure 11 shows the inputs and the outputs of the composite interlocutor, while Figure 12 reports the message structure of the composite interlocutor and points out the texts and the envelopes of the three interlocutors.

Lastly, the possible composite interlocutor needs to have as its set of internal states a subset of the Cartesian product of the internal state sets of A and B, that is:

$$S \subseteq S^A \times S^B \quad .$$

It is now necessary to determine the conditions which the two interlocutors A and B have to satisfy so that their M-T interconnection may give rise to a new interlocutor. Referring to the above, the following theorem is valid:

Theorem 1: The necessary and sufficient conditions in order that the two M-T interconnected interlocutors A and B give rise to a new interlocutor are:

1. the sets of the output commands of A and B respectively for B and A are embodied (or equal) in the sets of the input commands of B and A respectively coming from A and B, that is:

$$U^{A}_{c_M} \subseteq J^{B}_{\eta_T} \quad , \quad U^{B}_{c_T} \subseteq J^{A}_{\eta_M}$$

2. for each input of the resulting PU and for each state (S^A, S^B), the number of commands which are exchanged by the two PU in the loop $c^A_M = \eta^B_T - c^B_T = \eta^A_M$ is finite (see Figure 9).

This theorem is proved in Appendix A.

Given the formal descriptions of A and B by means of Mealy automata, the verification of condition 1 is performed by a simple visual inspection, whereas for that of condition 2, it is necessary to study the internal loop $c^A_M = \eta^B_T - c^B_T = \eta^A_M$. In order to deal with such a study, with the aim of describing the composite interlocutor , the following hypotheses were made:

Hyp. 1: The composite interlocutor does not accept external inputs until it has completely performed the actions relating to the previous input

Hyp. 2: If at least one of the two interlocutors has an order greater than 1, it is possible that, because of an input, it will emit for the other interlocutor one more non nul command (segment); this latter interlocutor will emit for the first one an output segment only after it has completely processed the input segment.

In these hypotheses we propose an algorithm allowing us to determine if condition 2 is satisfied (see Appendix B). Moreover, such an algorithm gives the maximum W_{max} of $W = W_A + W_B$, where W_A and W_B are respectively the number of times that the interlocutors A and B treat segments because of the reception of an input by the composite interlocutor. W_{max} can be utilized in order to determine a number greater than the order of the possible composite interlocutor. In fact, the following theorem, is valid:

Theorem 2: Let n_A and n_B be the orders of the two interlocutors; the order of the composite interlocutor, if it exists, satisfies the inequality:

$$n \leq \begin{cases} \sum_{i=1}^{\frac{1}{2}W_{max}} (n_A n_B)^i , & \text{if } W_{max} \text{ even} \\ \max(n_A, n_B) \sum_{i=1}^{\frac{1}{2}(W_{max}+1)} (n_A n_B)^{i-1} & \text{if } W_{max} \text{ odd} \end{cases}$$

This theorem is proved in Appendix A. If $n_A = n_B = 1$, the previous inequality becomes:

$$n \leq \begin{cases} \frac{1}{2}W_{max} , & \text{if } W_{max} \text{ even} \\ \frac{1}{2}(W_{max}+1) , & \text{if } W_{max} \text{ odd} \end{cases}$$

and from this it is possible to deduce that the order of the composite interlocutor is 1 both for $W_{max} = 1$ (degenerate case in which the two interlocutors do not exchange commands) and for $W_{max} = 2$ (case in which each interlocutor works at least once). In order to consider how two interlocutors of the first order can give rise to a composite interlocutor of order greater than 1, we consider the example of Figure 13. The interlocutor A, receiving the command "SEND N PACKETS" from its user, sends the command "SEND PACKET" to B and sets a variable of state equal to N. The interlocutor B, receiving a command "SEND PACKET" from A, sends a packet acknowledging A by means of the command "PACKET SENT". On the reception of this command from B, A decreases by 1 the variable of state S^A and, if $S^A > 1$, it emits the command "SEND PACKET" to B. The composite interlocutor is clearly of order N, because on the reception of the single input command "SEND IN PACKETS" it emits N packets.

If the conditions of theorem 1 are verified, the algorithm 2 described in Appendix B can be utilized in order to determine:

-- the set of internal states,
-- the transition/output tables,
-- the order of the composite interlocutor.

CONCLUSIONS

In this paper the M-T interconnection of two interlocutors representing adjacent levels in a computer network has been studied. In particular, we have given the necessary and sufficient conditions for the existence of the composite interlocutor. We have proposed algorithms to verify these conditions and to determine the automaton which represents the composite interlocutor and its order.

The results of this paper can be applied by a programmer who has to implement two adjacent levels in a computer network. Generally he can choose between two different solutions: either implement two distinct programs, one for each level, or implement only one program which provides the functions of the two programs.

Sometimes the solution to be adopted is imposed by the operation which has to be performed by the interlocutors. Therefore in the asynchronous operation between the two levels, the solution of implementing two distinct programs must be adopted, whereas in the synchronous operation the solution of implementing only a single program must be adopted*.

---

*The multiplexing between the levels is an example of asynchronous operation, whereas a control module to test and measure the communication subnetwork behavior by transmitting sequences of packets with fixed interdeparture time intervals [6] is an example of synchronous operation.

If the above operating constraints are not present, the choice can be made generally by evaluating the following points:

a) Memory space: the single program solution allows us to avoid the interface files between the two programs, whereas the memory space is substantially the same,
b) Processing rate: if the processes are running on the same computer, the solution of the single program offers a higher processing rate in relation to the solution of the two distinct programs because there is no delay due to the I/O operations relative to the interface file; if the two processes are running in distinct processors, for example in a multimicroprocessor arrangement, it is necessary to evaluate if the I/O time on the interface buffers will be compensated by the contemporaneous activity of the programs,
c) Modularity: a single program can also be modular if the software is organized by means of suitable subroutines.

The proposed algorithms can be applied usefully by the programmer who has to implement one or more levels if, before he writes the program, he studies the protocol and interface with the adjacent levels. The formal description of the interlocutor which must be used can be either the one presented in this paper or others from which it is possible to pass to this description by means of suitable algorithms [7].

In summary, when the programmer intends to implement two adjacent interlocutors by means of a single program, he should structure his work in the following phases:

1. Formalization,
2. Application, if necessary, of the algorithms in order to pass from the initial formalization to the automaton one,
3. Application of algorithm 1,
4. Application of algorithm 2,
5. Implementation.

Lastly, it should be noted that the M-t interconnection study has been carried out considering an interlocutor model in which the contexts [5] and [8] do not explicitly appear. Nevertheless, such a study can easily be extended to consider this case also under the condition that it is applied to the proper processing unit.

APPENDIX A

PROOF OF THEOREM 1.

Necessary Condition

If we suppose that the composite interlocutor exists, due to any input and from any state pair (of A and B), the PU's pair passes to a final state pair. If, ab absurdo, condition 1 is not verified because, for example, a command $c_M^A \notin J_T^B$ exists, when this command is emitted from A and received by B, the next state of B cannot be computed; this is absurd as, due to the hypothesis, the final state must always exist. Analogously, if an input exists which, from a given state pair, produces a command exchange between A and B without term (periodical exchange), the final state of A and B cannot be computed.

Sufficient Condition

Vice versa, if conditions 1 and 2 are verified, from any state pair and for any input, it is possible to compute the final state of the pair A and B and their outputs. Because it is possible to organize these outputs so that they belong to a set

$U^{(n)}$ with n being suitable, it is consequently possible to describe the interconnection between A and B as a Mealy automaton and therefore it is an interlocutor.

PROOF OF THEOREM 2.

In order to prove the inequalities, we assume that for both the interlocutors, any input causes the emission of the maximum number of outputs (equal to the corresponding orders n_A and n_B) directed both to the other interlocutor and outside. In this hypothesis, we activate one of the two interlocutors, say A, by means of an input. A will emit an output segment constituted by n_A commands to B and n_A outputs. On receiving such a segment, and for each command received from A, B will emit, subsequently, n_B commands to A and n_B outputs. We have therefore the emission of a segment constituted by $n_A n_B$ commands to A and $n_A n_B$ outputs. Proceeding in this way until the interaction is finished (due to the emission of a nul command segment from one of the two to the other), the two interlocutors emit to the outside two segments whose lengths are (Figure 14):

$$L_A = n_A + n_A^2 n_B + \ldots + n_A^{W_A} n_B^{W_A - 1} \quad . \qquad \text{(for A)}$$

$$L_B = n_A n_B + n_A^2 n_B^2 + \ldots + n_A^{W_B} n_B^{W_B} \quad . \qquad \text{(for B)}$$

Given that the elements of the two segments are of different types (m^A,c_T^A, and c_L^A for A,m^B,C_M^B, and c_L^B for B), in order to determine a number greater than the order, it is necessary to find the maximum of L_A and L_B.

Since W_A and W_B are either equal or differ by one ($W_A = W_B + 1$ because A is activated first), we have the following:

-- if $W = W_A + W_B$ is even, $W_A = W_B = \frac{1}{2}W$, and then

$$L_B = \sum_{i=1}^{\frac{1}{2}W} (n_A n_B)^i \geq L_A \quad ;$$

-- if $W = W_A + W_B$ is odd, $W_A = W_B + 1 = \frac{1}{2}(W + 1)$, and then

$$L_A = n_A \sum_{i=1}^{\frac{1}{2}(W+1)} (n_A n_B)^{i-1} \geq L_B \quad .$$

Analogously, if B is activated first, we have:

-- if W is even,

$$L_A = \sum_{i=1}^{\frac{1}{2}W} (n_A n_B)^i \geq L_B \quad ;$$

-- if W is odd,

$$L_B = n_B \sum_{i=1}^{\frac{1}{2}(W+1)} (n_A n_B)^{i-1} \geq L_A \quad .$$

In conclusion, given that W_{max} is the maximum of W for all the possible inputs and the possible state couples of A and B, that is:

$$W_{max} = \max_{J \times S^A \times S^B} W$$

we have the inequalities of theorem 2.

APPENDIX B

ALGORITHM 1.

For simplicity, the algorithm is described for two first order interlocutors. Given the formal descriptions of the PU's as transition/output tables, the study of the loop $c_M^A = \eta_T^B$ - $- c_T^B = \eta_M^A$ can be made by building the graph C_{AB}. The nodes of this graph represent the commands c_M^A and c_T^B and are labeled respectively by means of two sets S_{cM}^A and S_{cT}^B constituted by states of A and B in which it is possible that the command left the machine. The arcs of G_{AB} are oriented and labeled by state transition (the arcs directed from a node c_M^A to a node c_T^B are labeled by states of B and vice versa as shown in Figure 15). Two new oriented graphs G_A and G_B can be associated to G_{AB}. The G_A nodes represent the triplet constituted by a command c_M^A and the state S^A, S^B of A and B after the command c_M^A was emitted; the oriented arcs connect the nodes in such a way that the arrival node may represent the triplet constituted by the next command which will be emitted from A to B and the state S^A, S^B of A and B in this situation*.

*It should be noted that in graph G_A there are no nodes labeled with a triplet (c_M^A, S^A, S^B) in which S^A does not belong to $S_{c_M}^A$; analogously for G_B.

The node in which the command emitted from A to B or vice versa is null is called the final node.

It is possible to prove that:

> The necessary and sufficient condition so that the number of commands exchanged between A and B is finite, is that the two graphs G_A and G_B are trees.

In order to verify this, we note that the G_A and G_B nodes have only one subsequent node, whereas they can have more than one preceding node; therefore it is possible to proceed as follows:

Step 1: building of the vectors V_A and V_B in which the element j indicates the subsequent node of the node j,

Step 2: updating of V_A and V_B respectively for a number of times greater than:

$$\log_2[p(U^A_{C_M})p(S^A)p(S^B)] \text{ and } \log_2[p(U^B_{C_T})p(S^A)p(S^B)] \quad .$$

in such a way as to substitute each time in the entry j, which points to the element k, the entry to which the element k points*,

Step 3: verifying if the elements of V_A and V_B point to elements different from the final node (the case of periodical exchange between A and B; that is, G_A and G_B are not trees because they contain at least one loop), or verifying if all the elements point to the final node (finite exchange between A and B; that is, G_A and G_B are trees).

---

*Given a set . , the operator p(.) supplies the number of elements of the set.

In order to determine W_{max} it is possible to introduce other vectors R_A and R_B in which the element j is initially set to 1, if it points to the final node (because B and A respectively emit no commands); otherwise it is set to 2. Increasing the elements of such vectors by a method analogous to that described in step 2, we obtain finally the number W_{max}, which is the maximum of the elements of R_A and R_B plus one.

ALGORITHM 2.

In order to obtain a formal description of the composite interlocutor, once the conditions of interconnection of theorem 1 are verified, the following procedure can be followed:

Step 1: starting from a state pair, which for problems of reachability should coincide conveniently with the initial state pair of the two interlocutors, we activate the composite interlocutor by means of all its inputs and we determine both the outputs and new states,

Step 2: starting from the states in step 1, we analogously proceed by determining new states, for which we repeat the procedure described in step 1, if they differ from the states already determined.

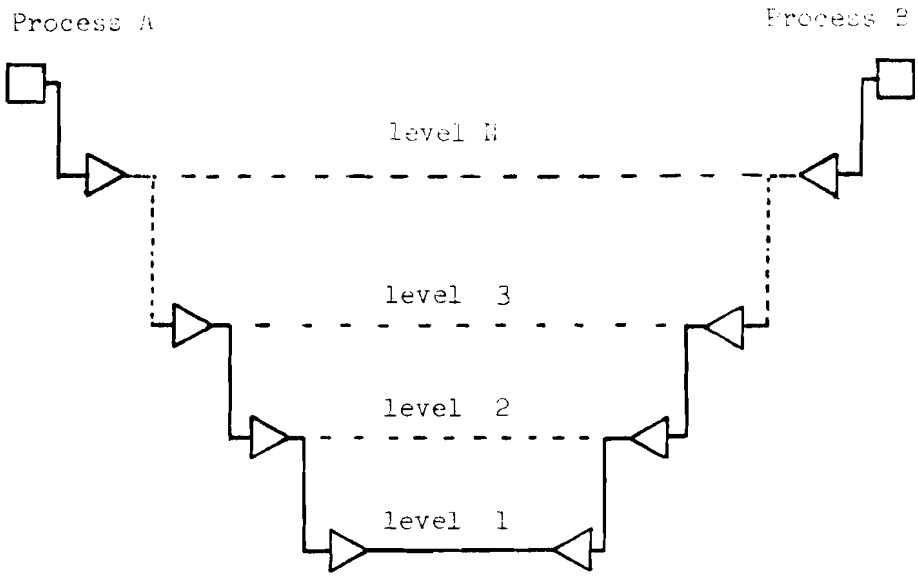


Fig. 1 - Communication levels in computer networks.

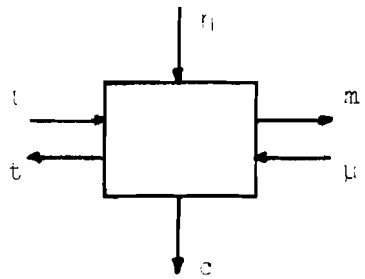


Fig. 2 - Interlocutor.

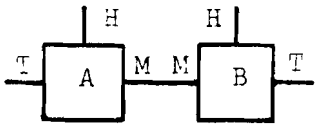


Fig.3- M-M Interconn.

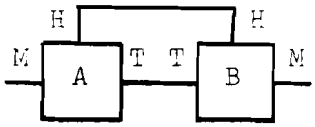


Fig.4- T-T Interconn.

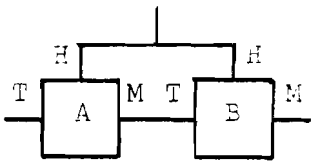
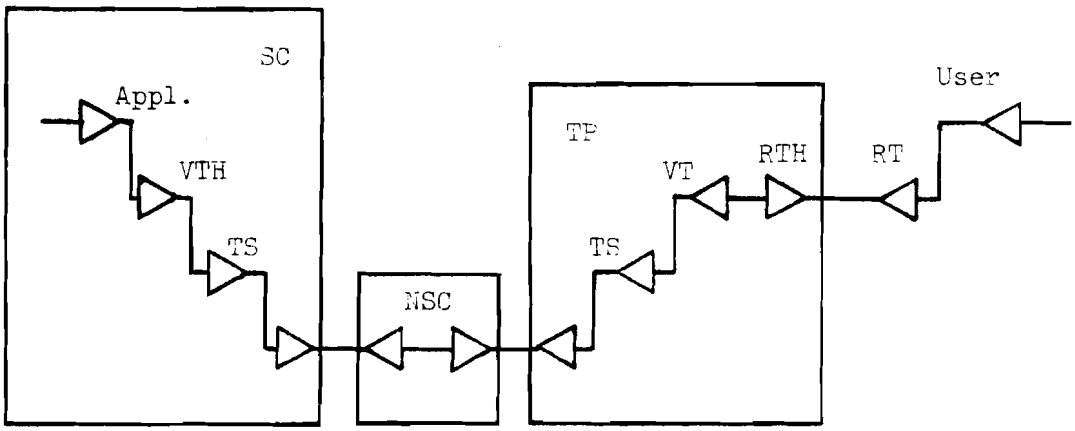


Fig.5- M-T Interconn.



VT = Virtual Terminal VTH = VT Handler

RT = Real Terminal RTH = RT Handler

TS = Transport Station

Fig. 6 - Protocol structure of EIN.

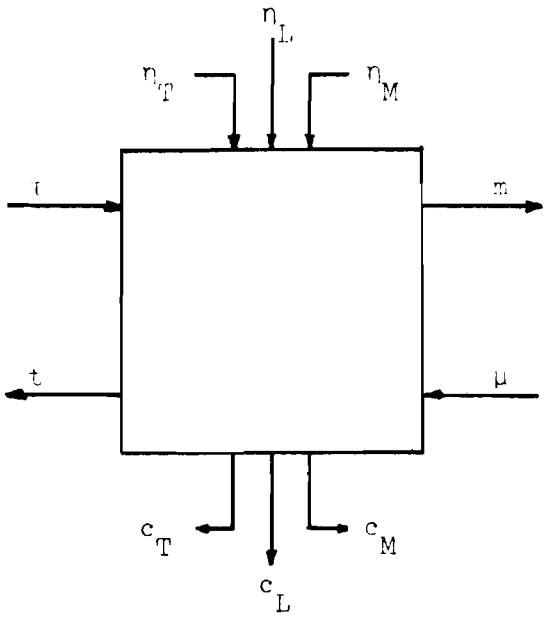


Fig. 7 - Inputs and outputs of an interlocutor.

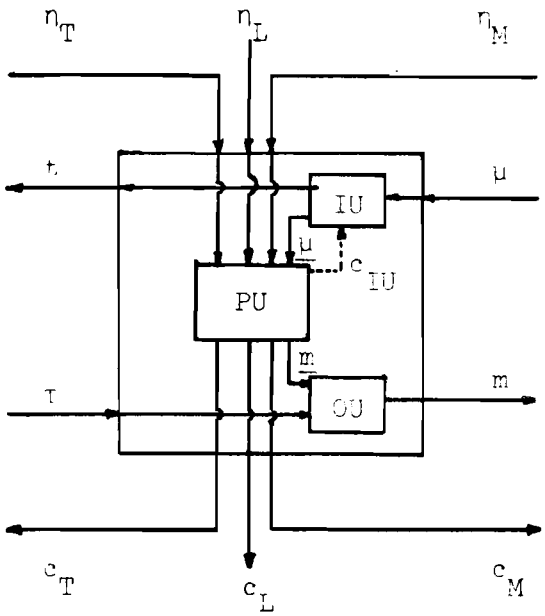


Fig. 8 - Internal structure of an interlocutor.

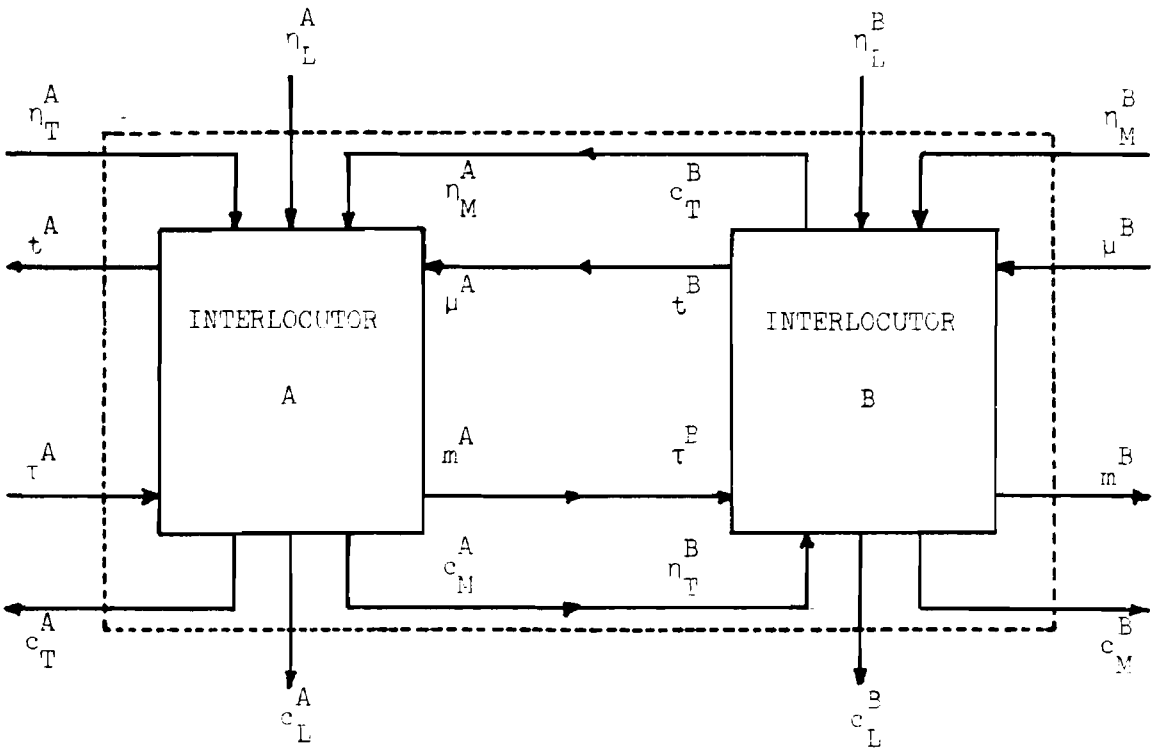


Fig. 9 - M-T interconnection of two interlocutors A and B.

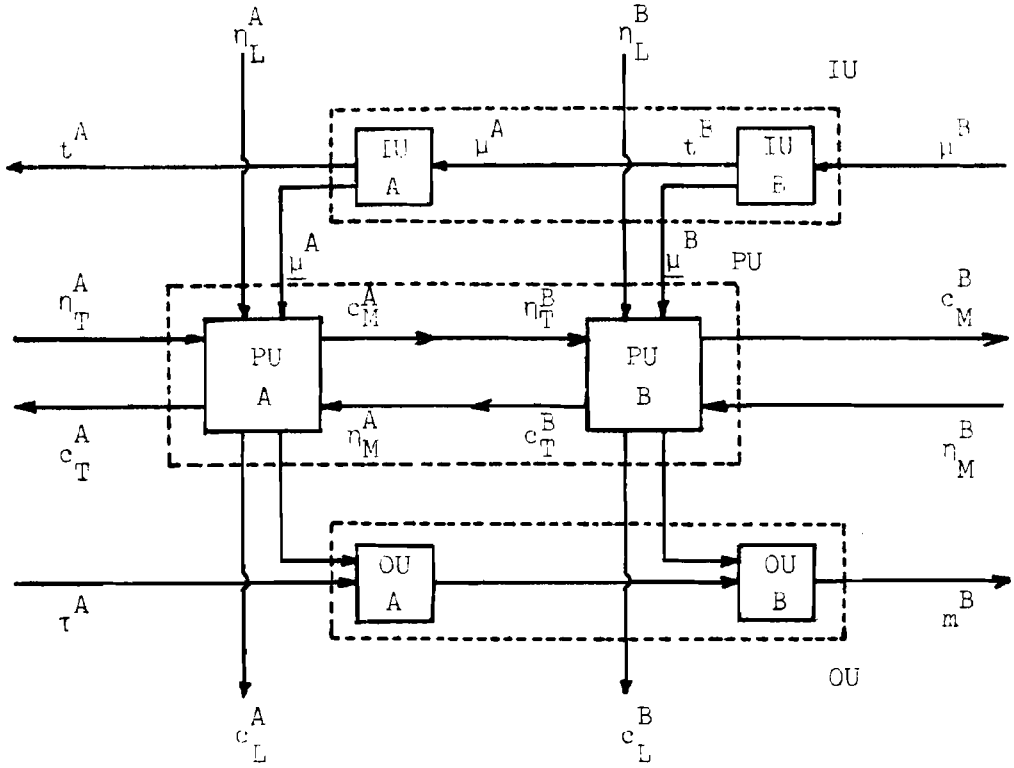


Fig. 10 - Resultant PU, IU and OU of a composite interlocutor.

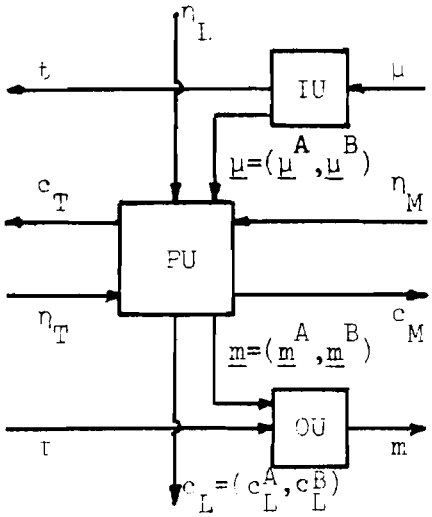


Fig. 11 - Internal structure of a composite interlocutor.

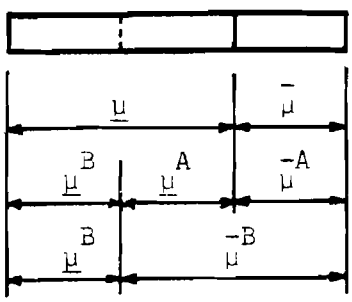


Fig. 12 - Message structure in a composite interlocutor.

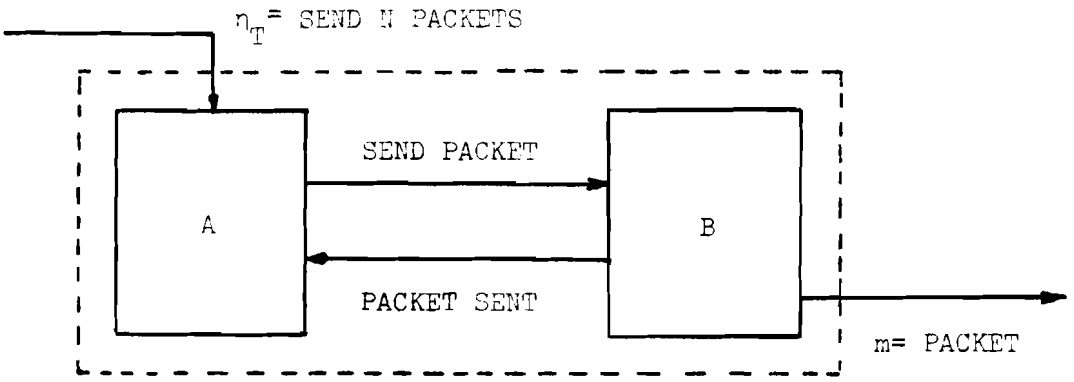


Fig. 13 - Example of composite interlocutor.

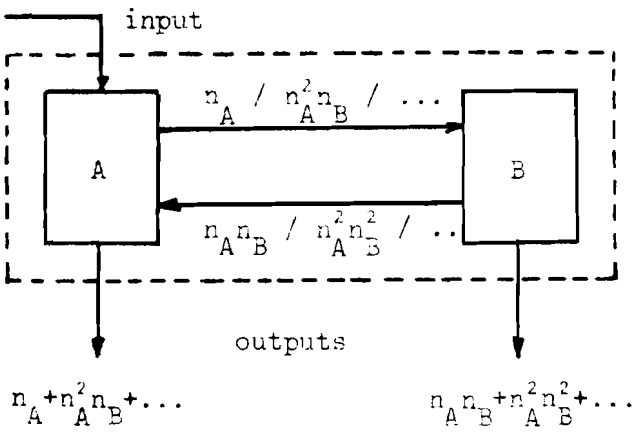
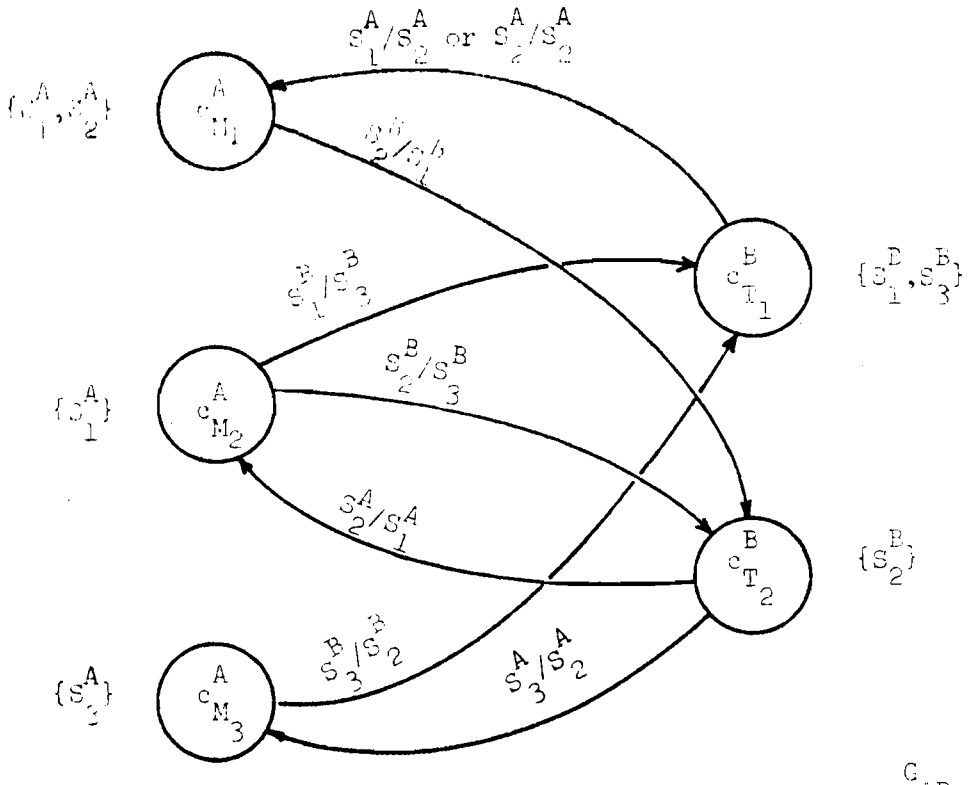
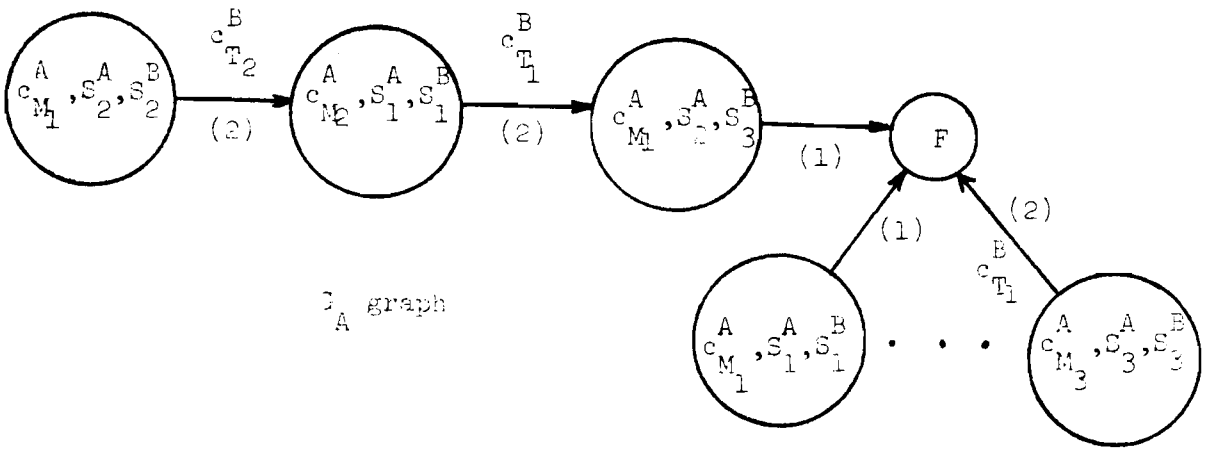


Fig. 14 - Segment lengths to evaluate the number greater than the order.



G_{AB} graph



G_A graph

G_B graph

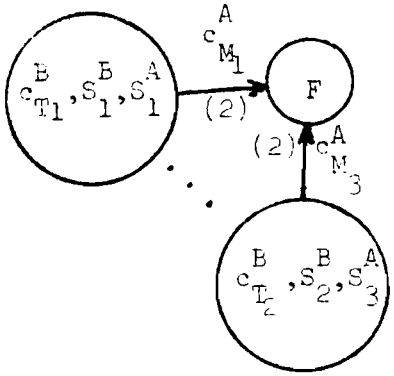


Fig. 15 - Example of graphs G_{AB}, G_A and G_B.

REFERENCES

[1] Sunshine, C.A. (1978) Survey of Protocol Definition and Verification Techniques. Proceedings of Symposium on Computer Network Protocols, Liege, 13 - 15 February. Liege: University of Liege.

[2] Le Moli, G. (1973) A Theory of Colloquies. Alta Freqnenza 10(XLII):223 E - 230 E, October.

[3] Le Moli, G. (1973) Colloquies in Computer Networks. Presented at International Advanced Study Institute on Computer Communication Networks, Brighton, 9 - 15 September. Noordhoff Leyden, 1975.

[4] Danthine, A. and J. Bremer (1975) Définition, Représentation et Simulation de Protocoles dans un Contexte Réseaux. Journ. Intern. Mini-ordinateur et Transm. de Données: 115 - 126. Liege: AIM, January.

[5] Danthine, A. and J. Bremer (1976) An Axiomatic Description of the Transport Protocol of Cyclades. Professional Conference on Computer Networks and Teleprocessing, Aachen, 31 March - 2 April.

[6] Faro, A., G. Le Moli and E. Repossi (1977) Specifications of the Subnetwork Control Module for EIN. EIN/CREI/77/007. Teddington, Middlesex: European Informatics Network.

[7] Alfonzetti, S., S. Casale, A. Faro and S. Palazzo (1979) Algorithms to Pass Between Various Protocol Formalizations: from Logical Matrices to Variable Structure Sequential Machine and Vice Versa. Alta Frequenza 2 (XLVIII):28 E - 37 E, February.

[8] Alfonzetti, S., S. Casale and A. Faro (1979) A Formal Description of the DTE Packet Level in the X.25 Recommendation. Alta Frequenza 8(XLVIII):339 E - 348 E, August.

[9] Rinaldi, S. (Editor)(1974) Teoria degli Automi. Milan: CLUP.

PROTOCOL PARAMETERS AND NETWORK CHARACTERISTICS: CLASSIFICATION AND SOME INTERRELATIONS

A. Butrimenko, G. Scollo

The purpose of this paper is to study closely the class of problems that arise in the interconnection of different computer systems through a packet switching network.

The layered protocol's architecture is assumed to separate functionally and to identify the tasks to be performed in various parts of the network, either in the packet switching subnetwork, or in the end processors. Concepts are then introduced to identify the characteristic parameters of each protocol layer. A further step is taken to consider a sample architecture built on well-known protocols at different levels, up to the transport level, and to develop an analysis of their interaction in order to identify interdependencies and constraint relations on the values of the characteristic parameters.

INTRODUCTION

As computer networks continue to have an increasing impact on communications and resource sharing, the need is growing for a universally acceptable method of describing the means by which computer systems of different size and manufacture, and displaying different features, connected by a single network, can "speak" to each other.

The trend of establishing public data networks raises international standards of computer communication and of "open networking". Once a satisfactory set of standards has been agreed upon, any digital device, using the minimum amount of hardware/software resources required to comply with the standard rules of the "colloquy", can call any other and interact with it.

The discussion on "Open Systems Interconnection" has already led international standardization bodies to issue a draft proposal for the formulation of a vocabulary [1] and, what is more relevant to the purpose of this paper, a "Reference Model" [2], to be taken into account for the implementation of Open Systems. The architecture of the Reference Model consists of layered functions based upon certain major layering concepts, some of which are shown in Figure 1.

For the purpose of computer communications, a set of standards has been established over the last few years, applying to the lower levels. For example, the X.25 Recommendation [3] of the International Telegraph and Telephone Consultative Committee (referred to after this as CCITT) covers the lowest physical level of protocols, a bit-oriented protocol at the second link level (HDLC) and a packet exchange protocol at the third level, intended to guarantee the reliable and sequential transfer of "packets", i.e., data-units of maximum agreed length, across the physical interface between the computer - or DTE: Data Terminal Equipment - and the access point of a public data network - or DCE: Data Circuit Terminating Equipment (Figure 2). However, the requirements for the reliable transfer of larger data-units through the network from one DTE to another, the recovery from network failures, and the selection from among different communication "modes" of those which fit in with a variety of user traffic patterns, should all be accomplished by a fourth layer of end-to-end protocols. This layer is called the "Transport Layer" in the Reference Model.

The long debate which has preceded, accompanied and followed X.25 is already well-known [4]. The most controversial point has been the fact that X.25 networks provide their users with a virtual-circuit (VC) service, at the expense of a more complex implementation of the network interface in the DTE, and also cause a decrease in the efficiency of the communication subnetwork [5]. The need for complementary standardization of a simpler interface, called "Datagram" (DG), has been expressed by many areas and is now being studied by various standardization bodies.

At present, no standard end-to-end protocol exists. Out of the various proposals, the document of the International Networking Group (referred to after this as INWG) 96.1 [6] - even although a draft proposal - is of special interest, for the following reasons:

a) It is intended to be independent of the data transmission service characteristics, i.e., it can be implemented on top of either a Datagram service, or a Virtual Circuit (switched or permanent) service, or a Real Circuit (HDLC) service,
b) Experience of its implementation does already exist; a subset of it - only in "liaison mode" - is the end-to-end protocol of the CYCLADES network [7], and a version very close to it has already been implemented on top of a Datagram service and used as a basis for higher-level protocols - up to the Application Layer - in an international experimental network, the European Informatics Network [8].

In the rest of this paper, the following "Sample Architecture" (a partial one, i.e., up to the Transport Layer) will be taken into consideration:

Hyp. 1. A packet switching network provides its users (DTE's) with an X.25 interface to the Data Transmission Service;
Hyp. 2. The INWG 96.1 is implemented on the DTE's to perform the Transport Layer functions.

It is the opinion of the authors that the interconnection of the protocols belonging to the different layers can have quite a strong influence on the actual implementation of each of them; it is felt that the values of some characteristic parameters of each protocol should be tuned - and, perhaps, dynamically updated - taking into account the available information on what is outside-the-border of the protocol layer. Quoting from the Reference Model as an example, it is found that:

> "The Transport Layer is required to optimise the use of the available communications resources to provide the performance required by each communicating transport user at the minimum cost. This optimization will be achieved within the constraints imposed by considering the global demands of all concurrent transport users and the overall limited resources available to the Transport Layer".

CHARACTERISTIC PARAMETERS OF A PROTOCOL

A variety of methods, generally speaking, can be used to describe a protocol; from the least formal, such as word description, to the most formal methods, such as automata, grammars, etc., a wide spectrum of linguistic tools can be drawn upon. The importance of the word description is, however, fundamental, as the purpose of a protocol description is that it can be correctly implemented. For this reason, a correct and as simple as possible understanding is required by the implementors.

The word description can be accompanied - but not substituted - by a more formal description, in order to avoid the misunderstandings that the ambiguity of natural language may generate.

In every protocol description - be it a more or a less formal one - a set of variables can be found, the values of

which do not determine the nature of the description itself (it is usually sufficient to mention their existence), but which can play a fundamental role in every implementation of the protocol. A few explanatory points can lead to more precise conceptual definitions:

1. The formalisation of a protocol carried out by means of automata theory requires a number of "states" to be assigned to the Finite State Machine (FSM), representing the couple of interlocutors that follow the rules of the protocol. In order to avoid deadlocks (and, in practice, wastage of resources), most of the states of the FSM must be transient, i.e., they have to be protected by a time-out, "T". A time-out expiry may generate a state transition: however, it usually happens after a number of retries, "N", of the same action. For each trial, T is started again, usually with the same value (but this is not mandatory; it is just common practice).

   In the literature available, there are different ways of taking into account the T expiry and/or the reaching of N; for example, Le Moli [9] refers to these as "internal events", whilst Danthine and Bremer [10] treat them as equivalent to existing or newly introduced inputs.

   A completely formal description of the protocol must, however, indicate what actually constitutes the "transient" feature of its transient states. In the following, this parameter set will be referred to as "T-parameters".

2. In a layered architecture of protocols, each layer can be considered as the "communication device" of the next higher layer [9]. As such, it is characterised by a set of specification parameters that refer to the quality of service that it can provide to the next higher layer, with some environmental constraints. Throughput, response time, introduced delay, level of reliability, security, level of availability, are all concepts which require an unambiguous definition of a set of associated characteristic parameters

in the protocol specification, at least in the non-formal one.

Moreover, closely related to these concepts, the implementation of a "function" (for example, the techniques adopted for error control and recovery, fragmentation and reassembly, multiplexing, sequencing, and so on) introduces a new kind of parameter set that refers to the price to be paid - relevant to the protocol - to perform that function in terms of overheads for address and control information, either in the header-portion of message-carrying data, or in "special" messages carrying no data and invisible to the next higher level. In other words, the service offered by a layer to the next higher one, and the performing of the functions needed to offer that service, can be analysed in terms of cost/benefit ratio. As such, two distinct, but related, sets of parameters can characterise the performance of the layer, each set referring to each term of the ratio. In the following, these parameter sets will be referred to as "C-parameters" and "B-parameters", respectively. It is also worth stressing that hitherto, only those parameters have been considered that refer to the protocol definition and operation, and not those that refer merely to a local interface among layers.

3. Finally, each layer implementation requires a set of parameters to be stated and - statically or dynamically - assigned a value, in order to make the best use of the resources and services offered by the next lower layer. It will be said that these parameters have dynamically assignable values if information on remote events is needed in order to pursue this optimization task, and if the information is available from the interface to the next lower layer. If this information is either not needed - for example, parameters related to the network maximum configuration, which is usually fixed, at least in the short/medium run - or if this information is not available, and is therefore surrogated by average estimations - for example,

parameters related to the present network configuration, deduced from the routing tables, which are seldom available to the DTE processes - these parameters will be said to have statically assignable values.

Once again, it should be underlined that, even although these parameters refer to the local interface to the next lower layer, they will be taken into account if, and only if, their meaning is relevant to the protocol definition and operation. As the meaning of these parameters must always be related to the state of the "network", i.e., of what, globally, is under the layer, they will be referred to as "N-parameters".

In conclusion, the concepts hitherto developed can be grouped according to the following rationale:

The correct definition and the effective operation of a protocol in the layer L require certain protocol characteristic parameters to be stated and suitable values to be - statically or dynamically - assigned to them. Our classification of the protocol characteristic parameters identifies the following sets:

a) T-parameters, defined to protect the transient states of the protocol,
b) B-parameters, defined to provide the layer L+1 with a specified quality of service,
c) C-parameters, defined to evaluate the cost, in terms of overhead, of implementing those functions of the protocol necessary to provide the layer L+1 with the L-layer's services,
d) N-parameters, defined to make the best use of the network resources in the operation of the functions considered above.

From the definitions of these classes of parameters, it follows that they have empty intersection.

A variable declared in the implementation will be considered here as a protocol characteristic parameter only if it is semantically both relevant to the protocol definition and operation, and assimilable into one of the four classes defined above.

In [11] a consequent analysis was conducted to identify and explicitly specify the parameters involved in the X.25 implementation. It is not possible to go much more deeply into the specification of each parameter.

Tables 1 and 2 summarize the characteristic parameters hitherto defined for the physical and link layers, and the network layer, correspondingly. Their classification, following the concepts expressed in Section 2, is also given. It is the opinion of the authors that this set is far from being complete. One reason is "structural", in that the X.25 level 3 specifies only the interface between the DTE and DCE for packet transfer that is a part of the whole network layer. The network layer is, however, spread all over the network in other functions, such as routing, local flow control, acknowledgement between DCE's, and so on. Strictly speaking, the parameters of the topology of the subnetwork also belong to this layer. This matter will be treated in Section 5. In any case, the parameter set strictly referring to the third level interface between the DTE and DCE could also be "improved".

The INWG 96.1 document is a draft revision of a proposal (INWG 96) submitted to ISO as an International Federation for Information Processing (IFIP) contribution for a standard end-to-end protocol. The earlier proposal was based on a datagram data transmission service, whilst the revised version takes into account the need for adaptation to a variety of data transmission facilities, paricularly X.25 in public data networks. No formalization of this transport protocol is given in the document. The elements constituting the transport service are defined independently of the transport protocol mechanisms used to provide them. This is very useful for finding out what is called here the class

of B-parameters, and also some C-parameters.

Moreover, the document specifies the combinations of those functions that lead to different classes of overall service. Three classes of transport service are defined, namely: "lettergram", "regular liaison" and "super liaison". Consequently, the transport protocol characteristic parameters are defined in [11] with explicit reference to the class of the transport service within which they are relevant.

Table 3, which reproduces Figure 21 of the INWG 96.1 document [6], indicates the elements of the transport services that define the transport service classes; a short notation is added to the original Figure for the purpose of abbreviation in the next table.

Table 4 summarizes the characteristic parameters hitherto for the transport layer; it also shows their classification, following the concepts expressed in the above and the transport service classes and elements to which they are relevant.

As already mentioned in connection with the network layer parameters, this set may not be complete, also because some points in the INWG 96.1 document are "left for further study" (e.g., the specifications of "check sum on letters", negative acknowledgement, etc.).

THE SAMPLE ARCHITECTURE

The analysis hitherto developed has, among other things, taken into account the fact that each layer is "part of a whole"; for each layer, characteristic parameters have been defined in such a way that the following is known:

- how its transient states are protected,
- how it can use the services and resources available from the next lower layer,
- how much overhead is introduced in order to perform its functions,

-- how its services, provided to the next higher layer can be evaluated.

However, if the whole network architecture is taken into consideration, a first observation must be made: the analysis developed sofar is not adequate as no assumptions have been made about the way in which the DCE to DCE transfer of information takes place (Figure 3). Regarding this "questionmark" in the Figure, the following general working hypotheses have been made in this paper:

Hyp. 1. A packet switching subnetwork takes responsibility for the transfer of information between the DCE's,

Hyp. 2. A distributed, adaptive, minimum-delay routing algorithm is defined in the subnetwork operation.

We will assume two possible modes of implementation:

a) A datagram-type service which routes every packet separately, and

b) The establishing of a virtual call which presumes that all packets of the same call are sent over the same route.

Hyp. 3. The "regime" topology of the subnetwork (i.e., when all nodes and lines are available and not congested) is known.

This is a preliminary step that had to be taken for the sake of completeness of the Sample Architecture: the routing functions will be considered as belonging to the set of functions performed on the network layer of each node.

As the Sample Architecture is drawn in a general configuration, it is now important to consider the consequences that the interconnection of different layers will have on the definition for each layer of the characteristic parameter set. This analysis will be made in the following part of this Section for some of the "qualitative" aspects of the interconnections; in the next Section some "quantitative" interrelations will be deduced.

It should be underlined once more that only the matching of the Transport Layer specifications, given in INWG 96.1, with those of the Network Layer in the DTE, given in the X.25 level, is considered in this paper.

In an X.25 network, the available data transmission service between two Transport Stations can be:

1. a (set of) permanent virtual circuit(s): PVC
2. a (set of) switched virtual circuit(s): SVC

In both cases, in order to establish port associations, the Transport protocol may or may not be required to perform the multiplexing of the circuit(s) between its users. Therefore, the selection of the transport service elements that actually need to be put into operation depends strongly on what kind of data transmission service is used (PVC or SVC), and how it is used (multiplexed or non-multiplexed).

In the following, as a working hypothesis, it will be assumed that:

Hyp. 4. $N_V(a_T)$ switched virtual circuits are available to the Transport Station, the address of which is a_T; the Transport Station is required to perform the multiplexing of the virtual circuit between ports that have to be associated with remote ports belonging to the same remote Transport Station (i.e., are on the same DTE address).

Also for the liaison initialization/termination element of service a certain choice must be made, i.e., as to whether the exchange of LI-INIT messages has to be performed only on virtual circuits which have already been set up, or not.

It will be assumed here as a working hypothesis that:

Hyp. 5. the initialization of a liaison is always tried only on a virtual connection (of either type) which has already been established.

The main reason behind these assumptions is that it has been considered preferable in this paper to separate completely the connection between the Transport Station, i.e., the establishment of an X.25 DTE-DTE virtual connection, from the connection between Transport Users, i.e., a TS Port-Port Association.

With regard to the transport of letters, even if the data transmission service can provide the sequential delivery of packets having the more-data bit set to one, the need for fragmentation can still arise in order to avoid the monopoly of the virtual connection by long letters at the expense of other processes (on other TS ports) sharing the same virtual connection: the need for fragmentation is therefore a direct consequence of Hyp. 4. Moreover, when Error Control is in operation, the need for acknowledgement is a result of the fact that the updating of the lowest window edge, P(R), between the DTE and DCE has only a local meaning. However, if the Transport Stations have to operate only on a network that gives end-to-end significance to P(R), there is no need for further acknowledgement: each TS must only be "informed" by the next lower layer whenever a reset has occurred on a local logical channel.

The transport of telegrams can be mapped directly onto the X.25 level 3 transfer of interrupts. It is the opinion of the authors, however, that the need for telegram acknowledgement with a time-out/retransmission mechanism should be more fully discussed: in fact, it should be noted that retransmission of an unconfirmed interrupt can only take place after a reset of the logical channel has been performed, whether the significance of the interrupt confirmation packet be local or end-to-end.

Finally, the optional performance of End-to-End Flow control should be maintained as this element of the service is not provided by the X.25 data transmission service.

SOME INTERDEPENDENCIES BETWEEN CHARACTERISTIC PARAMETERS

In this Section, a short analysis will be made of the interdependencies that necessarily arise in the evaluation or assignation of values to characteristic parameters of different protocol layers. A complete and detailed analysis of all the parameters is far from the intention of the authors: only a few of them (marked with an asterisk in Tables 1, 2 and 4) will be taken into account in order to show how, for sample features of the architecture, modeling or experience, or (better) both, can be used as tools for the designers or the implementors of an architecture.

As a first example of this method of reasoning the following approach to the problem of the agreement of the values for the T-parameters, T1 and N2, of the X.25 link level interface between DTE and DTE proposed in the Recommendation X.25, can be considered.

As this timer is started from the DTE (DCE) on the transmission of an I-frame, no waiting time for DCE (DTE) busy condition clearing has to be taken into account. The worst occurrence is when the longest frame (containing N1 bits) is transmitted by the sender and is acknowledged after the maximum time, T2, by the receiver by means of an acknowledgement, piggybacked into the longest information frame. The following relation can be used:

$$T1 = T2 + 2\left(\frac{N1}{C1} + d^*\right) , \qquad (1)$$

where d* has to be a pessimistic estimation of additional delays introduced by modems, propagation time, and other possible secondary factors. The following relation, as anticipated in Section 3.2, takes place by definition:

$$N1 = N3 + H_M , \qquad (2)$$

and by the specifications of the HDLC frame format, the following relations also take place:

$$H_M = (N3 + H_m - 16)/5 + H_m \quad , \tag{3}$$

where this value of H_M can be achieved if the whole frame consists of ones. It is also clear that H_m includes flags (16 bits), address and control fields (16 bits) and checksum field (16 bits), and is always present. Therefore:

$$H_m = 48 \quad . \tag{4}$$

As a first result, we get:

$$T1 = T2 + 2(1.2N3 + 55)/C_1 + 2d^* \quad . \tag{5}$$

The evaluation of the maximum number of transmissions of the same frame, N2, should take into account the error rate of the line, from the one side, and the link level average delay, D_L and availability A_L (B-parameters), from the other side. It is quite obvious that increasing N2 means increasing both: whilst the second is an improvement, an increased delay should, however, be maintained under acceptable values. A_L can be estimated by the following:

$$A_L = 1 - \frac{n_r \bar{d}_r}{t_{op}} \quad , \tag{6}$$

where t_{op} is a defined time of operation of the link, $\bar{d}_r$ is the average time spent for link reset, and n_r is the number of times that the link had to be reset during the time, t_{op}. It can be written as follows:

$$n_r = n'_r + n''_r \quad , \tag{7}$$

where n'_r (N2) is the component of n_r due to N2 unsuccessful transmission of the same frame, and n''_r is the component of n_r due to other reasons (for example the receipt of the invalid frame format), and as a consequence:

$$A_L = 1 - \Delta A'_L - \Delta A''_L \quad . \tag{8}$$

with

$$\Delta A'_L = \frac{n'_r \bar{d}_r}{t_{op}} \quad . \tag{9}$$

$$\Delta A''_L = \frac{n''_r \bar{d}_r}{t_{op}} \quad . \tag{10}$$

The probability p_c that a frame is error-free depends upon the length of the frame, L_F, and the error rate of the line:

$$p_c(E_1, L_F) = (1 - E_1)^{L_F} \quad . \tag{11}$$

The probability, p_r, that a link reset condition is reached after N2 transmissions of the same frame, depends upon the traffic pattern. The worst case is when the link is fully utilized in both directions with frames of maximum length, N1: in fact in this case, the loss of an acknowledgement always generates the retransmission of the frame. This worst case is examined here in order to find an upper limit expression for $\Delta A'_L$ (N2).

In this case, it is true that:

$$p_r(N2) = \left[1 - (1 - E_1)^{2N1}\right]^{N2} \tag{12}$$

In fact, it is also true that:

$$p_r(N2) = \left[1 - p_c^2(E_1,N1)\right] p_r(N2 - 1) \quad ,$$

$$p_r(1) = 1 - p_c^2(E_1,N1) \quad ,$$

and, from (11), relation (12) follows.

The probability, p_s, that a frame will be successfully transmitted and acknowledged on the M-th trial ($M \leq N2$), always being in error in the preceding M-1 trials, is:

$$p_s(M) = \left[1 - p_c(E_1,N1)\right]^{(M-1)} p_c^2(E_1,N1) \quad ,$$

and again, from (11) it follows that:

$$p_s(M) = \left[1 - (1 - E_1)^{2N1}\right]^{(M-1)} (1 - E_1)^{2N1} \quad . \qquad (13)$$

The loss of a frame sent from A to B generates the "waste" of time T1, in the direction A → B, and the waste of time t_{wr} in the direction B → A, where

$$t_{wr} = T1 - \left(\frac{N1}{C1} + d^*\right) = \frac{T1 + T2}{2} \quad ,$$

or

$$t_{wr} = T1 \quad ,$$

if the succeeding frame is also lost. The sum of the average times wasted in both directions for the loss of a frame is therefore:

$$\bar{t}_w = T1 + p_c \frac{T1 + T2}{2} + (1 - p_c)T1 \quad , \qquad (14)$$

and from (11)

$$\bar{t}_w = 2T1 - (1 - E_1)^{N1} \frac{T1 - T2}{2} \quad . \tag{14'}$$

Then the average time, $\bar{t}_s$, spent (in both directions) to transmit a frame successfully, can be calculated as:

$$\bar{t}_s = \sum_{i=1}^{N2} p_s(i) \left[(i - 1)\bar{t}_w + \frac{N1}{C1} + d^* \right] \quad . \tag{15}$$

And from (1), (13) and (14'), it follows that

$$\bar{t}_s = (1 - E_1)^{2N1} \sum_{i=1}^{N2} \left[1 - (1 - E_1)^{2N1} \right]^{(i-1)}$$

$$\left\{ (i - 1) \left[2T1 - (1 - E_1)^{N1} \frac{T1 - T2}{2} \right] + \frac{T1 - T2}{2} \right\} \quad . \tag{15'}$$

The time, t_r, spent in unsuccessfully transmitting a frame that causes the reset of the link is:

$$t_r = N2 \cdot T1 \quad . \tag{16}$$

When $n_r'' = 0$, it can be assessed that

$$2t_{op} = 2n_r'(t_r + \bar{d}_r) + n_s \bar{t}_s \quad , \tag{17}$$

where n_s is the number of successfully transmitted frames during t_{op} (in both directions).

As it can be measured as follows:

$$p_r = \frac{n_r'}{n_r' + n_s} \quad , \tag{18}$$

(17) becomes:

$$2t_{op} = 2n'_r(t_r + \bar{d}_r) + \frac{1 - p_r}{p_r} \; n'_r \; \bar{t}_s \quad , \tag{19}$$

that is:

$$n'_r = \frac{t_{op}}{t_r + \bar{d}_r + \frac{1 - p_r}{2_{pr}} \bar{t}_s} \tag{19'}$$

Introducing (19') and (16) in (9), it becomes finally:

$$\Delta A'_L = \frac{\bar{d}_r}{\bar{d}_r + f_a} \quad , \tag{20}$$

with

$$f_a = f_a(p_r, \bar{t}_s) = \frac{1 - p_r}{2_{pr}} \bar{t}_s + t_r \quad , \tag{21}$$

and here the following are recalled:

$$t_r = t_r(N2, T1) = N2 \cdot T1 \quad ; \tag{22}$$

$$p_r = p_r(E_1, N1, N2) = \left[1 - (1-E_1)^{2N1}\right]^{N2} \quad ; \tag{23}$$

$$N1 = N1(N3) = 1.2N3 + 55 \quad ; \tag{24}$$

$$T1 = T1(T2, N1, C_1, d^*) = T2 \div 2(N1/C_1 + d^*) \quad ; \tag{25}$$

$$\bar{t}_s = \bar{t}_s(E_1, N1, N2, T1, T2) = \text{v. (15')} \quad . \tag{26}$$

$\bar{t}_s$ expresses also, in the worst case examined here, the average value of the component of the delays D_L(DTE → DCE) and D_L(DCE → DTE) - in each transmission direction - due to the transmission process. The actual value of this delay must take into account the waiting times due to "DCE busy" and "DTE busy" conditions, respectively. However, from (8), (9), (20), (21), (22), (23) and (26), one can easily see that the increase in N2 means, not only an increase in the availability, but also an increase in the delay.

It is now time to make a study from an upper-level point of view. An interesting problem can be the evaluation of inter-relations between the time-outs that regulate the opening of a liaison, and the B-parameters of this element of the transport service, such as the delay, D_{TEL}, and the availability, A_{TL}.

The setting of proper values for the time-outs, T_{LLO}, T_{LRO} (Table 4), and TTp2, TCp3 (Table 2), can be established, taking into account some parameters of the distributions of the proper components of the delays D_{TEL} and D_{pe} (Table 2), respectively - on one side - and some objective values for the availabilities, A_{TL} and A_p (Table 2), respectively - on the other side. The availability depends both on local factors, e.g., the number of buffers and the average and peak-hour number of concurrent connections (provided by the third level) or associations (provided by the fourth level), and on network factors, e.g., the influence of the load of the subnetwork on its component to the delay. If:

1. d_{lpA}, d_{rpB} are random variables that represent the local third level delays of DTE_A, DTE_B, respectively, for the establishment of a locally-, remote- (respectively) requested connection,
2. d_{rTB} is the random variable that represents the local TS_B delay in accepting the establishment of the remote-requested connection,

3. $d_s(A,B,L_p)$ is the random variable that represents the delay introduced by the subnetwork for the transfer of a packet of length L_p from DCE_A to DCE_B (third level delays included),
4. $d_{lTC}(A,L_p)$, $d_{lCT}(A,L_p)$ are random variables that represent the link level delay from DTE_A to DCE_A, from DCE_A to DTE_A, respectively, for the transfer of a packet of length L_p,
5. d_{peAB} is the random value of the actual delay for the establishment of the virtual connection between TS_A and TS_B, requested by TS_A,

it is true that:

$$d_{peAB} = d_{lpA} + d_{rpB} + d_{rTB} + d_s(A,B,L_{RI}) + d_s(B,A,L_{AC}) + \\ + d_{lTC}(A,L_{RI}) + d_{lCT}(A,L_{AC}) + d_{lTC}(B,L_{RI}) + d_{lCT}(B,L_{AC}) \quad , \tag{27}$$

where L_{RI} is the length of the packets' Call Request and Incoming Call, and L_{AC} is the length of the Call Accepted and Call Connected packets. In order that the connection can actually be established, it is necessary that the following takes place:

$$TTp2_A > d_{peAB} - d_{lpa} = d_1 \quad ; \tag{28}$$

$$TCp3_B > d_{lTC}(B,L_{RI}) + d_{lCT}(B,L_{AC}) + d_{rpB} \\ + d_{rTB} = d_2 \quad . \tag{28'}$$

The availability, A_p, can be expressed as follows:

$$A_p = 1 - \Delta A'_p - \Delta A''_p + \Delta A'_p \, \Delta A''_p \quad . \tag{29}$$

where $\Delta A'_p$ represents the probability that the connection will not be established because either (28) or (28') is not respected, and $\Delta A''_p$ represents the same, but for different reasons (e.g., lack of buffers). $\Delta A'_p$ can depend, in its turn, on the number n_c

of concurrent connections on each of the two DTE's. It can be foreseen that it is a function growing with n_c. With the position:

$$\Delta A'_{p\phi} = \Delta A'_p(n_c = \phi) < \Delta A'_p(n_c > 0) \quad , \tag{30}$$

it can be established that:

$$\Delta A'_p = \Delta A'_{p\phi} + \Delta A'^*_p \quad . \tag{31}$$

If the delays that appear in (28), (28') are examined under the hypothesis that no other concurrent virtual connections will be (or are being) established, both on TS_A and on TS_B, the component, $\Delta A'_{p\phi}$ of A_p can be estimated.

Let us denote

$$d_{12} = d_1 - d_2 \quad . \tag{32}$$

From (28) and (28') it follows that:

$$d_{12} = d_s(A,B,L_{RI}) + d_s(B,A,L_{AC}) + d_{lTC}(A,L_{RI}) + d_{lCT}(A,L_{AC}) \quad . \tag{33}$$

If nothing can be assessed about the distribution of d_2 and d_{12}, except that they have average values $\bar{d}_2$ and $\bar{d}_{12}$, and variances S_1^2 and S_{12}^2, respectively, and that they are mutually independent (this is a very broad hypothesis, which assumes that no other traffic is generated by both DTE_A and DTE_B on the subnetwork), then Kolmogorov's inequality can provide an upper bound to $\Delta A'_{p\phi}$. In fact, it states, in this case, that - for every $t > 0$ - the probability of simultaneous realization

of the inequalities

$$|d_2 - \bar{d}_2| < t \cdot s_1 \quad ; \tag{34}$$

$$|d_1 - \bar{d}_1| < t \cdot s_1 \quad . \tag{34'}$$

is at least $1 - t^{-2}$ (being $\bar{d}_1 = \bar{d}_{12} + \bar{d}_2$; $S_1^2 = S_2^2$, for the assessed stochastic independence of d_{12} and d_2). Considering, for obvious reasons, only the case in which $d_{12} > \bar{d}_{12}, d_1 > \bar{d}_1$, it follows that:

$$1 - \Delta A'_{p\phi} \geq 1 - t^{-2} \quad .$$

That is:

$$\Delta A'_{p\phi} \leq t^{-2} \quad ; \tag{35}$$

if

$$TCp3_B = \bar{d}_2 + tS_1 = \bar{d}_2 + t\sqrt{Var(d_{pe})} \quad ; \tag{36}$$

$$TTp2_A = \bar{d}_1 + tS_1 = D_{pe} + t\sqrt{Var(d_{pe})} - d^*_{lpA} \quad . \tag{36'}$$

where d^*_{lpA} is a constant, evaluating the minimum value of d_{lpA}. If the distributions of d_1 and d_2 can be approximated by well-known distributions, better interrelations can be found. It is, however, confirmed that wide availability and small delay are opposite requirements, between which a balance should be not only made in the design phase, but should also be controlled and updated in the operational phase.

The availability of the transport service can be expressed also as:

$$A_{TL} = 1 - (\Delta A'_{TL\phi} + \Delta A'_{TL}) - \Delta A''_{TL} + \Delta A''_{TL}(\Delta A_{TL\phi} + \Delta A'^*_{TL}) \quad , \tag{37}$$

where the meaning of the components is analogous to that indicated for the third level. It is true that:

$$\Delta A'_{TL\phi} = \Delta A'_{p\phi} + (1 - \Delta A'_{p\phi})\Delta A'_{LV\phi} \quad ,$$

where $\Delta A'_{LV\phi}$ represents the probability that the liaison will not be established once the connection has been established because one of the time-outs, T_{LLO},T_{LRO}, expired for the excessive value of the corresponding delay. It is left to the discretion of the reader to apply theoretical tools to this case also.

In order to indicate another case of interdependency between time-outs and other parameters of different layers, the "transport of letters" element of the transport service will be considered as the last (but not least) case.

As an example, we can look at the time delivery of user packets in the liaison mode in the INWG 96.1, and in particular, parameter T_{LE} of the sending station, and parameter T_{RS} of the receiving station.

If the T_{RS} expires, and no new frame of the letter arrives, the letter is considered to be lost, and the letter should be retransmitted. Waiting time, T_{LE}, is set up at the sending station for expected acknowledgement from the receiving station.

It is quite obvious that, if T_{LE} if too small, this will lead to repetition of a letter, which can be particularly dangerous when the network is overloaded; therefore, if too small a T_{LE} is set up, the load will be unnecessarily increased. If T_{LE} is too great, this will slow down the whole system. Again, if T_{RS} is too small, this will lead to unnecessary interruption of the reassembly process, and if T_{RS} is too great, this will block the resources of the receiving station.

Similar problems arise also in the lettergram mode with T_{GE} and T_{RS} parameters. If T_{GE} is too small, there is a stronger possibility of the lettergram being considered lost, when it in

fact is not, and unnecessary actions can be caused at the higher level. The secondary effect of the inadequate setting of T_{GE} or T_{LE} is that it leads to incorrect counting as errors, of all cases when time-out expires. The letter is, however, actually delivered and ACK is not lost, but comes in later.

In the case that the setting of T_{GE} and T_{LE} is too strict, this causes an unjustified increase in $\bar{E}_{T1M}$ and $\bar{E}_{T1R}$ (or E_{T1S}), respectively. In another set of B-parameters like R_{TGE}, R_{TLE} and R_{TLF}, which expresses the reliability of the transport service, too strict a setting of T_{GE} and T_{LE} will cause an unjustifiable decrease.

If we consider the transport service between stations A and B, which we assume, for simplicity, to be connected with the switching nodes - DCE_A and DCE_B - then T_{LE} depends, among other parameters, on the sum of t_B^A and t_A^B, where t_j^i is the delivery time of a packet from the node i, to the node j. If the routing mechanism is based upon the packet delivery time, as is assumed above in our model (Hyp. 2), e.g., "relief", or as it is used in the ARPA network, then t_j^i is just an entry of the corresponding routing matrix.

We do not know, however, t_j^i in node i, but a reasonable estimation of t_j^i can be obtained from t_i^j, assuming that they are equal. Some experimental tests have shown that this assumption is feasible in most cases, and is more applicable to large networks than to small ones.

In order to estimate T_{RS}, again information from the routing matrix can be used. If receiving station B knows the t_A^B delivery time of a packet from A to B, or can estimate it on its own entry, t_B^A, then the following considerations could take place. As soon as a frame of a letter has been received, an estimation of the interval before the arrival of the next frame can be made. This interval depends on the time interval between the generated frames and some function of the distance and delays due to the queues in the network $t(1_B^A, d_B^A)$, where 1_B^A is the distance and d_B^A is the delay caused by the queues.

If we know that the minimal distance is equal to one and therefore the whole queueing delay is concentrated at one node only, we can estimate that the variance of the delay will be:

$$D(d_B^A) = d_B^A + (d_B^A)^2 \quad .$$

If the delay is distributed over n transit nodes, and if we assume that the delays are evenly distributed over all n nodes, we can calculate the variance of the delay as:

$$D(d_B^A) = \sum_{i=1}^{n} D_i\left(\frac{d_A^B}{n}\right) = d_A^B + \sum_{i}^{n}\left(\frac{d_A^B}{n}\right)^2 ,$$

where D_i is the variance of delay at node i, and as we know, is smaller than the value of the previous expression for the concentrated queue.

So if we explicitly know the delay between nodes, we can assume that the variation in this delay will be less for a long distance than for a short one, if the delay is the same. The estimation of the variance also allows one to set up time-out, to cover the risk of statistical oscillation of delivery time.

If INWG 96.1 is implemented on top of the X.25, as we assume in our reference model, and if X.25 is implemented in such a way that the actual route of the packets belonging to the same call is fixed for the whole duration of the call, the delivery time of a packet increases according to its number in the succession [5]. This dependence on the number of the packet is caused, as has been shown in [5], by the decreasing efficiency of the fixed route with the time. The delivery time of every subsequent packet is greater by 5 - 20% than the previous interval between the packets. This percentage increases with the load of the network and the distance, I_A^B, between the communicating nodes. This has the practical consequence that

the estimation of the time parameter, T_{RS}, should take into account the sequence number of the fragments of the same letters, and the load and distance.

CONCLUSIONS

In this paper the authors have made an attempt to study the set of protocols as one entity and tried to identify interrelations between some of the parameters of various layers.

This led to the necessity of defining a number of classes of parameters which go through all the layers. This attempt, although limited to a sample architecture, has shown a large variety of these parameters and the rather complicated interconnections between them. Some of the interrelations found, as well as others which could be found by following a similar methodology, can be used both in the design phase of a computer network and in the operational phase. The authors intend to continue the work begun in this paper in order to achieve a more clear and comprehensive understanding of the interdependence of the parameters.

Table 1. 1st and 2nd level parameters of X.25 interface

	Parameter	Class	Meaning
physical layer	* C_1	B	capacity of the physical link (bps)
	* E_1	B	nominal bit error rate of the physical link
link layer	* T2	N	maximum time from the reception of a frame to the transmission of the acknowledgement
	K	N	maximum number of outstanding Information frames
	* T1	T	timeout for retransmission of unacknowledged frames
	* N2	T	maximum number of transmissions of an unacknowledged frame
	* H_m	C	minimum overhead (bits per frame) of the link protocol
	* H_M	C	maximum overhead (bits per frame) of the link protocol
	H_a	C	average overhead (bits per frame) of the link protocol
	* N3	B	maximum user length of the Information field in an I-frame
	E_L	B	packet error rate, referred to an average packet length $\bar{L}_p$
	C_L	B	maximum throughput, referred to an average packet length L_p in packet per second
	* A_L	B	availability of the link to the third level
	* D_L	B	average delay of the link for the transfer of a frame between DTE and DCE

Table 2. 3rd level parameters of X.25 interface

Parameter	Class	Phase	Meaning
* TTp2	T	Setup	DTE timeout that protects the DTE waiting state p2
* TCp3	T	Setup	DCE timeout that protects the DCE waiting state p3
TTp3	T	Setup	DTE timeout for the acceptance of the incoming call from the 4th level (1)
TCp2	T	Setup	DCE timeout for the acknowledgement of the call request packet from the remote DCE (2)
TTp6	T	Clear	DTE timeout that protects the DTE clear request state p6
NTp6	T	Clear	DTE maximum number of trials to send a clear request
TCp7	T	Clear	DCE timeout that protects the DCE clear indication state p7
NCp7	T	Clear	DCE maximum number of trials to send a clear indication
TTp7	T	Clear	DTE timeout for the acceptance of the clear indication from the 4th level (1)
TCp6	T	Clear	DCE timeout for the acknowledgement of the clear indication packet from the remote DCE (2)
TTi2	T	Transf.	DTE timeout started on the transmission of a DTE-interrupt, cleared on the reception of a DCE interrupt confirmation
TCi3	T	Transf.	DCE timeout started on the transmission of a DCE-interrupt, cleared on the reception of a DTE interrupt confirmation
TTi3	T	Transf.	DTE timeout for the acceptance of the interrupt from the 4th level (1)
TCi2	T	Transf.	DCE timeout for the acknowledgement of the interrupt from the remote DCE (2)
TTd2	T	Reset	DTE timeout that protects the DTE reset request state d2
NTd2	T	Reset	DTE maximum number of trials to send a reset request
TCd3	T	Reset	DCE timeout that protects the DCE reset indication state d3
NCd3	T	Reset	DCE maximum number of trials to send a reset indication
TCd2	T	Reset	DCE timeout for the acknowledgement of the reset from the remote DCE (2)
NCd2	T	Reset	DCE maximum number of trials to send the reset to the remote DCE (2)
TTr1	T	Restart	DTE timeout that protects the DTE restart request state r1
NTr1	T	Restart	DTE maximum number of trials to send a restart request
TCr2	T	Restart	DCE timeout that protects the DCE restart indication state r2
NCr2	T	Restart	DCE maximum number of trials to send a restart indication
TTr2	T	Restart	DTE timeout for the acceptance of the restart from the 4th level (1)
TCr1	T	Restart	DCE timeout for the acknowledgement of the restart from the remote DCEs (2)
NCr1	T	Restart	DCE maximum number of trials to send restart to remote DCEs (2)
W_{tc}	N	Transf.	Size of the window for data transmission from DTE to DCE
W_{ct}	N	Transf.	Size of the window for data transmission from DCE to DTE
L_p	N	Transf.	Local maximum data field length in a packet: network standard
H_s	C	Setup	Overhead in the call set-up phase (bits per call)
H_c	C	Clear	Overhead in the call clearing phase (bits per call)
H_d	C	Transf.	Overhead in the data-transfer phase (bits per packet)
H_r	C	Reset	Overhead in the reset phase (bits per reset)
H_D	C	Restart	Overhead in the restart phase (bits per restart)
E_p	B	Any	Message error rate (3)
N_v	B	Any	Maximum number of concurrent virtual connections on the DTE
C_p	B	Any	Maximum message (net interrupt) throughput on a virtual connection (3), (4)
* A_p	B	Setup	Availability of the virtual connection service
* D_{pm}	B	Transf.	Average delay for the delivery of a message (not interrupt) (3), (4)
* D_{pi}	B	Transf.	Average delay for the delivery of an interrupt (4)
* D_{pe}	B	Setup	Average delay for the establishment of a virtual connection (4)

(1) This parameters is optional, as its existence depends upon the nature of the interface between the 3rd and 4th level of the DTE
(2) This parameter is optional, as its existence depends upon the internal mechanisms implemented on the subnetwork
(3) This parameter is referred to an average message length $\bar{L}_M$
(4) This parameter is defined under the hypothesis that only one virtual connection is being operated on both DTEs

Table 3. INWG 96.1: Transport service classes and elements

	CLASS / ELEMENTS	G LETTERGRAM	L_R REGULAR LIAISON	L_S SUPER LIAISON
PA	Port addressing	yes	yes	yes
IT	Liaison init/term	no	yes	yes
TL	Transport of LETTERS	yes	yes	yes
TT	Transport of TELEGRAMS	no	yes	yes
GD	Guarantee on transit delays	yes	yes	yes
GE	Guarantee on errors	minimum	regular	superior
GS	Guarantee on sequence	no	yes	yes
EC	Delivery confirmation on request	yes	yes	yes
FC	Credits for transmission option	no	yes	yes

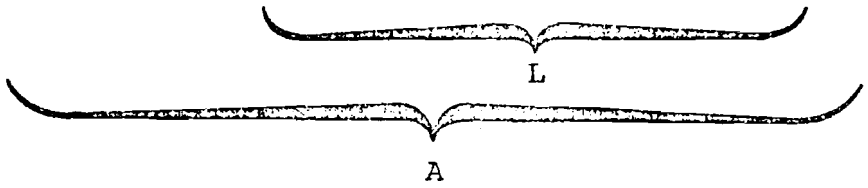


Table 4. Parameters of transport layer from INWG 96.1 proposal.

	Parameter	Class	t.s. class	t.s. element	Meaning
	N_{TA}	B	A	PA	Number of port addresses
	H_{TA}	C	A	PA	Overhead (bits/message) due to port addressing
*	A_{TL}	B	L	IT	Availability of the liaison service
	L_{TL}	B	A	TL	Maximum length of user-information carried by a letter
	L_{TT}	B	L	TT	Length of user-information carried by a telegram
	H_{TC}	C	A	IT,TL,TT,EC,FC	Overhead (bits/message) due to option codes and facilities
	H_{TR}	C	A	IT,TL,TT,EC,FC	Overhead (bits/message) due to message identification
	$E1$	B	A	GE	Signalled bit error rate of the transport service
*	E_{T1}	B	A	GE	Signalled message error rate of the transport service
	$E2$	B	A	GE	Non signalled bit error rate of the transport service
	E_{T2}	B	A	GE	Non signalled message error rate of the transport service
	E_{TS1}	B	L	GS	Letter sequencing error rate
	E_{TS2}	B	L	GS	Non detected letter sequencing error rate
	C_{TP}	B	A	TL,TT	Maximum throughput, as percentage of the maximum throughput of the data transmission service
	D_{TA}	B	A	TL,TT	Maximum additional delay introduced by the transport stations
	D_{TM}	B	A	TL,TT	Maximum transit delay (from port to port)
*	R_T	B	A	TL,TT,GE,GS	Reliability of the transport service
	C_T	B	A	TL,TT	Maximum throughput achievable within an established ports association
	D_{TL}	B	A	TL	Average delay for letter delivery on an established ports association
	D_{TT}	B	L	TT	Average delay for telegram delivery on an established liaison
*	D_{TEL}	B	L	IT	Average delay for the establishment of a liaison
	D_{TLR}	B	A	TL	Average round-trip delay for delivery and confirmation of a letter
	D_{TTR}	B	L	TT	Average round-trip delay for delivery and confirmation of a telegram
	L_{TF}	N	A	TL	Basic standard maximum fragment length
	N_{TMF}	N	A	TL	Maximum number of fragments in a transport frame
*	T_{RS}	T	A	TL	Timeout for reassembly of a letter being received
	H_{TGE}	C	G	TL,EC	Overhead for error control in lettergram mode (length of LG-ACK)
*	T_{GE}	T	G	TL,EC	Timeout for reception of lettergram acknowledgement
*	T_{LLO}	T	L	IT	Timeout that protects the "opening by local user" state of liaison
*	T_{LRO}	T	L	IT	Timeout that projects the "opening by a remote user" state of a liaison
	T_{LC}	T	L	IT	Timeout that protects the "closing" state of a liaison
	H_{TT}	C	L	TT	Overhead for error control on telegrams (length of LI-TAK)
	T_{TT}	T	L	TT	Timeout for telegram acknowledgement
	N_{TT}	T	L	TT	Maximum number of transmissions of a telegram
*	T_{LE}	T	L	TL,EC	Timeout for letter acknowledgement on a liaison
	N_{LE}	T	L	TL,EC	Maximum number of transmissions of a letter on a liaison
	W_C	N	L	TL,FC	Maximum delay of the receiving TS to acknowledge a letter on a liaison
	L_{TLP}	B	L	TL,IT,FC	Flow-control maximum size of letters on a liaison (agreed on opening)
	D_{LA}	N	L	TL,EC	Flow-control dynamical window (number of credits for transmissions)

REFERENCES

[1] ISO (1978)(The International Organization for Standardization) Draft Proposed International Standard 2382/IX - Data Processing - Vocabulary - Section 09: Data Communication. ISO Report TC 97/SC6, May, and INWG (International Networking Group) General Note No. 178, November.

[2] ISO (1978)(The International Organization for Standardization) Comments on Document ISO/TC97/SC 16 N 117 (Version 3) - Open Systems Interconnection. IFIP (International Federation for Information Processing) Report No. TC6, November, and INWG (International Networking Group) General Note No. 202, March, 1979.

[3] CCITT (1977) (The International Telegraph and Telephone Consultative Committee) Provisional Recommendations X.3, X.25, X.28, X.29 on Packet-Switched Data Transmission Services: 8-49. ISBN 92-62-00591-8. Geneva: CCITT.

[4] Pouzin, L. (1976) Virtual Circuits Versus Datagrams: Technical and Political Issues. INWG (International Federation for Information Processing) General Note No. 106, 5 February.

[5] Butrimenko, A., and U. Sichra (1979) Virtual Circuits Versus Datagram - Usage of Communication Resources. Pages 525 - 537, Proceedings of the 4th International Symposium on Modeling and Performance Evaluation of Computer Systems, edited by M. Arato, A. Butrimenko and E. Gelenbe, Vienna, 6 - 8 February. Amsterdam: North-Holland.

[6] Cerf, V., A. McKenzie, R. Scantlebury and H. Zimmermann (1978) Proposal for an Internetwork End-to-End Transport Protocol. Pages H-5 - H-25, Proceedings of the Symposium on Computer Network Protocols, edited by A. Danthine, Liege, Belgium, 13 - 15 February. Liege: Universite de Liege.

[7] Garcia, C., R. Gardier, A. Marchand, M. Martin and H. Zimmerman (1975) Specifications de Realisation de la Station de Transport ST2 Portable, February. Rocquencourt, France: Reseau CYCLADES.

[8] Deparis, M., A. Duenko, M. Gien, J. Laws, G. le Moli and K. Weaving (1976) The Implementation of an End-to-End Protocol by EIN Centres: A Survey and Comparison. Pages 351 - 360, Proceedings of the Third International Conference on Computer Communication, edited by P. K. Verma, Toronto, 3 - 6 August. Toronto: ICCC.

[9] Le Moli, G. (1973) Colloquies in Computer Networks. Paper presented at the International Advanced Study Institute of Computer Communication Networks, University of Sussex, Brighton, 9 - 15 September. Nordhoff-Leyden, 1975.

[10] Danthine, A., (editor) and J. Bremer (1978) Modelling and Verification of End-to-End Transport Protocols. Pages F5-1 - F5-12, Proceedings of the Symposium on Computer Network Protocols, Liege, Belgium, 13 - 15 February. Liege: Universite de Liege.

[11] Butrimenko, A., and G. Scollo (1980) Protocol Parameters and their Interrelations in X.25; INWG 96.1 Sample Network Architecture. WP-80-17. Laxenburg, Austria: International Institute for Applied Systems Analysis.

MAILING SYSTEMS IN COMPUTER NETWORKS

F. Caneschi

INTRODUCTION

A MAIL system is one of the most important services that a computer network is required to provide. Furthermore, the ability to send documents around a community of users is regarded today as a basic tool for scientific work.

A "normal" MAIL system provides its users with a transport service, by means of which documents are carried on to their destination. It is necessary for the user to supply only an address, i.e., a set of keywords which identify (more or less approximately) the destination of the document and, of course, the document itself.

The correspondence between the end user (the destination) and the address which was specified by the sender is not univocal and, sometimes, can generate confusion. If, for example, one sends a document with the following address:

Clark Kent
Smallville (NY)
USA

there is a strong probability that the letter will arrive at the correct destination (provided that there is only one Smallville in the New York State and that Smallville is small enough).

On the other hand, an address like:

John Smith
New York (NY)
USA

would probably be insufficient, despite the fact that this address has an identical format to the former one.

The mailing system process is normally as follows: the letter is sent to the nation specified on the address, then to the city, the street (if specified), and finally a person with the specified name is sought in the neighbourhood of the geographical location identified by the previous step. This system can work (and actually does work in most cases) when the final part of the search is performed by human beings. Problems arise mainly when both the transport service and the searching work are performed by an automata, i.e., an entity which is able only to make a limited number of choices. This is the case which must be studied: a mail system built upon a computer network.

MAIL SYSTEMS IN A COMPUTER NETWORK

In the early period of the first computer networks, special attention was paid to the File Transfer Service; thus it is possible to say that every operational (and also planned) computer network provides its users with a File Transfer Service. For this reason, the choice of File Transfer as a transport service for the mail system is quite natural. As we have seen, the transport service is, however, only one aspect of a mail system; the delivery strategy is the other side of the coin. The author does not intend to deal now with all the possible strategies: the main purpose of the following is to develop a particular delivery strategy, based on the type of mailing system service required by the user.

MAILING SERVICES

A user normally expects much more from a computerized mailing service than from a "normal" one. The main differences are as follows:

a) the user mail should always arrive at its destination, and in a short time,
b) the destination user should be, not only acknowledged if mail arrives when he is logged-in, but should also be prompted by the system when he logs-in if mail is ready for him,
c) users should be addressed, not only by names, but also by interest groups. For instance, one should be able to send mail to anybody (known to the mail system) interested in science fiction,
d) the user should be able to change his address without too much effort (bureaucracy), and without the necessity of notifying all his friends of his new address,
e) the user should be able to specify a device (normally a printer) to which his mail should be directed after having remained for a suitable period of time in the on-line memory (disks, for instance), if nobody has received it in the meantime.

All the other attributes of a "normal" mail system are also, of course, requested, the most important being the integrity and security of the mail.

All of these problems are not trivial, be it from an implementational point of view ("Give me a good team of programmers and I will be the 'Emperor of the Universe'", said a Project Leader), or from a more theoretical point of view.

The main factor is the "addressability" of the users who announce themselves as being "ready to accept mail". As these users have to be divided into interest groups and each of these groups is defined in each host of the network, a database-type problem arises: should this database be organized in a distributed or in a centralized manner?

The "cleanest" way is to have in each host a database of the local users. But how can the local mailing system associate a name supplied by the user with a non-local address? And furthermore, if the name supplied by the user also contains a host name as an address, how can the local mail system be sure that the address supplied by the user is indeed correct? Finally, if a user moves to another address, should his address be kept only in the new destination, or also in the old one?

On the other hand, if there is a unique place in the network where the users' names (and their addresses) are stored, should the host which holds these data fail, there will be no way of making the mail system work. A third solution might be to keep the complete main list in each host; this is, however, not feasible for two main reasons:

a) a change in an address would start a sort of "reconfiguration process" among all the local mailing systems, which is time-consuming and difficult to control,
b) a "complete" mailing list in each node is a waste of memory.

It follows directly from the above discussions that a distributed database should be chosen for the mailing list. In the next Section an attempt is made to answer the questions posed above, as well as others connected with the problem, according to whicever strategy is adopted.

MAIL SYSTEM ARCHITECTURE

Mail integrity is the responsibility of the File Transfer Service, which we presume to be sufficiently reliable. Using such an approach, at the request of the sender user, the local mailing system takes the mail file out of its private space (main memory or disk) and forwards it to the remote mailing system specified in the address by means of the File Transfer Service. We assume, of course, that only one mailing system exists in each node.

As we have already pointed out, however, the transport service is only one facet (perhaps the most trivial) of the problem.

Due to the fact that there are certain processes (local mail systems) which colloquiate by means of the network, a control problem protocol arises. In the following, we will try to explain the choices we made, rather than to demonstrate that they are the only feasible ones. Our approach is a "top-down" one: we start by taking into account the characteristics of a mail system as they appear to an external user and then derive our design.

If a user could be addressed by name and address, the problem would be much simpler; in a complete mailing system, however, he has to be addressed also by category. In other words, a sender user should be able to send messages to all the possible addressees in a host, or even better, to everybody who is interested in a particular subject, without considering the actual network addresses of these persons. In order to perform these kinds of services, the mail system requires to have a knowledge of the local users and of the protocol to colloquiate with the other mail systems.

Local Data

All those who wish to be recognized by the mailing system have to notify the system of their name and "computer name". The "computer name" is the name normally used for the center where the host computer resides, e.g., MIT, provided that this name is unique in the network. In addition, the user has to specify one or more "categories", i.e., interest fields, in which he wishes to be included, in order to receive "general" mail. The mailing system keeps this "address book" for its local users. Although this is a purely implementational aspect, this file should ideally be structured in such a way as to permit a by-name, by-category, or by-host search order.

Another table maintained by the mail system is the host table, in which a correspondence is established between "computer name" and network address, and an indicator is maintained which specifies whether a host computer is reachable or not.

Control Protocol

Each local mail system tries to alert all the other mail systems as soon as it starts (or is started by the system operator). This permits the mail, previously unable to reach its destination, to be sent.

When a mail system has to send mail, the procedure is as follows:

1. it sends a message (network message) to the addressed mail system (one mail system for each host), asking whether the addressee is defined there,
2. it sends the mail (letter) by means of the File Transfer Service to the remote mail system,
3. it notifies the sender that his mail has arrived and closes the connection.

If the destination is unattainable, the mail is kept in mass-storage (disk) until the remote system sends a message, notifying its readiness to accept mail. If the addressee is not defined in the remote mail system, the user is alerted and the mail is not sent. Should the addressee have changed his address, the new address is returned to the sender user, and the procedure restarts automatically from point 1. above. The sender user is notified of the new address. After a suitable period of time, both the unsent letters and the old (changed) addresses will be deleted. The same procedure applies when mail has to be sent to one or more category, the only difference being that it is repeated as many times as the many hosts are defined in the network.

MAIL SECURITY

Although it might be possible to consider the mail system described above as "sure", it can be seen from the following that no system can be regarded as "absolutely sure". The main features, as far as security is concerned, are as follows:

a) Mail is sent from one mail system to another, not from user to user,
b) Only the addressee is allowed to receive his mail. (When a name is made known to the mail system, a password can be added),
c) The mass-storage space of the mail system is accessed only by the mail system.

There is no mechanism which allows a user to "sign" his letter in order that the adressee may know the identity of the sender. The main reason for this is that, if a cryptic mechanism is implemented in the mail system, it should be protected, and so on. Two users can, however, agree on a common cryptic system, and then use the mail system to transport coded texts.

CONCLUSIONS

Computerized mail and teleconferencing systems are now being studied all over the world. It was not the author's intention in this report to specify a protocol exactly, but simply to present some ideas on what a computerized mailing system should be and how it should work. This is a typical "top-down" approach, and in my opinion much closer to the top than to the bottom. However, a mail system with the above explained characteristics will be implemented on the RPCNET network. By way of conclusion, in the spirit of a "top-down" approach, the following table shows how a possible "MAIL" command could be formulated.

```
MAIL

SEND      [ ALL
            TO ALL  category 1  category 2...[ node 1 node 2 [DISK]
                                             [NODISK]
            TO      name (node)

RECEIVE   [FROM ALL  category 1  category 2....]  [DISK  ]
          [FROM      name [SINCE  date  TO  date]]
          [ALL       ALL                        ]  [NODISK]

QUERY     [LOCATIONS
          CATEGORIES
          MYADD
          NAME      [ ALL                ALL ]
          RECV      [                        ]
          SEND      [ category 1.... (node 1...]
                    RECV   [category 1.... (node 1...]
          ALL       SEND   [                        ]
                    NAMES  [ ALL                ALL ]

PURGE     [FROM ALL  [category 1.....
          FROM        name 1.....
          TO ALL     [category 1.....
          TO          name 1.....
                     [RECV]
          ALL        [SEND]
                     [ALL]

          MYADD                  [OFF ]
                                 [name]

          NAME      [name   category 1....  ]
                    [       OFF             ]

SET       NEWCAT    category 1....

          PRIVIL-   OFF
          EDGED     name   CLASS  class  [CATEGORY  category 1]
```

Legenda:

a) Square brackets = mean optional parameter(s)

b) Keyword underlined = default value

Table 1. Possible formulation of a mail command.

IIASA'S X.25 GATEWAY

A. Labadi

INTRODUCTION

IIASA has been working in the field of computerized data communications for some five years, carrying out studies and experiments in various areas of data exchange. This work has been made possible by the experience IIASA has gained in establishing and operating connections to a number of computers and networks. Because of its unique position in the international scientific world, IIASA is a diligent promoter of East-West data communications. IIASA and its collaborators are able to provide on request various services to the NMO countries over the leased lines which have been established for this purpose.

As the number and variety of external data communication connections has increased, the need has arisen for a general switching facility, capable of handling any of the external lines coming into IIASA. In addition, great importance has been attached to obtaining reliable control over international data traffic.

The so-called "IIASA gateway function" was developed in order to satisfy these needs. From a technical point of view,

the main problem here is how to provide as much transparent connection between end points as possible in view of the diverse nature of the communication lines. The introduction of the notion of a "virtual communication line" provides a flexible solution to this. Authorization is handled by a logon service which has full control over the establishment of connections.

THE VIRTUAL COMMUNICATION LINE

Units of data processing equipment (computers, terminals, etc.) situated far apart are connected via data communication lines. A major characteristic of these lines is their lack of reliability. In order to achieve reliable information interchange (where reliability is relative and depends upon the particular requirements of a situation), line control protocols are used. A line control protocol is a set of format restrictions, procedures and time-outs. One of the main functions of these line control protocols is to insure reliable data transmission (segmenting, usage of check sum, positive and negative acknowledgement, etc.). Another main function is the exchange of control information between the two partners at the extremities of the line. Control information includes the opening and closing of the line, some flow control, etc. In some cases, line control protocols have special functions, such as peripheral selection, maintenance of independent data streams, etc.

Let us assume that two data communication lines having two different line control protocols are to be connected via computer. If we regard the line control protocols as two languages each having its own vocabulary and grammar, the program which is required to connect them can be considered as an interpreter for the two languages. This interpreter will be responsible for making conversions such as re-segmenting and re-formatting, a task which on the surface does not seem to be very difficult. However, in designing the program, one is sometimes faced with the problem of how to translate a function which can be expressed in only one of the languages. The basic problem arising here is one of what to do when one language cannot be completely translated into another. In specific applications, if all such

circumstances are known, a proper decision can be made: either the interpreter can process and act upon each of those special functions available in only one of the languages; or there is no satisfactory solution at all.

Now let us examine those cases where interpreters can be created. Where there are three lines, any pair of which could be connected, we will need three interpreters. At n lines, the number of interpreters required will be $\binom{n}{2}$, a figure which grows rapidly as n increases. Thus the value of the notion of a "virtual communication line" represented by its "virtual language" in order to decrease the number of interpreters necessary seems readily apparent.

Let us name the languages of the real data communication lines "native languages". With this new technique of a virtual language, only one interpreter will be written for each real communication line (Figure 1). Each interpreter will translate back and forth between its own native language and the virtual language. The virtual language acts as the link between any interpreter-pair. The number of interpreters needed when one uses a virtual language is far smaller than the number required without this language. Where n = 7, for instance, the use of the virtual language results in the need for 7 interpreters; without it, 21 would be needed. The reasoning behind the introduction of a virtual language is somewhat similar to that for the well-known "virtual terminal" concept.

When designing a general purpose switching system, the virtual language must be carefully written. It is of the utmost importance that each of the native languages be capable of being entirely translated into the virtual one. At any given point in time, with the knowledge of all native languages and some guesswork on their likely enhancement in the future, a suitable virtual language can be designed.

But what will happen if a new native language emerges, if a new kind of terminal must be handled, and/or if a new computer or network having a different kind of access method is to be connected? If the new native language is capable of being

entirely translated into the virtual language, then only its own interpreter need be written (Figure 2). But if it is not, other solutions must be sought.

One possibility might be to redefine the virtual language and to rewrite all the interpreters. This solution is definitely the costliest in terms of effort required (Figure 3).

Another solution might be to redefine the virtual language in such a way that the old virtual language can be translated entirely into the new one. In this case, in addition to the interpreter needed between the new native and the new virtual language, only another interpreter between the old and the new virtual language need be written (Figure 4). This would require much less effort than the previous solution.

Finally, where there are new communication lines with their own virtual languages and interpreters, it would be possible to handle the new parts of the system similarly to, but separately from, the old parts. This would require less effort, but would not allow intercommunication between "new" and "old" parts of the system.

THE GATEWAY FUNCTION

The gateway was designed with the following line-types in mind:

- -- IBM BSC (IBM 3780 RBT) line, 2400 baud synchronous connection to terminal or computer [1];
- -- CDC 200 UT line, 2400 baud synchronous connection to computer [2];
- -- TTY-like terminal lines, 110-9600 baud asynchronous connections [3];
- -- Host lines, 110-9600 baud asynchronous connections;
- -- X.25 lines [3], 2400 baud synchronous connections to networks or remote concentrators applying bitstuffing or BYSINC framing

From the communication point of view, the switching system is shown in Figure 5.

A subset of the X.25 level 3 protocol has been chosen as the virtual language. Almost everything has been included from the protocol except the time-outs and the error recovery procedures. Interpreters are expected to work as "ideal" X.25 Level 3 automats.

Individual data communication lines are handled by "INTERFACES". Their duties match those mentioned in connection with the interpreters. Within the system, pairs of INTERFACES are designated as "associations". A given connection between two INTERFACES is marked by an internal association number which is established at call-time and released at clear-time. The internal association numbering is somewhat similar to the channel numbering of the X.25. During any one transaction, the INTERFACES know no more than the association number to which the traffic belongs. An INTERFACE can be part of any number of associations at any one time.

All address-like information concerning requested destinations is handled by the "LOGON" process. The LOGON is responsible for security check, association establishment and for the selection of the destination INTERFACE. After having established a particular association, the data flow is maintained by the "ROUTE", which recognizes partner INTERFACES by their association numbers.

The establishment of a connection between two INTERFACES handling external X.25-type lines is shown in Figure 6. The initial call can be rejected (cleared up) at several places if buffer space or a free channel are lacking, or if the password is invalid.

A special monitoring function is built into the system allowing the operator of the Gateway to monitor any of the connections. This means that his screen can produce a replica of any user's terminal. This and his capacity to exchange messages with the user provides a remote network support for the users.

The following gives additional details on particular INTERFACES providing access to X.25 type lines as well as on INTERFACES connecting TTY-like terminals and asynchronous hosts.

X.25 INTERFACE

The "X.25 INTERFACE" is responsible for the X.25-type data communications lines. As is common in the System, it resembles a reliable X.25 Level 3 Automat; i.e., every kind of error relating to the X.25 line, including procedural errors, will be handled by the "X.25 INTERFACE" without disturbing the gateway. Via internal association numbers, the Interface connects with other INTERFACES. It may have as many active connections as there are virtual calls at any one time.

The "X.25 INTERFACE" is made up of three parts, the "FRAM", the "LAPB" and the "XLEV3".

The FRAM is responsible for the frame level of the X.25. A HDLC or BISYNC type of framing can be used. The format of the BISYNC type is defined in [4] and [5]. The FRAM maintains a trace area for debugging and monitoring purposes. It actually contains two separate parts for characters received and transmitted. A part of the trace file of the FRAM is shown in Figure 7.

The LAPB is responsible for reliable data transmission over X.25-type lines, following the rules of the X.25 level 2 protocol (Link Access Protocol, Balanced) recommended by the CCITT. It maintains a trace area for debugging and monitoring purposes. A part of the trace file is shown in Figures 8/A and 8/B.

On the file there are two bytes for each frame received or transmitted. These are the ADDRESS and CONTROL bytes. One line is printed for each two such bytes. Trace information about frames transmitted is printed from the beginning of the line. Information about frames received is tabulated by four positions. Addresses :20 and :11 (hexadecimal) are used by "LAPB, whereas :10 and :21 are used by the remote partner. Command frames are signed by addresses :10 and :20. The behaviour of "LAPB" in

various erroneous situations, such as lack of buffer space, missing I frames, etc., as well as the strategy of Poll/Final exchange, are shown on the trace.

The "XLEV3" of the "X.25 INTERFACE" is responsible for the packet level protocol of the X.25 as recommended by CCITT [3]. Among its duties is mapping between the virtual call numbering and the internal association numbering. It communicates with "LAPB" via I frames carrying packets and service messages concerning the status of the line and possible procedural errors occurring on the second level of X.25. "XLEV3" maintains a trace area for debugging and monitoring purposes.

Parts of the trace files of the "XLEV3" are shown in Figures 9 and 10. On these files there are four bytes for each packet exchanged between "XLEV3" and either "LAPB" or the System. The first of the four bytes must differentiate among the four possible kinds of packet-exchange; the others contain the first three bytes of the packet exchanged, (format and logical channel group number, channel number and packet identifier.)

A trace line contains information about a given packet exchange. In the first column:

L-X means from "LAPB" to "XLEV3"
S-X means from System to "XLEV3"
X-L means from "XLEV3" to "LAPB"
X-S means from "XLEV3" to System.

The four bits of FORMAT field are written in the second column. In the third column:

LCGN means logical channel group number.

In the fourth column:

CHNB means channel number.

In the fifth column:

PID means packet identifier, and the value of PID is written in a hexadecimal format. In the sixth column the packet type is written in terms of key words used by the CCITT X.25 documentation.

Where lines are marked by X-L or L-X FORMAT, LCGN and CHNB have the original meanings, while at lines X-S and S-X, CHNB

contains the internal association number to which the traffic belongs, LCGN has meaning at call time only, and FORMAT has no meaning at all.

In Figure 9, initial restart, call set-ups and clears are shown. From "LAPB" three calls come on LCGN 1 and one on LCGN 2. From the System one call comes on LCGN 1 and two on LCGN 2. All the calls are accepted and later cleared up.

In Figure 10, initial restart, a call set-up, data transfer, reset phase and clear-up are shown.

At System generation, the following parameters can be defined:

Frame level:
-- HDLC or BISYNC type framing
-- if BISYNC, the codes of control characters (SYN, DLE, SOH, ETB)
-- station address
Level 2:
-- window size
-- maximum number of retransmissions
-- time outs
Level 3:
-- logical channel groups
-- available channels for each group
-- available associations at a time
-- DCE or DTE
-- low or high channel numbering
-- window size
-- time-outs

PAD INTERFACE

The "PAD INTERFACE" is responsible for the start-stop mode terminal lines. It follows the rules of the CCITT X.3 and X.28 recommendations [3]. All the PAD parameters and the two standard profiles are implemented. Since the CCITT recommendations are incomplete in some cases, i.e., in the SELECT and PROFILE commands, certain local conventions have been introduced.

The "PAD INTERFACE" can handle both local and remote terminals. In the case of locally attached terminals, the speed can vary between 110 and 9600 baud, while the remote terminals can run at up to 300 baud. "PAD" maintains trace area for each of the terminals. A part of a trace file is shown in Figure 11. The format of the trace is similar to that mentioned at X.25 Level 3, but FORMAT and LCGN have no meaning, and CHNB shows the internal association number assigned to that particular data flow.

At System generation, the following parameters can be defined:

-- default profile
-- window size
-- time-out

HPAD INTERFACE

The "HPAD INTERFACE" is responsible for connections to host computers accessed via asynchronous channels. This is done by simulating a proper start-stop mode terminal known by the host. "HPAD" performs almost the same packet assembly-disassembly function that PAD does. Its mode of operation is controlled by an internal profile which can be adapted to various host connections. In addition to the parameters of the packet assembly-disassembly function, this profile can provide character conversions as well as flow control on the host line. Flow control by means of the CTRL S and CTRL Q characters is implemented on almost all time-sharing host computers.

The speed of remote host lines varies from 110 to 1200 baud. Locally attached host lines can run at up to 9600 baud.

"HPAD" maintains a trace area from each of the host lines. A part of a trace file is shown in Figure 12. The format of the trace is the same as that described for "PAD".

At System generation the following parameters can be defined:

-- profile
-- window size
-- types of flow control--if any
-- time out.

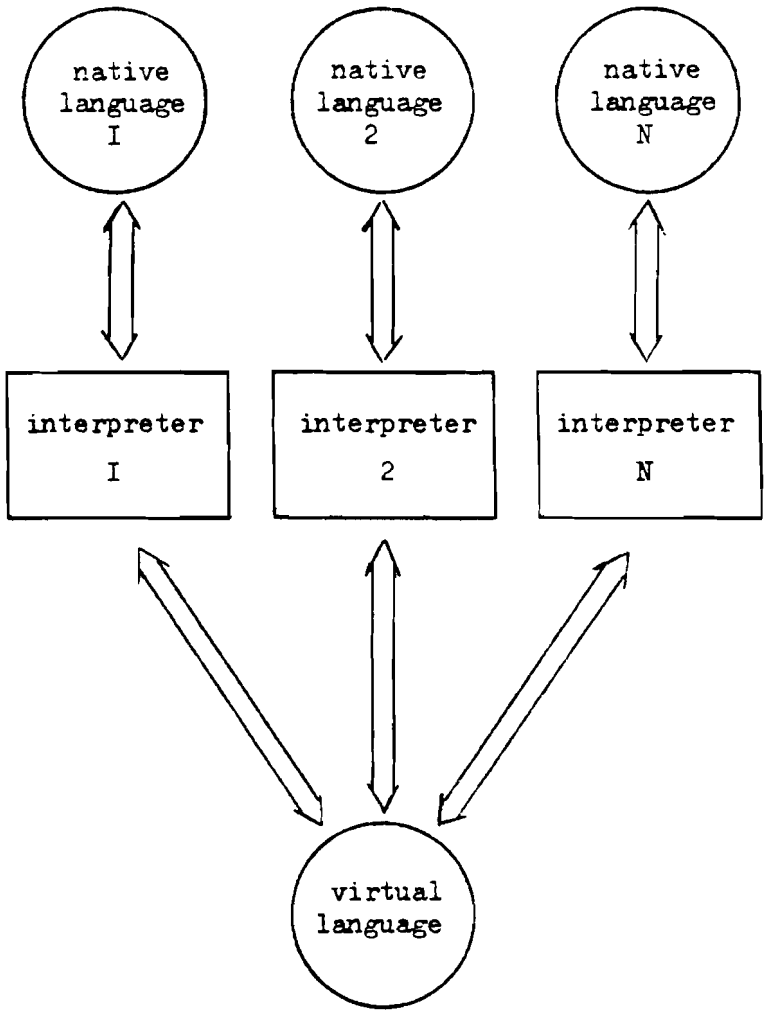


Figure 1. Interpreters and the virtual language.

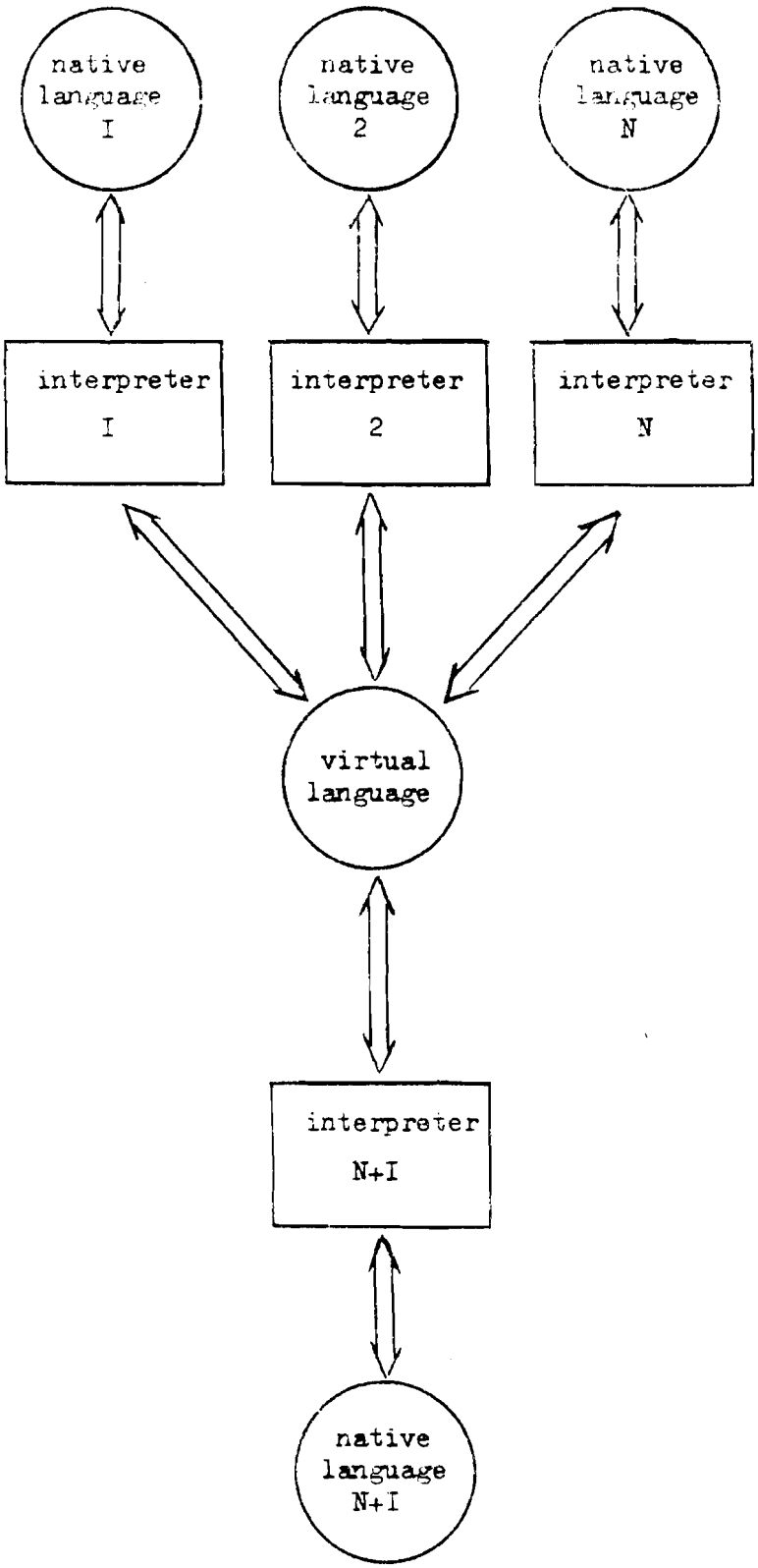


Figure 2. Integration of a new native language when it fits the virtual language available.

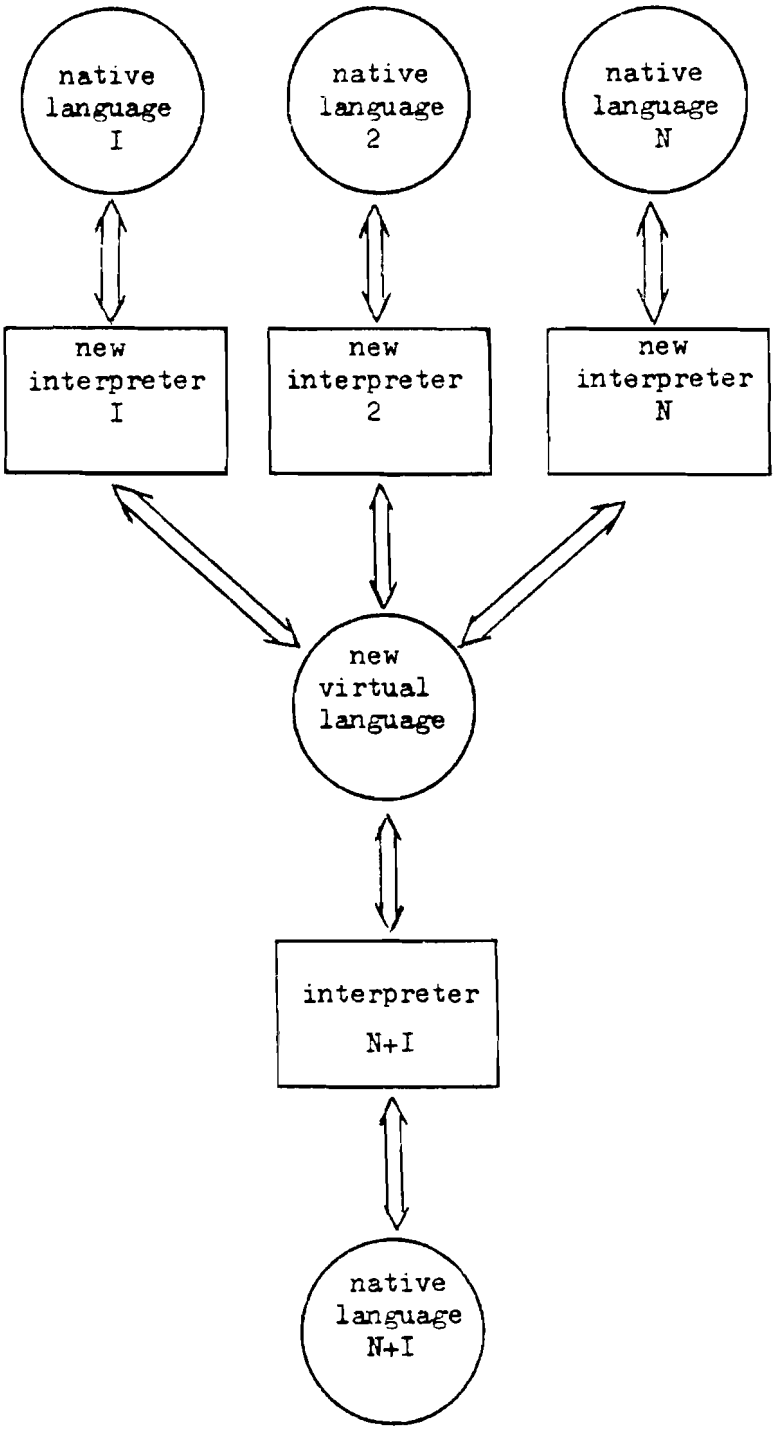


Figure 3. Redesigned system with the new virtual language.

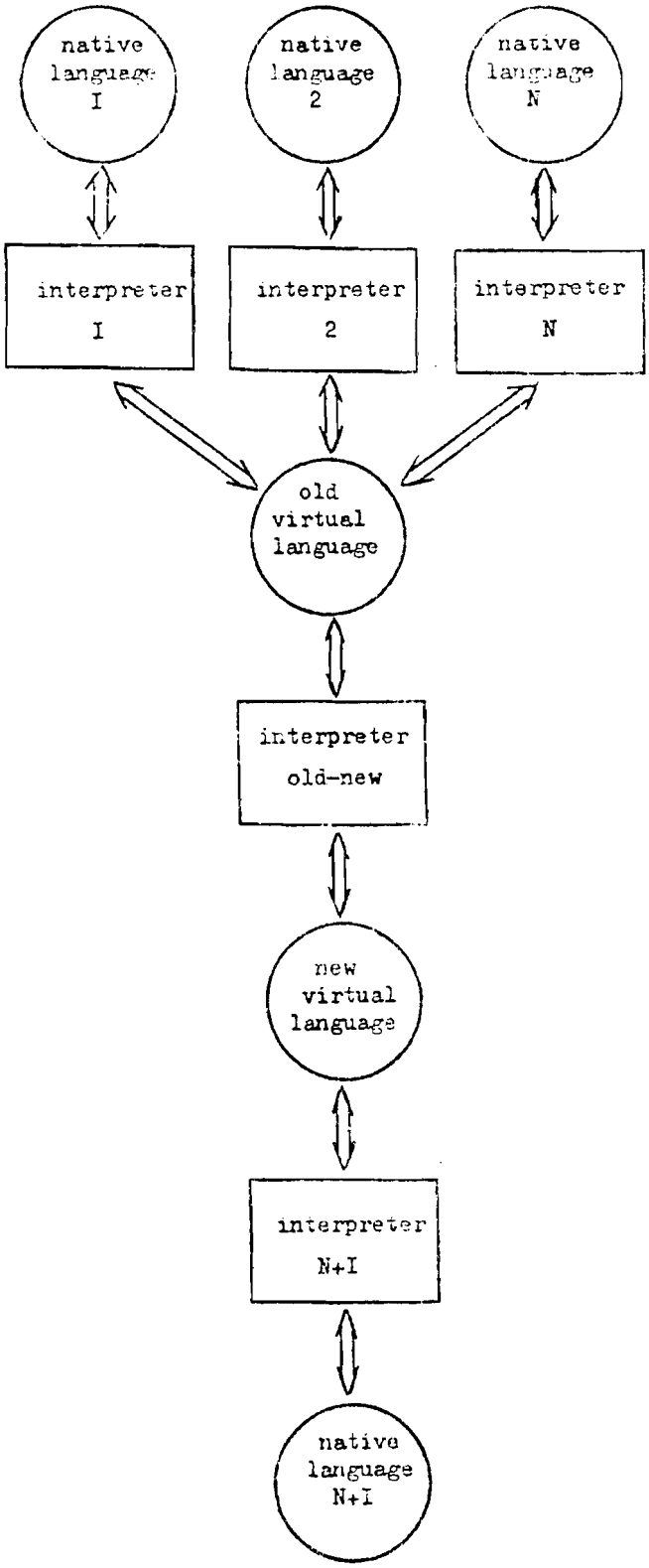


Figure 4. Integration of new native language by means of a new virtual language and an interpreter between the old and new virtual languages.

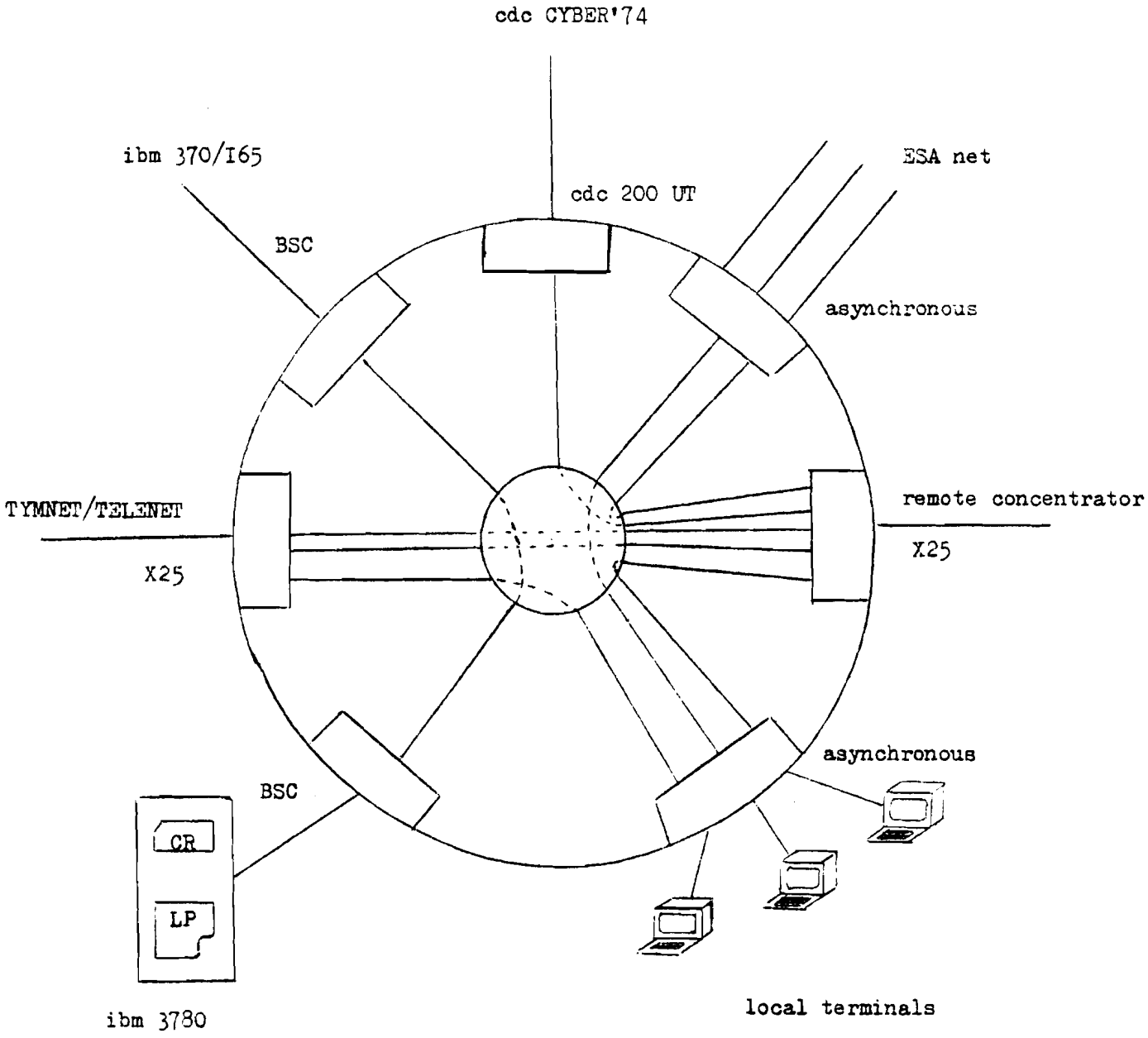


Figure 5. Communication through the Gateway.

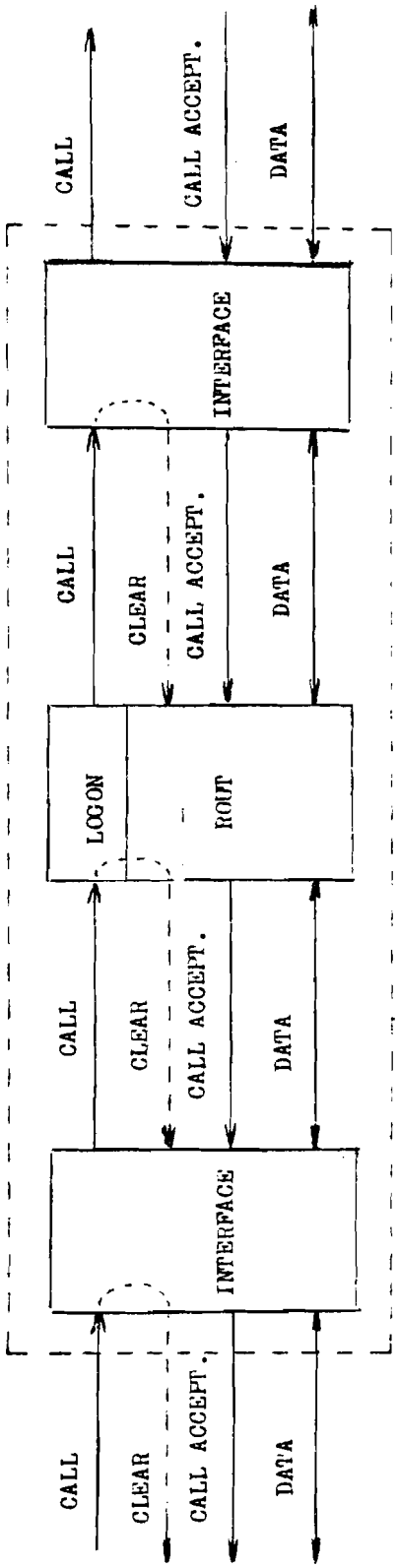


Figure 6. Call establishment between two "Interfaces".

```
16 SYN
16 SYN
16 SYN
16 SYN
10 DLE
01 SOH
10 DLE
10 ADDRESS
2F SABM P/F=0
10 DLE
17 ETB
EB CRC1 EB
5D CRC2 5D

16 SYN
16 SYN
16 SYN
16 SYN
10 DLE
01 SOH
21 ADDRESS
63 UA P/F=0
10 DLE
17 ETB
5F CRC1 5F
6E CRC2 6E

16 SYN
16 SYN
16 SYN
16 SYN
10 DLE
01 SOH
10 DLE
10 ADDRESS
11 RR P/F=1 N(R)=0
10 DLE
17 ETB
59 CRC1 59
71 CRC2 71

16 SYN
16 SYN
16 SYN
16 SYN
10 DLE
01 SOH
21 ADDRESS
11 RR P/F=1 N(R)=0
10 DLE
17 ETB
2B CRC1 2B
AD CRC2 AD
```

Figure 7. Frame format.

```
A[illegible]: 20 SABM P/F=0
     A[illegible]: 10 SABM P/F=0
A[illegible]: 11 UA P/F=0
     A[illegible]: 21 UA P/F=0
A[illegible]: 20 DISC P/F=1 N(R)=0
     A[illegible]: 10 DM P/F=1 N(R)=0
A[illegible]: 11 DM P/F=1 N(R)=0
     A[illegible]: 21 DM P/F=1 N(R)=0
A[illegible]: 20 I  N(S)=0 P/F=1 N(R)=0
A[illegible]: 20 I  N(S)=1 P/F=0 N(R)=0
A[illegible]: 20 I  N(S)=2 P/F=0 N(R)=0
A[illegible]: 20 I  N(S)=3 P/F=0 N(R)=0
     A[illegible]: 21 RR P/F=1 N(R)=1
A[illegible]: 20 I  N(S)=4 P/F=1 N(R)=0
     A[illegible]: 10 I  N(S)=0 P/F=1 N(R)=2
     A[illegible]: 10 I  N(S)=1 P/F=0 N(R)=2
     A[illegible]: 21 RR P/F=0 N(R)=3
A[illegible]: 11 I  N(S)=5 P/F=1 N(R)=2
     A[illegible]: 10 I  N(S)=2 P/F=0 N(R)=3
     A[illegible]: 21 RR P/F=0 N(R)=4
     A[illegible]: 10 I  N(S)=3 P/F=0 N(R)=4
     A[illegible]: 21 RR P/F=1 N(R)=5
A[illegible]: 20 I  N(S)=6 P/F=1 N(R)=4
     A[illegible]: 10 I  N(S)=4 P/F=0 N(R)=5
     A[illegible]: 10 RR P/F=1 N(R)=6
A[illegible]: 11 I  N(S)=7 P/F=1 N(R)=5
A[illegible]: 20 I  N(S)=0 P/F=0 N(R)=5
A[illegible]: 20 I  N(S)=1 P/F=0 N(R)=5
     A[illegible]: 10 I  N(S)=5 P/F=0 N(R)=5
A[illegible]: 20 I  N(S)=2 P/F=0 N(R)=6
     A[illegible]: 21 RR P/F=1 N(R)=7
     A[illegible]: 10 I  N(S)=6 P/F=1 N(R)=0
A[illegible]: 11 RR P/F=1 N(R)=7
     A[illegible]: 10 I  N(S)=7 P/F=0 N(R)=2
A[illegible]: 20 I  N(S)=3 P/F=1 N(R)=0
     A[illegible]: 10 I  N(S)=0 P/F=0 N(R)=2
     A[illegible]: 10 I  N(S)=1 P/F=0 N(R)=2
A[illegible]: 20 I  N(S)=4 P/F=0 N(R)=2
A[illegible]: 20 I  N(S)=5 P/F=0 N(R)=2
     A[illegible]: 21 REJ P/F=1 N(R)=2
     A[illegible]: 10 RR P/F=1 N(R)=2
A[illegible]: 11 I  N(S)=2 P/F=1 N(R)=2
A[illegible]: 20 I  N(S)=3 P/F=1 N(R)=2
     A[illegible]: 21 RR P/F=0 N(R)=2
A[illegible]: 20 I  N(S)=4 P/F=0 N(R)=2
     A[illegible]: 10 RR P/F=1 N(R)=3
     A[illegible]: 10 I  N(S)=2 P/F=0 N(R)=3
A[illegible]: 11 I  N(S)=5 P/F=1 N(R)=3
     A[illegible]: 21 RR P/F=1 N(R)=4
     A[illegible]: 10 I  N(S)=3 P/F=0 N(R)=4
     A[illegible]: 21 RR P/F=0 N(R)=5
A[illegible]: 20 I  N(S)=5 P/F=1 N(R)=4
A[illegible]: 20 I  N(S)=7 P/F=0 N(R)=4
A[illegible]: 20 I  N(S)=6 P/F=0 N(R)=4
A[illegible]: 20 I  N(S)=1 P/F=0 N(R)=4
```

Figure 8A. Trace file of "LAPB"

```
ADDR: 10 I  N(S)=4 P/F=0 N(R)=5
ADDR: 10 I  N(S)=5 P/F=1 N(R)=6
ADDR: 11 RR P/F=1 N(R)=6
ADDR: 10 I  N(S)=5 P/F=0 N(R)=6
ADDR: 11 RR P/F=0 N(R)=6
ADDR: 21 RR P/F=1 N(R)=6
ADDR: 10 I  N(S)=6 P/F=0 N(R)=1
ADDR: 20 I  N(S)=2 P/F=1 N(R)=7
ADDR: 10 I  N(S)=7 P/F=0 N(R)=1
ADDR: 10 I  N(S)=0 P/F=0 N(R)=1
ADDR: 21 RR P/F=0 N(R)=2
ADDR: 20 I  N(S)=3 P/F=0 N(R)=1
ADDR: 20 I  N(S)=4 P/F=0 N(R)=1
ADDR: 10 I  N(S)=1 P/F=0 N(R)=2
ADDR: 21 RR P/F=1 N(R)=3
ADDR: 10 I  N(S)=2 P/F=1 N(R)=3
ADDR: 21 RR P/F=0 N(R)=4
ADDR: 11 I  N(S)=5 P/F=1 N(R)=3
ADDR: 20 I  N(S)=6 P/F=1 N(R)=3
ADDR: 20 I  N(S)=7 P/F=0 N(R)=3
ADDR: 20 I  N(S)=0 P/F=0 N(R)=3
ADDR: 10 I  N(S)=3 P/F=0 N(R)=4
ADDR: 21 RR P/F=0 N(R)=5
ADDR: 11 RR P/F=0 N(R)=4
ADDR: 20 I  N(S)=1 P/F=0 N(R)=4
ADDR: 10 I  N(S)=4 P/F=0 N(R)=5
ADDR: 21 RR P/F=1 N(R)=7
ADDR: 10 I  N(S)=5 P/F=1 N(R)=7
ADDR: 11 RR P/F=1 N(R)=6
ADDR: 10 I  N(S)=6 P/F=0 N(R)=7
ADDR: 20 I  N(S)=2 P/F=1 N(R)=7
ADDR: 21 REJ P/F=0 N(R)=7
ADDR: 20 I  N(S)=7 P/F=0 N(R)=7
ADDR: 20 I  N(S)=0 P/F=0 N(R)=7
ADDR: 20 I  N(S)=1 P/F=0 N(R)=7
ADDR: 21 RR P/F=1 N(R)=7
ADDR: 10 RR P/F=1 N(R)=0
ADDR: 11 I  N(S)=2 P/F=1 N(R)=7
ADDR: 10 I  N(S)=7 P/F=0 N(R)=0
ADDR: 21 RR P/F=0 N(R)=1
ADDR: 10 I  N(S)=0 P/F=0 N(R)=1
ADDR: 20 I  N(S)=3 P/F=1 N(R)=1
ADDR: 21 RR P/F=0 N(R)=2
ADDR: 10 I  N(S)=1 P/F=0 N(R)=2
ADDR: 10 RR P/F=1 N(R)=3
ADDR: 11 I  N(S)=4 P/F=1 N(R)=2
ADDR: 20 I  N(S)=5 P/F=0 N(R)=2
ADDR: 20 I  N(S)=6 P/F=0 N(R)=2
ADDR: 10 I  N(S)=2 P/F=0 N(R)=3
ADDR: 11 RR P/F=0 N(R)=3
ADDR: 21 RR P/F=1 N(R)=4
ADDR: 10 RR P/F=1 N(R)=5
ADDR: 11 RR P/F=1 N(R)=3
ADDR: 21 RNR P/F=0 N(R)=6
ADDR: 21 RNR P/F=0 N(R)=6
ADDR: 20 RR P/F=1 N(R)=3
ADDR: 21 RR P/F=1 N(R)=6
ADDR: 20 I  N(S)=5 P/F=1 N(R)=3
```

Figure 8B. Trace file of "LAPB".

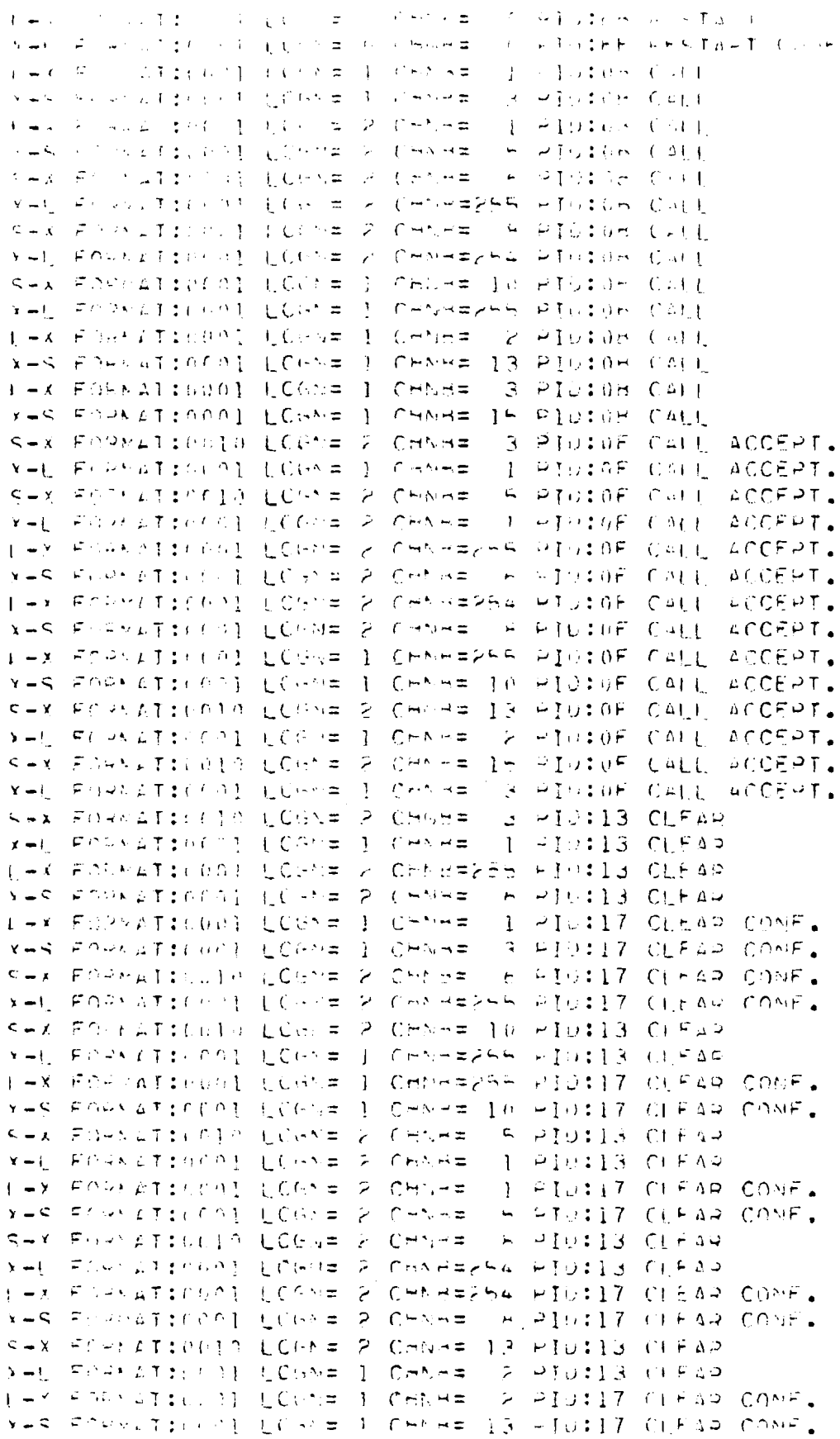


Figure 9. Trace file of "XLEV3"

```
L-X FORMAT:0000] LCGN= 0 CHNM= 0 PID:FF RESTART
X-L FORMAT:0000] LCGN= 0 CHNM= 0 PID:FF RESTART CONF
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:0F CALL
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:0F CALL
S-X FORMAT:0000] LCGN= 1 CHNM= 3 PID:0F CALL ACCEPT.
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:0F CALL ACCEPT.
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:00 DATA          P(R)=0 P(S)=0 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:00 DATA          P(R)=0 P(S)=0 M=0
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:02 DATA          P(R)=0 P(S)=1 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:02 DATA          P(R)=0 P(S)=1 M=0
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:04 DATA          P(R)=0 P(S)=2 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:04 DATA          P(R)=0 P(S)=2 M=0
S-X FORMAT:0010] LCGN= 2 CHNM= 3 PID:01 R. READY      P(R)=3
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:01 R. READY      P(R)=3
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:06 DATA          P(R)=0 P(S)=3 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:06 DATA          P(R)=0 P(S)=3 M=0
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:08 DATA          P(R)=0 P(S)=4 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:08 DATA          P(R)=0 P(S)=4 M=0
S-X FORMAT:0010] LCGN= 2 CHNM= 3 PID:0] R. READY      P(R)=5
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:0] R. READY      P(R)=5
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:0A DATA          P(R)=0 P(S)=5 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:0A DATA          P(R)=0 P(S)=5 M=0
S-X FORMAT:0010] LCGN= 2 CHNM= 3 PID:1F RESET
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:1F RESET
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:1F RESET CONF.
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:1F RESET CONF.
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:00 DATA          P(R)=0 P(S)=0 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:00 DATA          P(R)=0 P(S)=0 M=0
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:02 DATA          P(R)=0 P(S)=1 M=0
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:02 DATA          P(R)=0 P(S)=1 M=0
S-X FORMAT:0010] LCGN= 2 CHNM= 3 PID:01 R. READY      P(R)=2
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:01 R. READY      P(R)=2
L-X FORMAT:0000] LCGN= 1 CHNM= 1 PID:13 CLEAR
X-S FORMAT:0000] LCGN= 1 CHNM= 3 PID:13 CLEAR
S-X FORMAT:0010] LCGN= 2 CHNM= 3 PID:17 CLEAR CONF.
X-L FORMAT:0000] LCGN= 1 CHNM= 1 PID:17 CLEAR CONF.
```

Figure 10. Trace file of "XLEV3".

```
X-S FORMAT:0010 LCGN= 0 CHNB=  1 PID:0B CALL
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:0F CALL ACCEPT.
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:00 DATA          P(R)=0 P(S)=0 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:20 DATA          P(R)=1 P(S)=0 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:22 DATA          P(R)=1 P(S)=1 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:24 DATA          P(R)=1 P(S)=2 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:61 R. READY      P(R)=3
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:26 DATA          P(R)=1 P(S)=3 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:81 R. READY      P(R)=4
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:28 DATA          P(R)=1 P(S)=4 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:A1 R. READY      P(R)=5
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:A2 DATA          P(R)=5 P(S)=1 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:41 R. READY      P(R)=2
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:4A DATA          P(R)=2 P(S)=5 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:C1 R. READY      P(R)=6
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:4C DATA          P(R)=2 P(S)=6 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:E1 R. READY      P(R)=7
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:4E DATA          P(R)=2 P(S)=7 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:01 R. READY      P(R)=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:40 DATA          P(R)=2 P(S)=0 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:42 DATA          P(R)=2 P(S)=1 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:1B RESET
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:44 DATA          P(R)=2 P(S)=2 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:46 DATA          P(R)=2 P(S)=3 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:1F RESET CONF.
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:00 DATA          P(R)=0 P(S)=0 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:20 DATA          P(R)=1 P(S)=0 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:22 DATA          P(R)=1 P(S)=1 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:24 DATA          P(R)=1 P(S)=2 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:61 R. READY      P(R)=3
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:26 DATA          P(R)=1 P(S)=3 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:81 R. READY      P(R)=4
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:28 DATA          P(R)=1 P(S)=4 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:A1 R. READY      P(R)=5
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:2A DATA          P(R)=1 P(S)=5 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:C1 R. READY      P(R)=6
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:23 INTERRUPT
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:27 INT. CONF.
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:C2 DATA          P(R)=6 P(S)=1 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:41 R. READY      P(R)=2
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:4C DATA          P(R)=2 P(S)=6 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:E1 R. READY      P(R)=7
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:4E DATA          P(R)=2 P(S)=7 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:01 R. READY      P(R)=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:40 DATA          P(R)=2 P(S)=0 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:42 DATA          P(R)=2 P(S)=1 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:44 DATA          P(R)=2 P(S)=2 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:61 R. READY      P(R)=3
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:64 DATA          P(R)=3 P(S)=2 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:61 R. READY      P(R)=3
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:13 CLEAR
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:66 DATA          P(R)=3 P(S)=3 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:13 CLEAR
```

Figure 11. Trace file of "PAD".

```
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:0B CALL
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:0F CALL ACCEPT.
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:00 DATA          P(R)=0 P(S)=0 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:20 DATA          P(R)=1 P(S)=0 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:22 DATA          P(R)=1 P(S)=1 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:24 DATA          P(R)=1 P(S)=2 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:61 R. READY      P(R)=3
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:26 DATA          P(R)=1 P(S)=3 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:81 R. READY      P(R)=4
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:28 DATA          P(R)=1 P(S)=4 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:A1 R. READY      P(R)=5
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:A2 DATA          P(R)=5 P(S)=1 M=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:4A DATA          P(R)=2 P(S)=5 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:C1 R. READY      P(R)=6
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:4C DATA          P(R)=2 P(S)=6 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:E1 R. READY      P(R)=7
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:4E DATA          P(R)=2 P(S)=7 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:01 R. READY      P(R)=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:40 DATA          P(R)=2 P(S)=0 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:42 DATA          P(R)=2 P(S)=1 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:44 DATA          P(R)=2 P(S)=2 M=0
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:46 DATA          P(R)=2 P(S)=3 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:1B RESET
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:1F RESET CONF.
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:00 DATA          P(R)=0 P(S)=0 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:20 DATA          P(R)=1 P(S)=0 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:22 DATA          P(R)=1 P(S)=1 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:24 DATA          P(R)=1 P(S)=2 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:61 R. READY      P(R)=3
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:26 DATA          P(R)=1 P(S)=3 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:81 R. READY      P(R)=4
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:28 DATA          P(R)=1 P(S)=4 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:A1 R. READY      P(R)=5
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:2A DATA          P(R)=1 P(S)=5 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:C1 R. READY      P(R)=6
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:23 INTERRUPT
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:27 INT. CONF.
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:C2 DATA          P(R)=6 P(S)=1 M=0
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:4C DATA          P(R)=2 P(S)=6 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:E1 R. READY      P(R)=7
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:4E DATA          P(R)=2 P(S)=7 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:01 R. READY      P(R)=0
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:40 DATA          P(R)=2 P(S)=0 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:21 R. READY      P(R)=1
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:42 DATA          P(R)=2 P(S)=1 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:41 R. READY      P(R)=2
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:44 DATA          P(R)=2 P(S)=2 M=0
S-X FORMAT:0010 LCGN= 8 CHNB=  1 PID:61 R. READY      P(R)=3
X-S FORMAT:0000 LCGN= 0 CHNB=  1 PID:64 DATA          P(R)=3 P(S)=2 M=0
S-X FORMAT:0000 LCGN= 0 CHNB=  1 PID:61 R. READY      P(R)=3
X-S FORMAT:0010 LCGN= 8 CHNB=  1 PID:66 DATA          P(R)=3 P(S)=3 M=0
S-X FORMAT:0101 LCGN= 4 CHNB=  1 PID:13 CLEAR
X-S FORMAT:0101 LCGN= 4 CHNB=  1 PID:17 CLEAR CONF.
```

Figure 12. Trace file of "HPAD".

REFERENCES

[1] IBM 3780 Remote Batch Terminal Hardware Reference Manual.

[2] CDC 200 User Terminal Hardware Reference Manual.

[3] CCITT (1977) (The International Telegraph and Telephone Consultative Committee) Provisional Recommendations X.3, X.25, X.28, X.29 on Packet Switched Data Transmission Services. ISBN 92-62-00591-8. Geneva: CCITT.

[4] Sexton, J.H. (1975) IIASA Data Communication Network, Data Link Control Procedure. Internal Paper: CSN 004.3. Laxenburg, Austria: International Institute for Applied Systems Analysis.

[5] SZTAKI (1976) (Computer and Automation Institute of the Hungarian Academy of Sciences) Note to the Participants of the IIASA Computer Science Network from the Hungarian Party on the Line Control Procedure. Internal Paper: CSN 019. Laxenburg, Austria: International Institute for Applied Systems Analysis.

A MICROCOMPUTER BASED CONTROL SYSTEM FOR X.25 NETWORKS

A. Faro, V. Saletti and G. Scollo

INTRODUCTION

In computer network design and operation, great attention has always been devoted to measuring and controlling network performance in order to ensure suitable levels of continuity and reliability of service.

In order to guarantee such a satisfactory use of the network, service managers have always considered it necessary to create suitable network control and information centres, such as the Network Measurement Centre (NMC) in ARPA, the Network Control Centre (NCC) in EIN and the Network Management Centre (NMC) in EURONET.

The network control systems so far developed are mostly used for measuring, testing and controlling the Communication Network (CN). Their architecture consists, generally, of a common Network Centre (NC) and of a set of processes implemented within the nodes of the communication network, which exchange information with the NC. These processes may send data to the NC under normal and abnormal conditions of the CN.

In particular the data collected by these processes gives useful information on the internal behaviour of the nodes (i.e., queue lengths, delays, packet duplications and losses, special occurrences, and so on) and on their input-output behaviour (i.e., traffic rates, line faults, distribution of messages in length and time, distribution of sources and destinations of the messages, and so on).

The NC requests the data collected from the above processes by means of a suitable command language and produces concise information available to the users at local and remote terminals. In addition the NC allows manual and/or automatic control operations on the CN (e.g., remote reloading of nodes and lines, looping of lines and modems, updating of routing tables and so on.

However in order to utilize the network in a simple, correct and efficient way, other control and information systems can be considered useful or necessary by the users. The aim of this paper is to present an architectural model of management and control systems in computer networks, and to propose also a structure to test, measure and control the X.25 network connections at their endpoints. Lastly, a first version of such a control system is presented emphasizing the advantages of implementing this system in multimicrocomputer Data Terminal Equipment (DTE).

ARCHITECTURE MODEL OF THE CONTROL SYSTEMS

In order to define an architectural model of the control systems in computer networks, we consider the computer network model proposed by the Theory of Colloquies (TC) [1].

In this theory, a computer network can be represented by modules, called interlocutors, variously interconnected through communication channels. Figure 1 shows the interaction of two DTE's through a public network schematized using the concepts of the TC.

From this figure we can also see that the interaction between users and applications arises by means of a set of protocols structured in layers. Each protocol is implemented by a couple of interlocutors running, in general, on remote computers.

This hierarchical structure allows us to propose a functional multilayer approach to the computer network control systems. This approach consists of the following steps:

a) definition of a local control module for each interlocutor of the network, such as special procedures inside the interlocutor (Figure 2a) or a special interlocutor at the upper level of the interlocutor to be controlled (Figure 2b). Due to the substantial equivalence of the two approaches, the second one will be preferred in the following without loss of generality,
b) remote coordination of the local control modules of interlocutors at the same level by means of one or more network control modules (Figure 2c),
c) local coordination of the control modules at different levels running in the same computer by means of the definition of a control structure constituted by the set of the above local or network control modules, possibly coordinated by another control module.

It should now be noted that a Network Control Module (NCM) is characterized generally by the level l at which it works, by the function f which it performs, and by the closed group g(l) of interlocutors which it controls at the level l, producing the following:

$$NCM_{l,f,g(l)} \cdot$$

For this reason we can have several NCM's at the same level performing different functions and/or controlling different user groups. Figure 3 shows a level/centre table in which a black square represents an NCM, and a white square represents an LCM. In this figure two control systems are represented, relating respectively to the function f_1 (control) and f_2 (information). Both the control systems are constituted by NCM's at each level and by LCM's for all the interlocutors on each level. The pattern of communication between the modules of the control system is also shown as follows:

-- the CM's (LCM's and NCM's) of each control system which belong to the same horizontal line (i.e., to the same level) interact by means of messages, but if they are in the same vertical line (i.e., in the same computer), they interact by means of commands,
-- the NCM's belonging to the same level and to different control systems interact by means of messages.

For a given $\bar{f}$ and $\bar{g}(l)$ with $1 \quad L$, a control system is said to be complete if all the NCM's exist for $1 \quad L$ and if, for a given $\bar{l}$, all the interlocutors belonging to $\bar{g}(\bar{l})$ are provided with LCM's. Only complete control systems will be considered in the following text.

Complete control systems can either be centralized or distributed. A complete control system is said to be centralized if all its NCM's are implemented in the same computer. In this case the control structure constitute by the set of all the NCM's, and possibly coordinated by another control module (local coordination), is called the Network Control Centre (NCC)(Figure 4). The control structure constituted by the set of all the LCM's running in the same computer, and possibly coordinated by another control module, is on the other hand called the Local Control Centre (LCC)(Figure 5).

In general the $NCM_{l,\bar{f},\bar{g}(l)}$'s devoted to the function $\bar{f}$ and to the user group $\bar{g}(l)$ are not in the same computer. In this case the control system is said to be distributed, and the $NCC_{\bar{f},\bar{g}}$ is the centre which has the $NCM_{l,\bar{f},\bar{g}}$ at the highest level. On the other hand, for a given $\bar{f}$ and $\bar{g}$, the centres whose CM's at different levels are both LCM's and NCM's, are called Partial Network Control Centres (PNCC's) if they do not have the highest level NCM. In distributed control systems the highest level CM in the PNCC's must be an LCM. Thus the NCC coordinates all the $NCM_{l,\bar{f},\bar{g}}$'s of the network.

In summary, in order to implement the proposed control architecture, it is necessary to define:

-- the function to be performed by the LCM's and NCM's,
-- the levels and the interlocutor group to be controlled for each level,
-- the protocol between the LCM's at the same level and their NCM,
-- the possible protocol between the NCM's at the same level,
-- the procedures of remote and local coordination in order to optimize the network parameters and to manage the collection, processing and inquiry of data bases.

Figure 6 shows a scheme of this architecture in which we point out the network control modules at each level: one controls the interlocutors of the CN; the others control the interlocutor outside the CN.

Based on the control system model described and on our experience of using control modules in EIN, we propose that the LCM's should perform the following tasks:

a) measure the input-output and the internal behaviour of the interlocutors,

b) control the interlocutor during the normal operations, modifying, if necessary, the parameters which characterize the interface and the protocol procedures (time-outs, buffer lengths, etc.),
c) control the interlocutor during the abnormal operations, isolating the faults and advising the operator.

On the other hand the NCM's have to perform the following tasks:

a) collect the measurements performed by the LCM's,
b) define the control policy of the network in normal and abnormal conditions on the basis of the above measurements and on the information coming from the lower and upper NCM's.

Moreover the NCM's (or the NCC) process the collected data and give information to the users on network behaviour (traffic, service availability, service tariffs, etc.).

In real networks only a few control subsystems have already been implemented, such as the communication network management in ARPA, EIN and EURONET, but other interesting control subsystems could also be implemented, such as the Network Connection Control System and the End-Points or the Transport Control System (Figure 7).

The first experience of controlling these systems was gained by EIN. The structure of this control system is as follows:

-- NCC implemented by NPL (London), which is an institutional centre responsible for managing certain special processes implemented in the nodes,
-- SCM (Subnetwork Control Module), implemented by CREI (Milan) especially to test, measure and control the EIN CN by network endpoints,
-- NMC (Network Measurement Module) implemented by CREI (Milan) to perform the mapping of the Transport Stations (TS's).

Another module was also defined at the upper level of the NCC in order to make possible the interaction between the NMC and the NCC.

In the following section, the specifications of the LCM's and of the NCM which test, measure and control the X.25 connections at the endpoints are presented. Such specifications are based on similar experience already gained by EIN.

CONTROL OF THE X.25 NETWORK CONNECTIONS AT THE ENDPOINTS

The interface between the DTE and the Data Circuit Equipment (DCE) for terminals operating in packet mode has been standardized in the International Telegraph and Telephone Consultative Committee's (CCITT) X.25 Recommendation [2], which defines the following levels:

- -- Level 1: relating to the physical, electrical functional and procedural characteristics of the operation of the link between the DTE and DCE,
- -- Level 2: relating to the link access procedure for the exchange of frames across the DTE/DCE interface,
- -- Level 3: relating to the procedures for the transfer of packets at the DTE/DCE interface.

Thus the behaviour of the X.25 networks (X.25-nets) can be controlled by modules implemented at the upper level of the DTE Level 3. As we have said, network managers have already designed such systems to control the communication network. Other useful control systems can, however, be conceived at the same level at the endpoints of the network, thereby providing for example, information on the packet traffic of the user connected to the X.25-net (i.e., the work load), and on the behaviour of the network in response to this traffic (i.e., network performance).

Based on similar experience gained by EIN [3],[4], we would propose a control system constituted by:

-- Local Control Modules (LCM's) located at the upper of each DTE - Level 3,
-- a Network Control Module (NCM) located at the upper level of a DTE - Level 3, which collects data from the above LCM's and coordinates the control system.

Tasks of the LCM

Each LCM:

-- produces artificial packet traffic directed to other network ports (i.e., other LCM's or TS's),
-- manages the subnetwork facilities,
-- measures the traffic between the DTE - Level 3 and its neighbours (i.e., the TS and the DTE - Level 2),
-- modifies if necessary the parameters which affect the DTE-L3 behaviour (e.g., timeouts and buffer lengths) on the basis of local algorithms or of network algorithms managed by the NCM or the NCC,
-- isolates the fault conditions, advising the operator,
-- records its activity on local files and transmits the collected data to the NCM.

LCM Structure and Management

The LCM is an interlocutor at the upper level of the DTE-L3 (Figure 9). It can be used either by local or remote terminals of X.25 nets. Port 1 of the LCM is reserved for one local terminal (called the Master TTY) and can be used as the OPERATOR FACILITY. Ports 2 to 5 can be assigned to other users, while ports 6 to 8 are reserved for special services: port 6 to DROP, port 7 to ECHO and port 8 to the I/O Controller (I/O-C).

The users at local terminals manage the LCM by means of commands written in a suitable user format [3] which are interpreted by the LCM Interpreter LCMI. The users at distant terminals also manage the LCM by means of commands in a user format, but these are put into the text of the packets directed to the LCMI via the I/O-C.

Commands

The commands to the LCM can be divided into two groups: PUBLIC COMMANDS which control the way in which the LCM produces packets, and CONTROL COMMANDS reserved for the Master TTY, which control access to the LCM from the users and set the LCM operation mode.

Public Commands

a) EMIT 1

This command causes the LCM to transmit a sequence of packets (data or interrupt) directed to a network port. The text of the packets is fictitious. The length of the text and the interdeparture time interval of the packets can be selected on a time function basis or randomly. Single and cumulative acknowledgements can also be requested from the packets of the sequence.

b) EMIT 2

This command causes the LCM to transmit a sequence of packets (reset and clear) to reset or to clear an existing virtual call. The interdeparture time interval of the packets can be according to time function or random.

c) SEND

This command is used to send readable text in a single data or interrupt packet from a port of the LCM to another port of the LCM or the TS. The text of this command can also be used to carry a command directed to the distant LCM (port 8) to be executed by that LCM.

d) SWEEP G

This command sweeps all the DTE addresses belonging to a prefixed group G (for example all the ECHO processes in the nodes) in order to find out which DTE addresses in the selected group G correspond to the available services in the X.25-net.

e) CHECK X

This command causes the LCM to transmit a packet to the DTE address X in order to find out if this DTE address is up.

f) MAP G

This command causes the LCM to return to the user the MAP of the TS's belonging to the closed user group G.

g) STAT N

This command causes the LCM to transmit the N statistics to the user.

h) STOP C

This command is used to stop the execution of the command C previously given to the LCM.

Control Commands

a) PERMIT

This command is used by the Master TTY to enable remote or local users to manage the LCM.

b) ANNUL

This command is used by the Master TTY to cancel a previous PERMIT or to refuse a port request coming from distant users.

c) RESTART

This command is used by the Master TTY to restart the DTE-L3.

d) GO UP, DOWN, GO DOWN, STOP, DISPLAY, LOAD F

These commands are used respectively by the Master TTY to set up the LCM, to put down the LCM immediately, to put down the LCM gently, to stop the LCM definitively, to find out the status of the ports and to issue on any port of the LCM a set of commands stored on the file F.

LCM IMPLEMENTATION IN A MULTIMICROPROCESSOR ENVIRONMENT

The solution chosen for the LCM implementation differs from that adopted for the SCM at EIN which exists on a large computer and is therefore subject to a task scheduling Executive-process. This solution not only requires buffers and CPU time from the Host-computer, but also imposes some limitations on the use of the LCM and influences the collected measurements (e.g., it introduces a delay in the response time measurement due to the I/O operations on the interface files). To implement the LCM, a multimicroprocessor-based solution has been adopted because of its modularity and low cost.

The architecture of the multimicroprocessor configuration adheres strictly to the one adopted by EIN for the development of the EIN Matching Unit [6]. Our configuration comprises a RAM memory board and four microcomputer boards (Figure 10) connected through a common bus:

- -- L2-X.25 which performs the HDLC and the DTE link access procedure,
- -- L3-X.25 which performs the DTE packet level,
- -- LCM which performs the tasks described in Section 3.2,
- -- Utility Board (UB) which contains diagnostic software and interface software with user processes (TTY, tape, etc.

The information exchange between the boards arises by means of interrupts and pigeon-holes (Figure 11).

General Software Architecture

The software of each board is chiefly made up of a main program, initialisation procedures, I/O procedures and Timing procedures.

The main program performs the proper functions of each level. It comprises the procedures associated with EVENT VARIABLES. A procedure is called by the main only when the EVENT relative to this procedure is set up by the Input-Procedure (IP) or by internal mechanisms. During their running, the procedures of the main provide buffers and commands (data-bytes) to the modules at the upper and lower levels. To transmit this information to the adjacent levels, the procedures of the main send an interrupt to the Output-Procedure (OP). The control then passes to this procedure, which performs the following operations:

a) asks the common bus for its own board,
b) puts the address of the buffer or the value of the command in the prefixed RAM locations (pigeon-holes),
c) transmits an interrupt to the IP of the adjacent level to which the information has to be sent.

The control thus returns to the main.

As regards the IP of the addressed board, on the reception of the interrupt from the OP of another board, it performs the following operations:

a) asks the common bus for its own board,
b) reads the pigeon-hole loaded by the board from which the interrupt comes,

c) puts the address of the buffer in the appropriate input queue and sets up the EVENT variable of one of the following four procedures:

   -- lower level receive,
   -- upper level receive,
   -- lower level command handler,
   -- upper level command handler.

d) transmits an interrupt to the OP of the other board to request another buffer or command, if any, from that board.

The control thus returns to the main.

The boards have access to the common RAM, following suitable priority criteria [5]. When a board works on the common bus, the others work on the local bus.

LCM Software

The LCM software comprises:

-- initialisation procedures with regard to initialisation of the four pigeon-holes between the LCM and the L3 or UB, the start of the internal updating mechanism of the LCM and the start of the L3 by means of transmission of a suitable command to the L3,
-- timing procedures to activate main procedures,
-- I/O procedures which behave in the way described in Section 4.1,
-- the main program which consists of:

   o four procedures, activated when their EVENT has been set up by the IP, to manage the input buffers and the commands coming from the L3 or the UB,

- o two procedures, activated when their EVENT has been set up by internal mechanisms, to manage respectively the input queue of each LCM port and the transmission of buffers or commands to the L3 or the UB.

Input Buffer and Command Management

The procedures which manage the input buffers and commands coming from the adjacent levels behave in the following way:

-- UB RECEIVE manages the commands coming from the users. It controls the password to avoid illegal use of the LCM, interprets the commands from the user format to the LCM format and puts them in the appropriate port queue,

-- L3 RECEIVE manages the packets coming from the remote users. The packets directed to ECHO are echoed, the packets directed to DROP are dropped, the packets directed to local users are delivered to UB, the packets containing remote commands to the LCM are delivered to the above UB RECEIVE procedure. In addition, for any packet requesting acknowledgement, the proper acknowledgement packet is produced,

-- L3 COMMAND HANDLER receives commands from the L3 concerning information on the DTE/DCE interface (line down, line up, etc.).

Command Queue Management

This procedure manages the commands stored in the port queues of the LCM. The commands which stop previous commands are executed immediately; the others generate contexts for the transmission of sequences of packets. A time-out destroys the commands which are not taken into account within a given period of delay. A response is in any case given to the user in order to notify him that the command has been executed, is being actioned or has not been executed at all.

Context Management

This procedure manages the contexts created by the above procedure from the input commands. As requested in these contexts, this procedure generates buffers to the LE containing commands for the transmission of data, interrupt, reset and clear packets on an existing call. Obviously, before the data transfer phase, the procedure executes a set-up phase, while at the end of the sequence it executes the clearing of the call. A time out destroys the context when the sequence has not been produced completely within a given period of delay. A response is also given to the user in order to notify him if the context has been executed completely or not.

CONCLUSIONS

In this paper the structure of a control system implemented at the packet level in the DTE's of X.25-nets has been proposed. It can be integrated with the management structure, if any, implemented at the packet level in the X.25 nodes. By means of a complete management system, it is possible to implement a general adaptive control at the packet level, giving also much more information and many more statistics on the node and the DTE behaviour.

The specifications of such a control system have also been described, as well as the first implementation of the Local Control Module in a multimicroprocessor-based X.25 DTE. The implementation of a DTE controlled by the above LCM has now been developed at the University of Catania [5] using SGS-ATES Z80 micro-computers. The NCM will be applied to EURONET [7],[8] by the University of Catania within a few months.

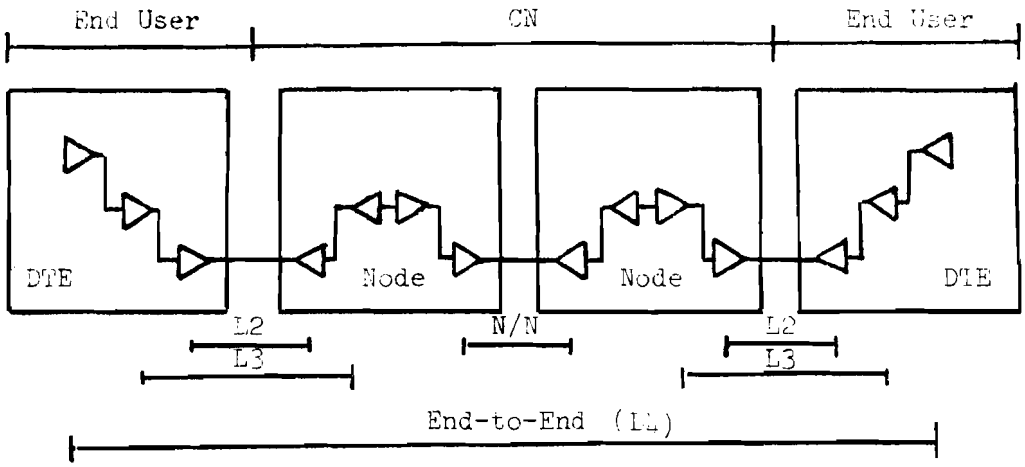


Fig. 1. Connection between two DTEs

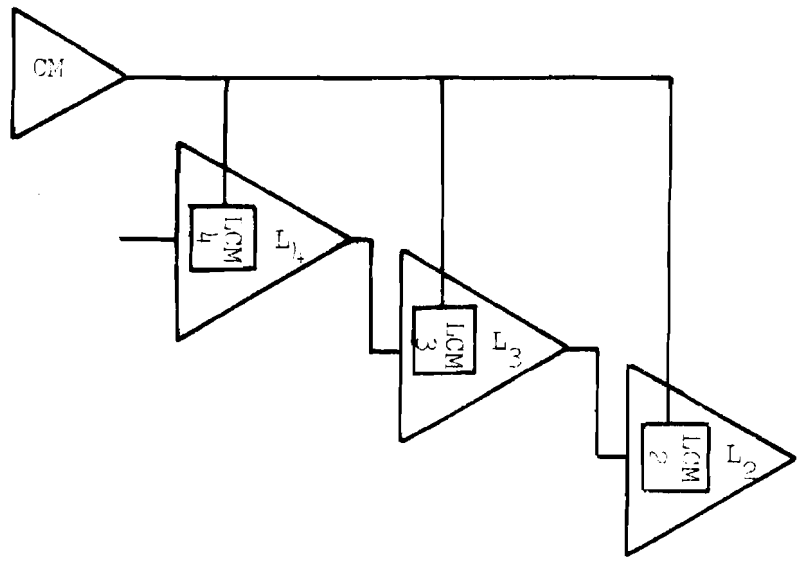


Fig. 2a. LCMs inside the interlocutors to be controlled

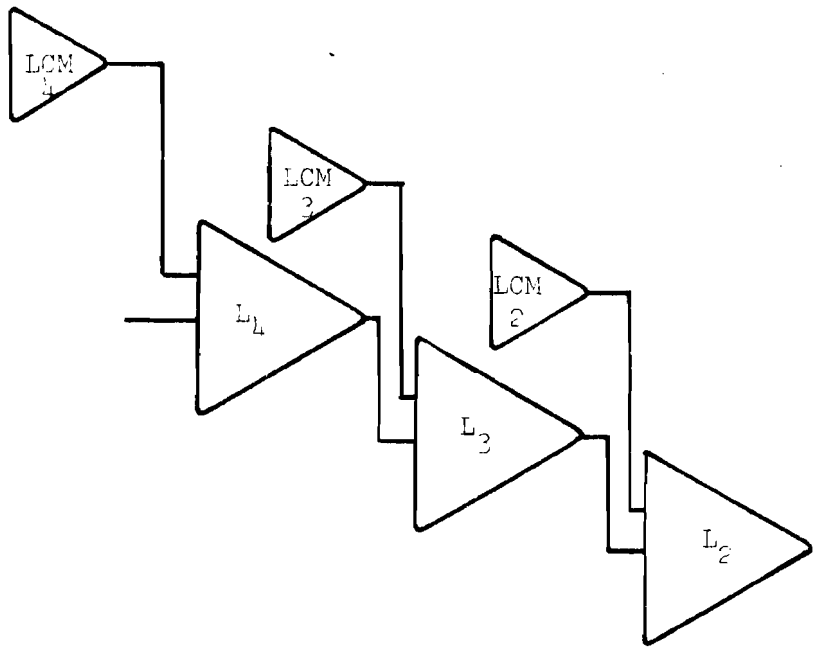


Fig. 2b. LCMs outside the interlocutors to be controlled

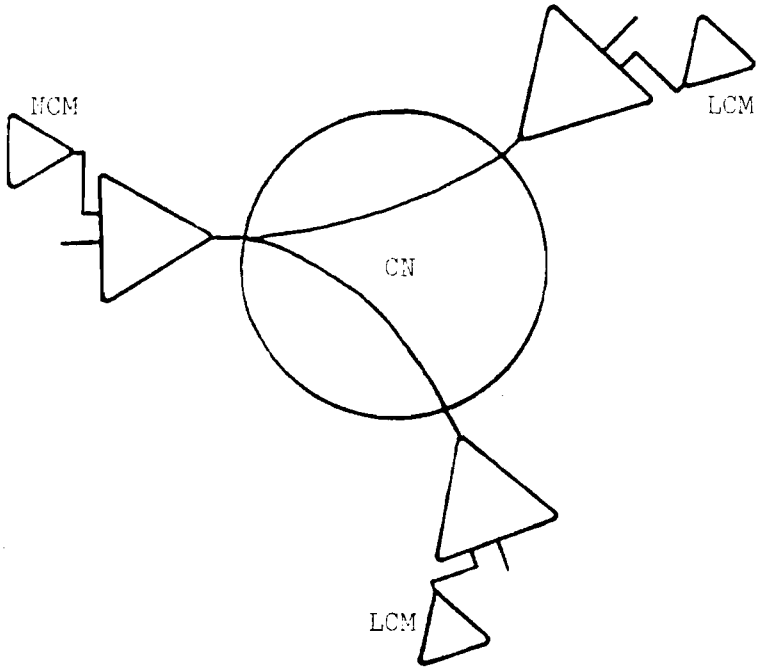


Fig. 2c. Remote coordination of the LCMs

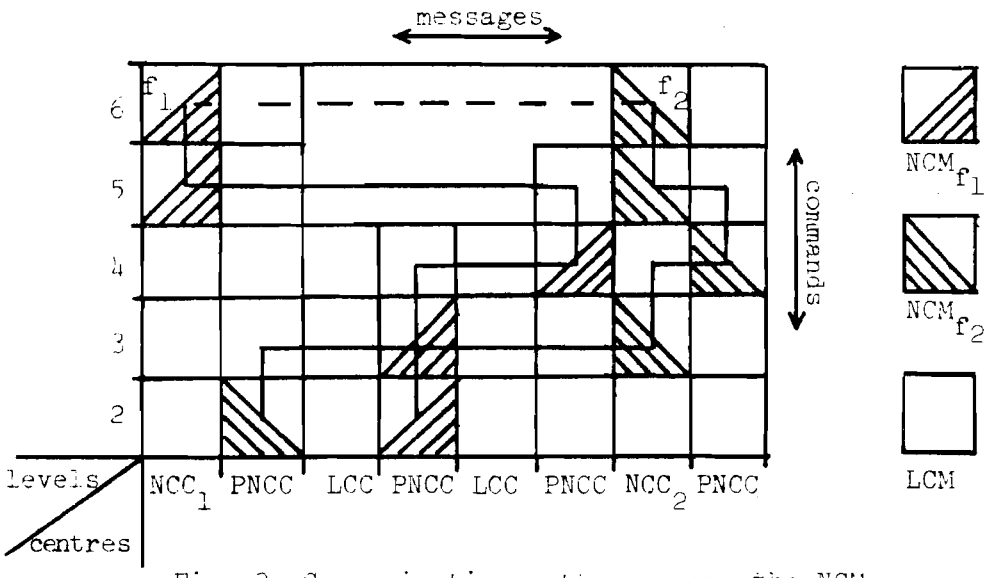


Fig. 3. Communication pattern among the NCMs at different levels and in different centres

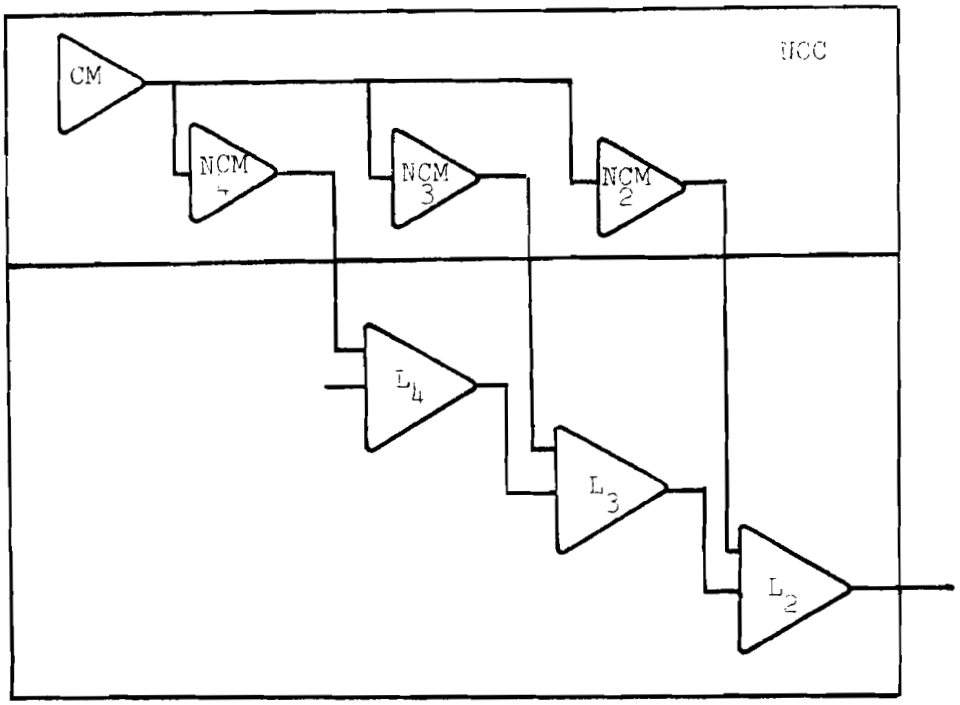


Fig. 4. Network Control Centre (NCC)

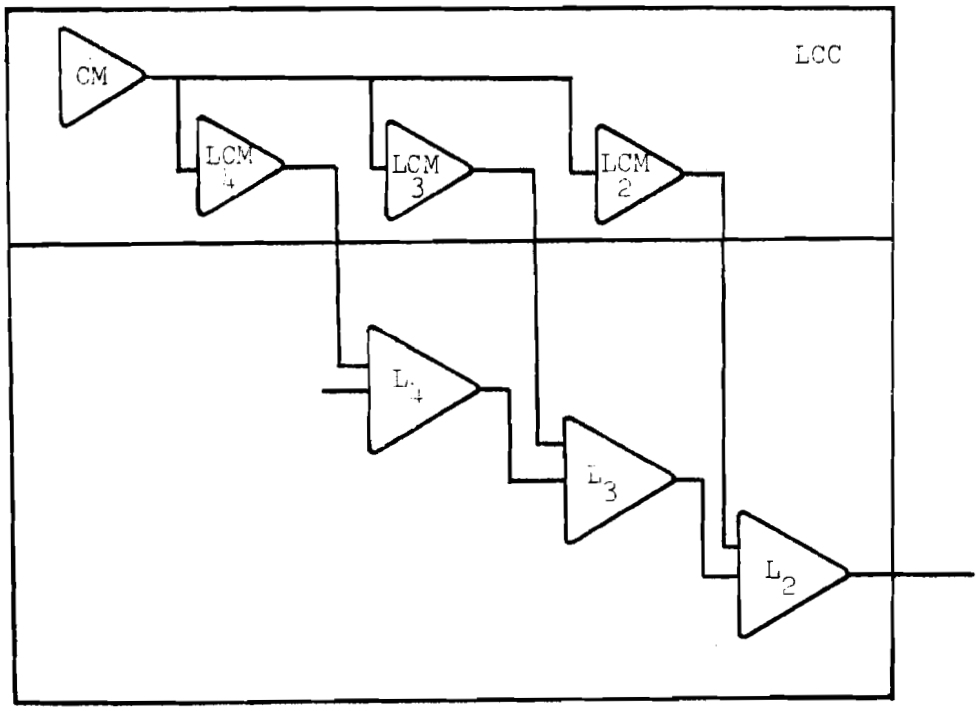


Fig. 5. Local Control Centre (LCC)

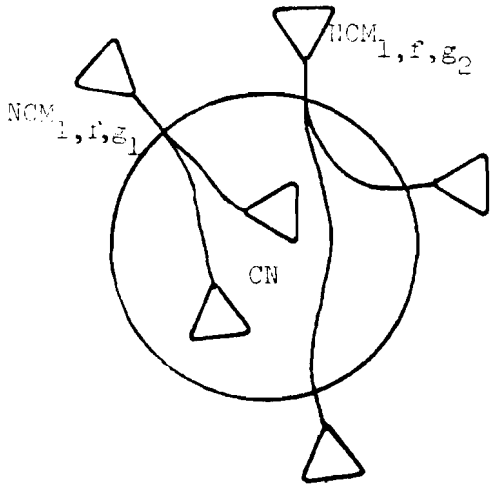


Fig. 6. Control subsystems at the same level and performing the same function, but regarding two different groups.

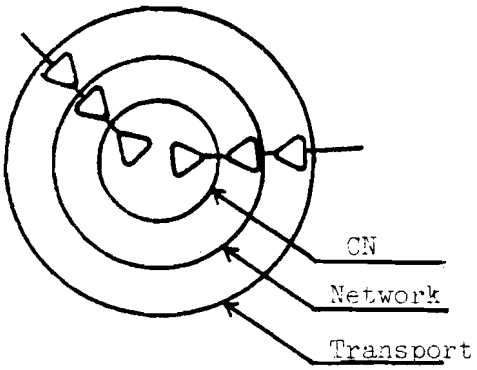


Fig. 7. Some subsystems in computer networks

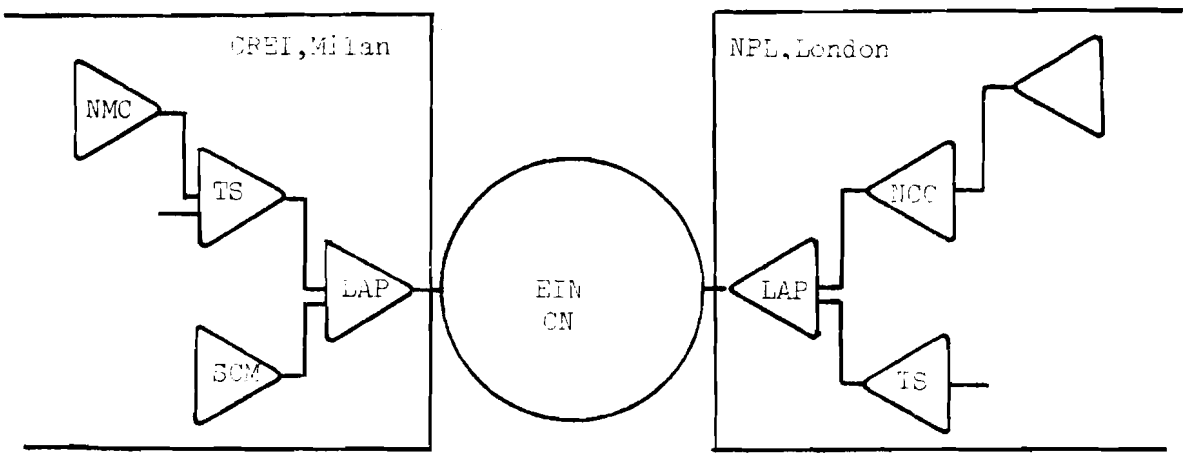


Fig. 8. Some CMs in EIN

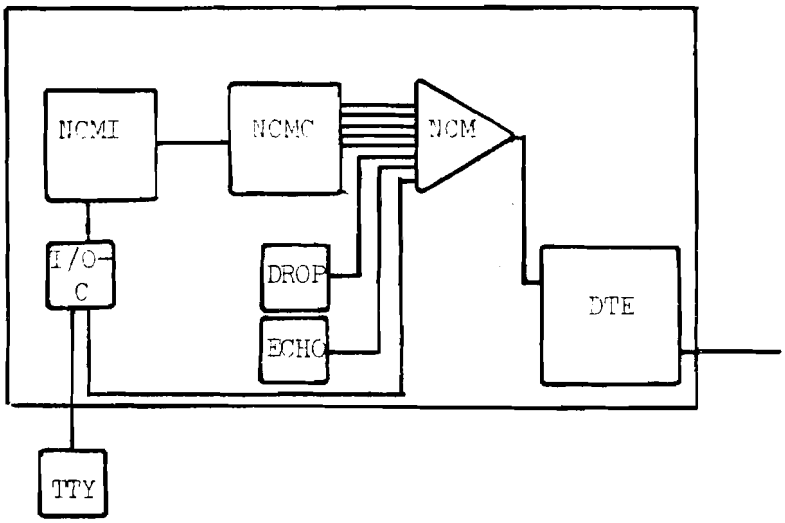


Fig. 9. Structure of NCM (or LCM)

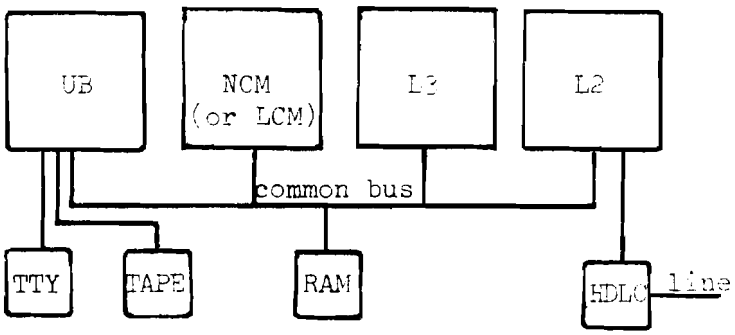


Fig. 10. Architecture of the multimicrocomputer arrangement

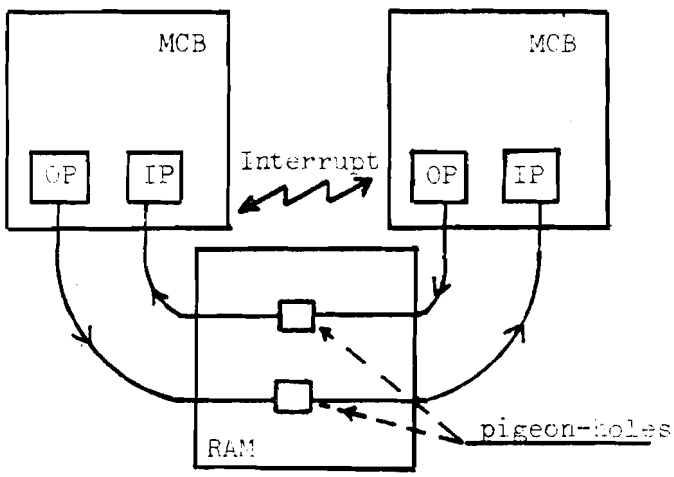


Fig. 11. Information exchange between two microcomputer boards.

REFERENCES

[1] Le Moli, G. (1973) A Theory of Colloquies. Alta Frequenza 10(XLII):493-223 E - 500-230 E, October.

[2] CCITT (1977)(The International Organization for Standardization) Recommendation X.25. Orange Book(VIII)2.

[3] Faro A., G. Le Moli and E. Repossi (1977) Specifications of the Subnetwork Control Module for EIN. EIN/CREI/77/007. Teddington, Middlesex: European Informatics Network.

[4] Alfonzetti S., S. Casale, A. Faro, V. Saletti and G. Scollo (1979) A Measurement System for the European Informatics Network Implemented on the Multimicroprocessor Interface with EURONET. Proceedings of the Eighth International Symposium on Mini and Microcomputers (MIMI), Zurich, May.

[5] Faro A., G. Messina, O. Mirabella and V. Saletti (1980) A Network Control Module for EURONET Implemented in a Multimicroprocessor Arrangement. Accepted for the Proceedings of the Fourth European Conference on Electrotechnics, EUROCON 80, Stuttgart, March 24-28.

[6] EIN (1978) The EIN Matching Unit. EIN/78/001. Teddington, Middlesex: European Informatics Network.

[7] Mangoni M. and A. Misino (1979) The EURONET Network: Origins, Reasons and Possible Future Applications. Alta Frequenza 8 - Special Issue on EIN and other European Computer Networks, August.

[8] Repichini, A.M. (1979) Network Management and Control in EURONET. Alta Frequenza 8 - Special Issue on EIN and other European Computer Networks, August.

THE END IS NIGH FOR EIN

D.L.A. Barber

As the COST Project 11 - a European Informatics Network - draws to an end, and plans are maturing for a new joint project, this article traces the evolution of EIN, introduces the participants who worked together to build and operate Europe's international computer research network and comments on their role in developing techniques for its application. It then examines the part played by the project in providing a focus for co-operative research, considers its influence on contemporary events and concludes with a review of some lessons that may be learned from this unique international venture.

INTRODUCTION

Just before Christmas 1970 I was sent as a delegate to an EEC meeting in Brussels to discuss the idea of a European pilot Informatics Network. The proposal was one of several that had been made by the European Communities PREST Committee (Scientific and Technical Research Policy), which had met under the chairmanship of M. Aigrain in 1968. These Aigrain proposals were taken up by the COST Group (European Cooperation in the field of Scientific and Technical Research) during 1969, and a number of

study groups were formed to examine them in detail. It was the first meeting of one of these groups that I attended in Brussels where, to my surprise, they asked me to become its chairman.

The study group met a further four times and by mid 1971 had prepared a report which was strongly in favour of the establishment of an informatics project, based on the construction of a packet switching communications network and the conduct of a joint research program to explore its applications. As a result, an international Agreement aimed at bringing such a project into being was formulated, and was signed by nine European Governments, together with EURATOM, on 23rd November 1971. The original Signatories - France, Italy, Norway, Portugal, Sweden, Switzerland, the United Kingdom, Yugoslavia and EURATOM were later jointed by West Germany and the Netherlands.

The Agreement stated that the project should:

-- facilitate research into data processing problems,
-- permit the sharing of resources at Centres,
-- allow the exchange of ideas and the coordination of research programs,
-- facilitate the comparison of ideas for national networks,
-- promote the agreement of standards and networks,
-- be a model for future networks whether for commercial or other purposes.

A technical annex to the Agreement estimated that the project would last five years. In the first two years a communications sub-network would be constructed linking Centres nominated by the Signatories. The computing systems at these Centres would then be linked together to form a Computer Network for advanced research, to be conducted over the remaining three years. The project was to be managed by a Committee of representatives of Signatories, and an Executive Body with a Director and three assistants would be in charge of day-to-day

operations. When the project began, it was my good fortune to become the Director.

It is in the nature of research projects that some tasks prove more difficult than anticipated and EIN proved to be no exception. Accordingly, the Management Committee decided to extend the duration of the project, firstly to the end of 1978 and subsequently for a further year. So, the end of EIN in December, 1979, will mark for me, the completion of nine exciting years of network research. During this period there has been a dramatic series of developments that would have been branded as science fiction could they have been imagined at that first exploratory meeting in 1970.

THE PACKET SWITCHING SUBNETWORK

At the time when EIN was conceived, there were widely differing views on the form that future data networks should take and, indeed, on whether special facilities for data communications would ever be required, for many then doubted whether data traffic would every grow significantly for many years.

The principles of packet switching had been proposed in the early 1960s, and research work had been carried out in the USA by ARPA and by NPL in the United Kingdom, but the idea of an international packet switching network was quite new. It was, therefore, something of a bold step for the study group to propose that the basis for a new project should be such a network. Fortunately, events have proved the rightness of the decision, because the majority of public data networks now in service, or being commissioned, are based on the packet switching concept. Indeed, recent studies suggest that it will ultimately play the dominant role in future world telecommunications, by carrying voice as well as data traffic.

The specification for the EIN subnetwork was devised by experts from the participating Centres meeting under my chairmanship. Eventually we reached a compromise between

views which later crystallised into the datagram versus virtual circuit debate. The subsequent analysis of tenders was also done by a Working Party of experts who marked them according to a predetermined marking system. In this way, we made an independent objective assessment, even although many different countries were involved in the selection of a contractor. As a result, a fixed price contract was awarded in October 1974 to SESA (France) and Logica (UK) as main contractors, with Selenia (Italy) and FIDES (Switzerland) as sub-contractors. This consortium designed, developed and installed the EIN subnetwork that was handed over to the Centres on schedule in May 1976.

A great deal of experience was gained in the design and development of the EIN subnetwork and this is well described in some 200 papers and reports published by the EIN Community over the past few years. In addition, a mass of information and statistics about the operation of the network has been collected and is freely available for detailed analysis by interested research workers. Currently, the subnetwork is being phased out of service as the participating Centres transfer their computer systems to EURONET, the international public packet switching network that is now becoming operational.

Already the EIN Centres are playing an important role in the assessment of EURONET because of their unique experience and their network measurement tools that have been developed in past years. There is still much to be learnt about the virtues and vices of the public network and an area of special interest will be the extent to which EURONET can replace adequately the user-oriented services that were a feature of the EIN Network Control Centre developed by NPL. This NCC played an important role in the management of the Subnetwork, but has now been closed down.

THE PRIMARY CENTRES

The Signatories that connected computer Centres to the communications subnetwork financed their EIN activities by what is called 'Concerted Action', whereby each Signatory was responsible for meeting its own costs, plus a share of the subnetwork costs, probably amounting to some 60 M BF for each Primary Centre over the life of the project.

The five Primary Centres, nominated by Signatories when the Project began were:

CREI - Centro Rete Europea di Informatica - Italy
ETH - Eidgenoessische Technische Hochschule - Switzerland
IRIA - Institut de Recherche d'Informatique et d'Automatique - France
JRC - European Communities Joint Research Centre, Ispra Establishment (Computing Centre - CETIS)
NPL - National Physical Laboratory - United Kingdom

THE SECONDARY CENTRES

In addition to its Primary Centre each Signatory was allowed to nominate any number of Secondary Centres to be connected to its Primary Centre by national leased lines. Of the several EIN Secondary Centres, those that took part in the 1978 presentation (see below) were:

AERE - Atomic Energy Research Establishment - United Kingdom
CICG - Centre Interuniversitaire de Calcul de Grenoble - France
CILEA - Consorzion Interuniversitario Lombardo per Elaborazione Automatica - Italy
CSATA - Centro Studi e Applicazioni di Tecnologie Avanzate - Italy

ASSOCIATED CENTRES

Once the network was operational, some Signatories nominated Associated Centres, not connected permanently to the network, but capable of access through the public switched telephone network to a number of Primary and Secondary Centres. The Associated Centres that joined in the 1978 presentation were:

GMD - Gesellschaft fuer Mathematik und Datenverarbeitung - West Germany

QZ - Stockholms Datamaskincentral foer forskning och hoegre utbildning - Sweden

RSS - Raziskovalna Skupnost Slovenije - Yugoslavia

THE ROLE OF THE PARTICIPATING CENTRES

In many cases the experts engaged in the early activities of the project came from the participating Centres, although some Signatories without Centres also provided experts for the various working parties. In parallel with the common activities, the Centres made their own plans for the installation of the Network Switches, the provision of communication links and so on. In addition, they began to consider the problem of interfacing their own computer systems, and in some cases networks, to the international subnetwork. This led to the emergence of a variety of solutions.

At that time it was a strength of the project that various interfacing methods would be adopted, because this enabled a comparison of techniques to be made. As ideas evolved and experience was gained, most of the Centres made changes to their original plans and the final arrangement was that each Centre used a mini computer between the network and its own system. These mini computers formed a ring of interfaces matching the Centre's systems to the subnetwork. These systems are shown in the Figure which illustrates EIN at its greatest extent. A wide variety of different computer systems and networks is depicted and these indicate the enormous efforts

made by the participants during the conduct of the project. However, it was seldom the case that all systems were simultaneously available because the prime aim was not the provision of such services. Indeed, the resources of the project were far too low for this to be possible except for demonstration purposes on special occasions.

Matching the complex computer systems of EIN to the communications network proved relatively straightforward, but adapting them to interact with each other was a much more difficult task. The now generally accepted solution lies in the agreement of a number of levels of protocol or procedure carried out by software in, or associated with, these systems. This approach, which has been well described in the literature, became the basis for international standardisation within CCITT and ISO. However when EIN began, no such concept existed and the pioneering work in this area conducted by the project has undoubtedly been a notable contribution to the subject.

By early 1978, the development of protocols within EIN had reached such a stage that a reasonable degree of interaction between Centres' systems was possible. The Management Committee therefore decided to give a public presentation of the activities of the participants and this was held on 5th April 1978. The ten Centres mentioned above staged a simultaneous demonstration of the network and its facilities. A wealth of information was gained by those taking part and this led to some reappraisal of the work. In particular, the need for an effective way of coordinating centres through a teleconferencing system was established as a result of using the experimental 'Conclave' scheme provided by NPL. As EIN draws to a close, plans are being made to set up an operational system at JRC Ispra as a focus for future European projects. This will be based on the COM system recently developed in Sweden.

ADAPTATION TO EURONET

In parallel with the growth of EIN as a research project, a plan developed for a network for the dissemination of Scientific and Technical Information within the European Community. The CEPT (Conference Europeenne Des Posts et Des Telecommunications) agreed to provide a packet switched service to support this network. Eventually, the overall network became known as DIANE (Direct Information Access Network for Europe) while the communications component was called EURONET. When it was agreed that third party traffic might also be carried on EURONET, the way was clear for it to replace the EIN communications subnetwork. However, this requires EIN Centres to adopt the CCITT (International Telephone and Telegraph Consultative Committee) recommendation X.25, which specifies a standard interface between Subscribers' Computer Systems and a public packet switching network.

When the EIN subnetwork was specified in 1973, it was impossible to foresee the details of any future CCITT standard, and indeed it was difficult to predict when a public packet-switching service would be available. The consensus of opinion at that time favoured the Datagram type of network and this was, accordingly, adopted as the basis for EIN. This, of course, conditioned the design of interfaces for the systems at the Centres.

By the time EURONET was designed, the CCITT had agreed that the use of Virtual Circuits was more suitable for a public network. The EIN Management Committee therefore decided to develop an adaptor box, suitable for interposition between an EIN switch and the X.25 public network, to allow the changeover to EURONET to take place using permanent virtual circuits with minimum disturbance to Centres. This development, which used a multi microprocessor architecture was completed early in 1978 and has since furnished ideas for further adaptor boxes known as EIN Matching Units. EMU's include an X.25 interface tester, a network exerciser and adaptors for higher levels of protocol.

With the experience gained we can now quickly design and build adaptors for a wide variety of practical requirements in interfacing computers to a communications subnetwork. This will become increasingly important in the next few years.

PROJECT MANAGEMENT

The basic tasks of the Management Committee and the Executive Body were, of course, laid down in the Agreement. But there were many unexpected problems encountered during the progress of the project that demanded action by the Committee and the Executive Body. Major example were the monitoring of contracts, the development of EMU and the coordination of Centres' activities. Other tasks have been the representation of the project at public conferences, at CCITT and ISO meetings and the discussions with other projects such as EURONET. These external interactions were facilitated by the IFIP Working Group 6.1, which I chaired from 1976 to 1979.

In all these EIN activities, a number of valuable lessons have been learnt about the management of an international cooperative research project, with distributed participants.

For the most part, cooperation between Centres was reasonably satisfactory because, once each particular objective had been agreed, each Centre was able to work independently to reach it, making its own decisions and using its own resources as required, in the manner laid down by the original agreement.

But with such a complex project a more detailed control of the work is often desirable, because the success of the whole project relies on the proper interworking of the systems of the individual participants. This has proved hard to achieve through the committee structure adopted for EIN.

As an example, the specification for the Transport Station protocol was implemented by each Centre. But naturally, there are various ways in which such specifications can be interpreted, and so separate implementation differed. Effective interworking

in EIN proved not to be possible until the situation had been clarified and changes made to some versions of the Transport Station. But even then secondary problems arose because the implementations ranged from partial ones providing only a basic service, to one that included all kinds of checking for protocol violations to make a comprehensive and robust package. With independent designs there is no easy way to assess their relative completeness and it is impossible to be sure that they will interwork under all future circumstances; furthermore, any changes that prove necessary cannot be introduced simultaneously throughout the network.

The Executive Body endeavoured to coordinate the activities of Centres in these kinds of tasks but, without a more direct involvement in the work than was provided for by the Agreement, this proved a very taxing and onerous task. Even in non-technical areas the coordination of the actions of Centres was difficult when a really precise objective was the aim.

A good example was the organisation of the maintenance of the Network Switches. Originally these were procured by Centres using their own funds and by separate negotiations with the contractor. This was necesssary because no mechanism was provided in the Agreement for the Centres to be legally represented by the Management Committee and the Executive Body.

Nevertheless, after prolonged discussion the contracts were made similar although they were legally independent. While this was clumsy but workable for the procurement of the switches, the same scheme was far less satisfactory for dealing with their maintenance. This is because they all had to interwork together within the framework of the subnetwork, which should preferably have been treated as one complete system.

Unfortunately, from a legal point of view, no-one owned the subnetwork so no-one could negotiate a common maintenance contract for all Centres. But if, for example, the Executive Body had been able to place a simple contract for network maintenance, and then charge each Centre accordingly, a much more satisfactory outcome would have been the result.

Problems of this kind bedevilled the EIN project almost from the start, so it is a great tribute to the goodwill and enthusiasm of all participants that, for the most part, the initial objectives have been very satisfactorily achieved.

THE FUTURE

Since EIN began, astonishing changes have occurred in the technological environment, brought about, in part, by the influence of the project itself. Public Packet Switching networks are becoming commonplace; a strong community of informed network users has been created by the Signatories, and Europe is currently a front runner in the development of Teleinformatics. This is the background against which the project will draw to a close, as the subnetwork is superseded by the use of EURONET.

The achievements of EIN seem to have been generally beneficial and there is great goodwill towards the idea of another similar project, this time aimed at research into the applications of teleinformatics systems, rather than their design, as was the present one. It is too early to say what form a new project may take, but it will be a COST project related to the EEC Commission's new four-year action in Informatics development, and will therefore be open to non-community countries. As this paper has indicated, experience with EIN has revealed a wealth of problems that need to be solved if the maximum advantage is to be gained from the investment now going into the new public data networks, and this gives a rich menu of possibilities for further coordinated research at the international level.

However, the problem of deciding exactly what research should be done in Teleinformatics seems far more difficult than ever before. Ten years ago some of us were sure that packet switching was the way to go - and this has proved to be right. Today, there seems no such clear objective. For we now have a plethora of new ideas like Teletext, Viewdata, Burotics and perhaps even Homotics* to take into account.

Developments like these of course stem from the microelectronics revolution and will continue in abundance as we achieve current expectations of the ultimate logic power and storage capability per chip. This rapidity of change makes crystal gazing an immensely difficult task - yet failture to anticipate and respond to such change can well render some of our research work irrelevant. Certainly, the next ten years promises to be even more exciting than the last, but in what respects remains, as ever, anybody's guess.

---

*Homotics - the study of HOMOsapiens at HOMe with informaTICS

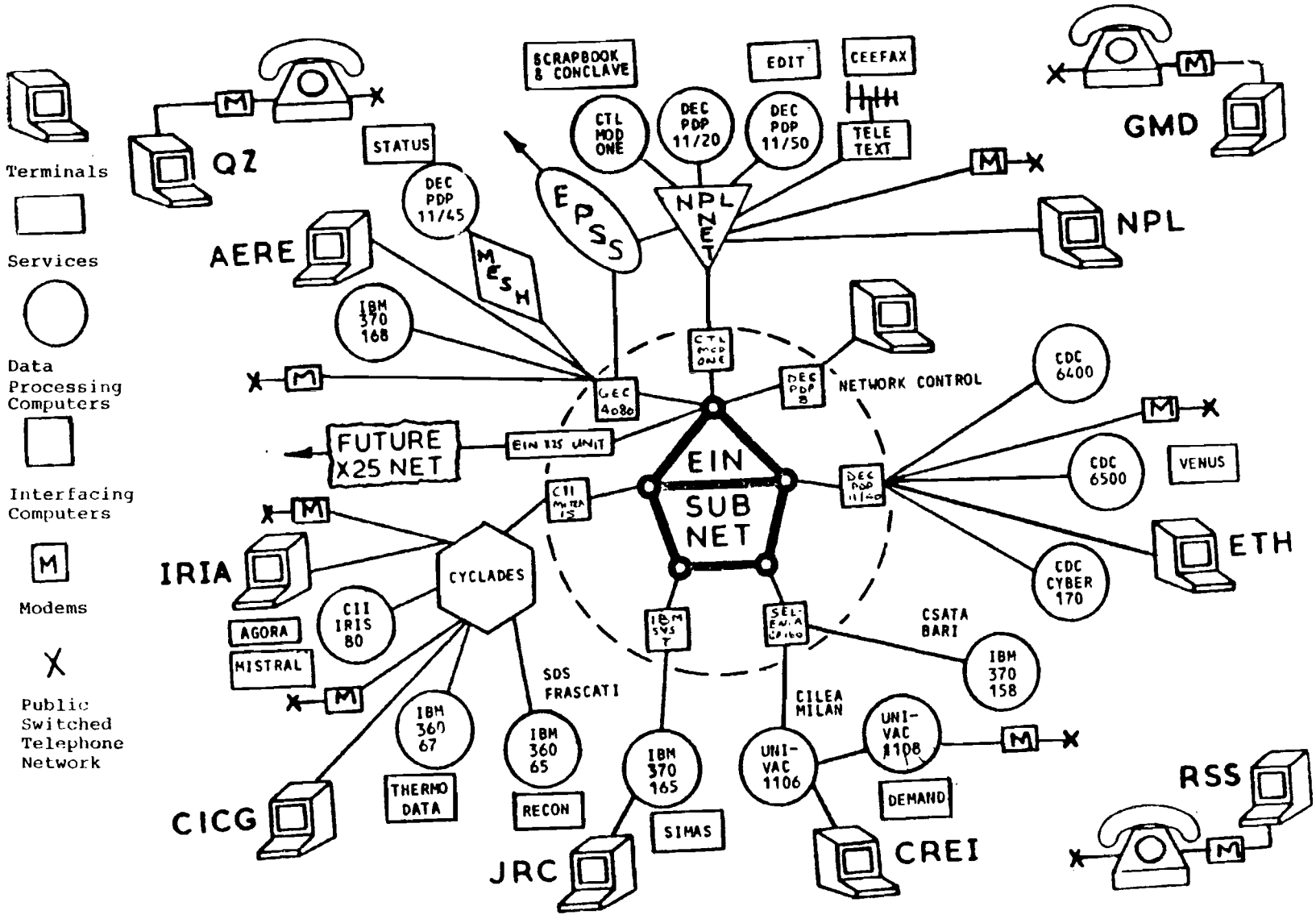


Figure 1. Some systems joined to EIN.

STUDY OF A COMPUTER NETWORK

I. Margitics

The following short report will describe a study of the setting up of a computer network consisting of homogenous and heterogenous domains. The modelling and verification of an end-to-end protocol characterizing the network will also be performed.

INTRODUCTION

The computer network to be studied has a dual character:

-- it is experimental, in order to put various higher-level protocols into operation (e.g., several protocols of the VTP protocol family) and
-- it provides a service, as the participating hosts must supply resources to the various users.

It should be noted that this network is not yet in operation; it can be considered as a project to be realized in the future.

NETWORK TOPOLOGY

The computer network to be set up, the subject of our examination, is basically of a heterogenous nature. This heterogenous network consists of homogenous and heterogenous domains (i.e., regions). The hosts to be found in the homogenous domains are indicated in Figure 1.

As can be seen, the network to be examined consists of the following domains:

S: Siemens TRANSDATA-NEA consisting of Siemens mainframes of the types 4004 and 7000 and front-ends of the types DUET 9685 and 0687,

I: IBM's SNA consisting of IBM mainframes of the types 370 and 4341 and front-ends of the types 3704 and 3705,

A: An "external" domain of a heterogenous character having a service similar to, but not identical with the X.25,

E: An "external" domain supplying X.25 services with an end-to-end transport service. This domain is almost identical to the EURONET network,

T: Two types of terminal sub-domain:
- -- independent network terminals capable of using X.25 services,
- -- terminals having basic mode (i.e., "traditional") line procedures and types bounded to the Siemens mainframe, e.g., TRANSDATA 8152 and 8161. This domain belongs to the domain "S".

From the viewpoint of network management (i.e., administratively), these domains can be split up into two categories:

-- internal: I,S,T and
-- external: A,E.

The interface processor indicated by "IP" in the center of Figure 1 has no node (e.g., packet switching) functions. In turn, it has to match the S and T domains to the other; in other words it has to perform protocol conversion functions depending on the directions.

The interface processor has a multimicro processor layout based on Zilog 80 processors. This equipment is commercially available, complete with an X.25 and IBM 3270 emulator service realized by the marketed software. Additional services, e.g., end-to-end transport control and virtual terminal functions, must be developed additionally. These auxiliary services can be provided by reentrant programming.

More specifically, the various connections realized by the IP between the domains, i.e., the protocol conversions, are as follows:

1. IP-S Domain

   Let us assume that the S domain consists of Siemens mainframes type 4004 and 7000 having FEP's type 968X and a terminal system supported by the Siemens' standard TP software.

   Taking into consideration that the Siemens domain has to be connected both to the A and to the E and I domains, which provide various transport services, and that Siemens has up to now realized only the LAPB (HDLC), the Siemens connecting module must have the following structure (Figure 2).

   The software module facing the DUET is a DSRE module emulating a Terminal Concentrator (Datenstation Rechner Emulation).

   There are two possible methods of realizing error and flow control:
   -- end-to-end control implemented on top of the X.25,
   -- stepwise error and flow control provided by the packet level of the X.25.

   As the domains to which the S domain must be connected have both possibilities, the software module must provide both facilities.

If the connecting port in the S domain is a Datenstation Rechner port, the Virtual Terminal will not be converted to real terminal handling. This virtual terminal service called VTSU (Virtual Terminal Support) more or less corresponds to the services of the internationally defined Virtual Terminal Support.

2. IP-A Connection

Due to the fact that the transport services of the A domain will be performed on the X.25/level 3, there is no additional transport station software in the A domain connecting module (Figure 3).

Moreover, as the services of the VT applied in the domain A differ from those of the domain S, it is necessary to incorporate the module indicated by VTC (Figure 3) which performs the necessary conversion.

As the transparency of the Link Access Procedure in the domain A is provided by the method applied in the BMC procedure (e.g., DLESTX at the beginning of a block and DLE ETB at the end), it is necessary to convert this "byte stuffing" to bit stuffing defined in the HDLC and in turn. This conversion is performed by the BSF module.

3. IP-E Connection

As the domain E supplies both X.25 and end-to-end transport control without modification, there is no need for any additional conversion or matching modules; in other words, the port can be in the X.25 module.

4. IP-T Connection

There are mainly basic mode synchronous terminals in the T domain, primarily of the Siemens type although independent of the S domain (i.e., they are not connected to Siemens FEP's). These terminals use primarily the services of the I domain, but they can also exploit the resources of the S, A and E domains. When connecting them, the PAD functions,

according to X.3 of CCITT, cannot be used because of the synchronous character of the terminals. The necessary conversions are achieved by means of the RVTC module. Thus they can be connected as shown in Figure 4.

The software structure of the above figure presupposes that there is no end-to-end transport control in the terminals of the T domain. Taking into consideration that several X.25 terminals can be in the domain as well, they can be connected directly to the X.25 port of the IP.

Summarizing the above, the overall software structure of the IP can be drawn as indicated in Figure 5. An internal protocol is shown in the center of this figure which has

-- to coordinate the operation of the various connecting modules, and
-- to supply several administrative functions (communicating with the operator of the IP).

It should be noted that there is also another possibility of directly connecting the I and S domains. This connection has been developed by Computer Konstanz GmbH and is called TRANSIT SNA. A detailed description of this connection is beyond the scope of this report; it should only be noted here that in this case the Siemens FEP (e.g., DUET 9687) emulates the IBM 3790 controller.

This kind of connection gives a higher performance than a connection established through the IP. However, to increase the reliability of the whole system by duplicating the connections and to give the user of both domains greater opportunity of accessing the resources, it would seem reasonable to realize both the above kinds of connections.

THEORETICAL WORK OF THE CASE STUDY SYSTEM

It is to be expected that systems similar to those of this case study will become more and more common, and

there are some - primarily internal - protocols (such as the X protocol in our case) which are not yet standard (e.g., they still have to be tested functionally). For this reason, it was decided to formulate a more or less universal method for the modelling and verification of protocols.

There are already several well-known and elaborated methods for modelling protocols, e.g.,

-- modelling with finite automata,
-- modelling by graphs,
-- assertion proof technique,
-- description by high-level program languages.

These methods have already been evaluated and have even been compared.

It was our aim to develop an automated (i.e., computerized) evaluation method on the basis of one of the above mentioned modelling techniques. After having made a short study of various other methods, and taking the promising development of the graph methods into consideration, we chose the graph modelling technique.

As is already known, there are several graph modelling methods (e.g., Petri nets, UCLA graphs, and Nutt's E-nets). We felt that, due to its compactness, one of the most appropriate descriptions would be given by E-nets. The basic elements of E-nets are already well-known; it is therefore unnecessary to give a detailed description here. It should, however, be noted at this point that the variables making up an attribute vector of the attribute token are particularly suitable for describing flow parameters when modelling, for example, the end-to-end transport protocol.

Formerly, Danthine gave a description of the end-to-end transport protocol by means of E-nets. He reported on the results at the COMNET '77 in Budapest. This paper presented only the graph of the session negotiation phase, which was however incomplete: it did not have the time-outs.

This method of applying E-nets differs from that described originally by Nutt in two ways:

1. Time handling is performed according to Merlin and not to Nutt,
2. Danthine uses the following primitive which is not considered by Nutt as a primitive (Figure 6).

This graph module can be regarded as an extension of the X and F primitives of Nutt: the input conditions are the A and B locations; r is the input (perhipheral) resolution location as input condition; M, W, R are the output locations indexed by 0 or 1, respectively. The physical meaning of these output locations is:

-- M: the message to be transmitted,
-- W: waiting state,
-- R: response (e.g., to the subscriber).

As has already been mentioned, this graph module does not correspond to any of the Nutt primitives. (However it can be constructed from the previously mentioned primitives).

As we wanted to use the formal description given by Nutt, we decided to apply only the Nutt primitives and macrographs, and not the one indicated in Figure 6. At the same time, we accepted Merlin's time handling because it may be considered as the generalized version of Nutt's time handling. Merlin's time consideration is particularly suitable for modelling various time-out mechanisms.

In order to be able to verify the method to be developed, we had to choose a well-known and properly operating protocol which had been tested previously; in other words we had to have references for our method. We thus chose the end-to-end transport protocol of CYCLADES as described in the SCH.569 documentation. This protocol has been verified by various authors and the results were partly available. It is, however, worth mentioning that a thorough study of this protocol led to

the discovery of the following desynchronization situations.

Let us suppose that one of the participating Transport Stations sends a Flow Initiation command (telegram) to the other with the recommended parameters of the flow to be opened. These parameters will be accepted by the distant Transport Station and another FL-INIT command will be returned by the parameters valid for the opposite direction (the flow is duplex). Any of these parameters will be refused by the "local" TS; it will send a FL-TERM command to the distant station and, either with or without time-out, will transfer to OFF (quiescent) state.

Let us now consider a real communication medium and let us suppose that the above FL-TERM disappears. As the distant station receives no FL-TERM, it transfers to the ON state, i.e., the two stations are completely desynchronized. The result is that the ON state station starts to send letters, but does not receive ACK's. After a predefined number of retransmissions, it closes the flow by sending a FL-TERM. In other words, this means that the session negation phase is not recoverable on its own.

The graph model of the complete protocol is fairly complicated: it consists of more than 250 locations (not including the number of locations which model the real communication medium). After constructing the graph model of the protocol, in principle we have the possibility of verifying and evaluating the protocol:

-- either to construct and analyze the ETM (Error Token Machine) described by Merlin and used by Danthine (the ETM is a sequence diagram of the marking available, including the erroneous states),
-- or to describe the graph model by means of a transition system, e.g., Vector Replacement System (VRS), and to construct a reachability tree using the VRS. By means of the reachability tree, it is possible

to provide certain criteria for the erroneous behavior of the protocol, e.g., for:

- o deadlock,
- o desyncrhonization,
- o tempo blocking.

Well defined criteria were established for the first property (i.e., deadlock), and in the case of the second property, for certain classes of protocols (for protocols of identical levels). The extension of this method to other classes of protocols is the subject of a further study.

In order to computerize this method, we decided to develop a program which could evaluate the above parameters (i.e., deadlock and desynchronization). Although several simulation programming languages were available and we had written the first modules in SIAS (Siemens Ablauf Simulator), because of the experience gained with SIAS, we rewrote the whole program in FORTRAN. In spite of the fact that we used the FORTRAN language (and not SIAS or the similar GPSS program), the runtime of the program of the whole graph model exceeded 800 CPU seconds, under the control of the BS 2000 operating system. This indicates that computerization of the protocol verification is not a "cheap game".

The Transition Scheme Chosen

Let us now consider the above method in more detail. As is known, a transition scheme can be characterized by the triplet:

$$(Q, \Sigma, \rightarrow) \quad ,$$

where Q is the set of states,

Σ is the finite set of the transitions,

$\rightarrow$ is the set of mappings, which assigns to each state and transition, a new state, e.g.,

$$\rightarrow \subseteq Q \times \Sigma \times Q \quad .$$

The possible transition scheme we have chosen is the VRS (Vector Replacement System). The VRS can be characterized by the following triplet:

$$(q_o, U, V) \quad ,$$

where Q is the set of r dimension vectors of states (e.g., the dimension of the VRS),

$\sum$ is the ambiguous set of indices.

For each element of this set, U and V are r dimension integer vectors. U is the test vector and V the replacement vector.

The transition from a q state to a q state can take place if

$$q + U_\sigma \geq 0 \quad , \text{ and } q + V_\sigma = 0 \quad .$$

It can be proved that to each E-net there is a VRS, which is identical with the net, if we establish certain preconditions for the resolution and transition procedures.

Although the proof of the above statement is beyond the scope of this report, it should, however, be mentioned here that, when proving it, the assumption must be made that the conditional parts of the resolution procedures can have only the following expressions:

$$V \geq V_{const} \tag{1}$$

where V is either the marking of a location, or an attribute, or a global environmental variable. (It should be noted that V can represent more than one expression, and that the logical relationship between them can be logical OR or AND). The condition expressed by (1) will be fulfilled as a minimum in our case.

After introducing the transition scheme, we can sketch the method for evaluating the most important characteristic of the protocol, e.g., deadlock. (A similar method is also valid for the desynchronization).

It should be noted that in the design phase of a protocol it is very difficult to evaluate the tempo-blocking properties: this depends very heavily on the method of implementation, and will be checked later.

The Construction of the Reachability Tree

As is well known, the reachability tree represents the generation of reachable states consisting of the initial state (root) branches and leaves. The construction of the tree corresponds to the diagram in Figure 7. It is possible to prove that the construction of the tree can be performed in a finite number of steps. The tree is constructed by means of the VRS as follows:

1. The initial state is q_o,
2. It can be advanced from a given leaf so that it is possible to examine whether the relation:

$$U_o + q \geq 0 \quad ,$$

is true for each $\sigma \in \Sigma$. If this is the case, then a successor of

$$q' = q + V_\sigma \quad ,$$

will be added to the tree,
3. If there is no further transition from a leaf, the branch will be completed by END,
4. If there is a return on a branch (repetition of states), it will be closed by LOOP.

After constructing the reachability tree, we can formulate the condition for the deadlock-free operation of the protocol, using the terminology of the graph.

When constructing the reachability tree according to the previously described algorithm, we omitted:

-- the global variables,
-- the attributes, and
-- the resolution and transitions.

It is obvious that, if none of the transitions appear in any of the loops, then the number of firings which can be performed can only be finite; consequently there is a state of deadlock. If a given transition is a component of each loop and no branch or subtree ends freely (labelled by END) having originated from its node, then the event examined is alive. Consequently, those transitions which are included in each loop and having the previously mentioned properties, are deadlock free, and the reachability tree does not have a subtree labelled END.

The above general algorithm can be modified so that if a subtree terminates in a state which possesses a token only in an "output peripheral location", then this subtree will be omitted from the tree (i.e., according to the former definition, it cannot influence the deadlock character of the graph).

In the method described up to now, the features of the E-nets havenot been taken into account. Let us now consider the special features of the E-net (e.g., procedures and attributes) which will lead to the simplification of the reachability tree.

Attributes of the E-Nets to be taken into Consideration

The following features of the E-nets:

-- attributes,
-- global (environmental) variables, and
-- procedures (resolution and transition),

will be taken into consideration in order that the tree may be walked round (in a preordered sequence) and we will examine whether a given transition can actually be fired or not.

In the case of the X and Y primitives defined by Nutt where two inputs or outputs are possible, we consider two states and two directed arcs corresponding to the resolution procedure while constructing the tree. While walking round the tree - keeping in mind the values of the attributes and environmental variables - we perform the resolution and transition procedures, and thus can determine the possible outputs. The other state and the subtree originating from it will be omitted.

Reducing the Number of States

It has already been mentioned that the number of states increases rapidly with the number of locations. Let us split the original graph into two subgraphs, and let us suppose that the number of states in one of the subgraphs is N, and in the other M. In this case the overall number of states in the original graph will be $N \cdot M$, whereas $N + M$ is sufficient to give the number of states in the decomposed graphs.

The connection between the subgraphs will be established by means of common locations. Decomposition will be carried out so that the number of attributes to be transferred and the common locations will be minimal.

Consideration of the Communication Network

When modelling, we must keep the real properties of the communication network in our mind. Our model considers the exchange of information between the communicating entities at the letter level, and these letters can be:

-- lost,
-- duplicated, and
-- possibly not in sequence.

These failures will occur in the communication network (primarily in the case of a datagram service).

This behavior can be modelled by means of the Nutt primitives: for example, the first characteristics will be realized by the absorber macrograph; the second and third by a priority-out queue.

CONCLUSIONS AND FUTURE WORK

In this paper, techniques for the modelling and verification of a network protocol were presented. The modelling method was proved to be primarily suitable for an end-to-end transport protocol but could also, hopefully, be suitable for other protocols (e.g., the virtual terminal protocol family). The verification and evaluation technique permits the deadlock and desynchronization characteristics of the protocol to be determined, but not as yet in a general form. Our future efforts will be focused on the development of a more generalized method for other classes of protocols and other properties.

ACKNOWLEDGEMENTS

The author wishes to express his thanks to Mrs. Toth whose theoretical work contributed largely to the elaboration of the protocol evaluation method, and also to his colleagues for their helpful comments, with special mention being given to Dr. Farkas for his valuable advice.

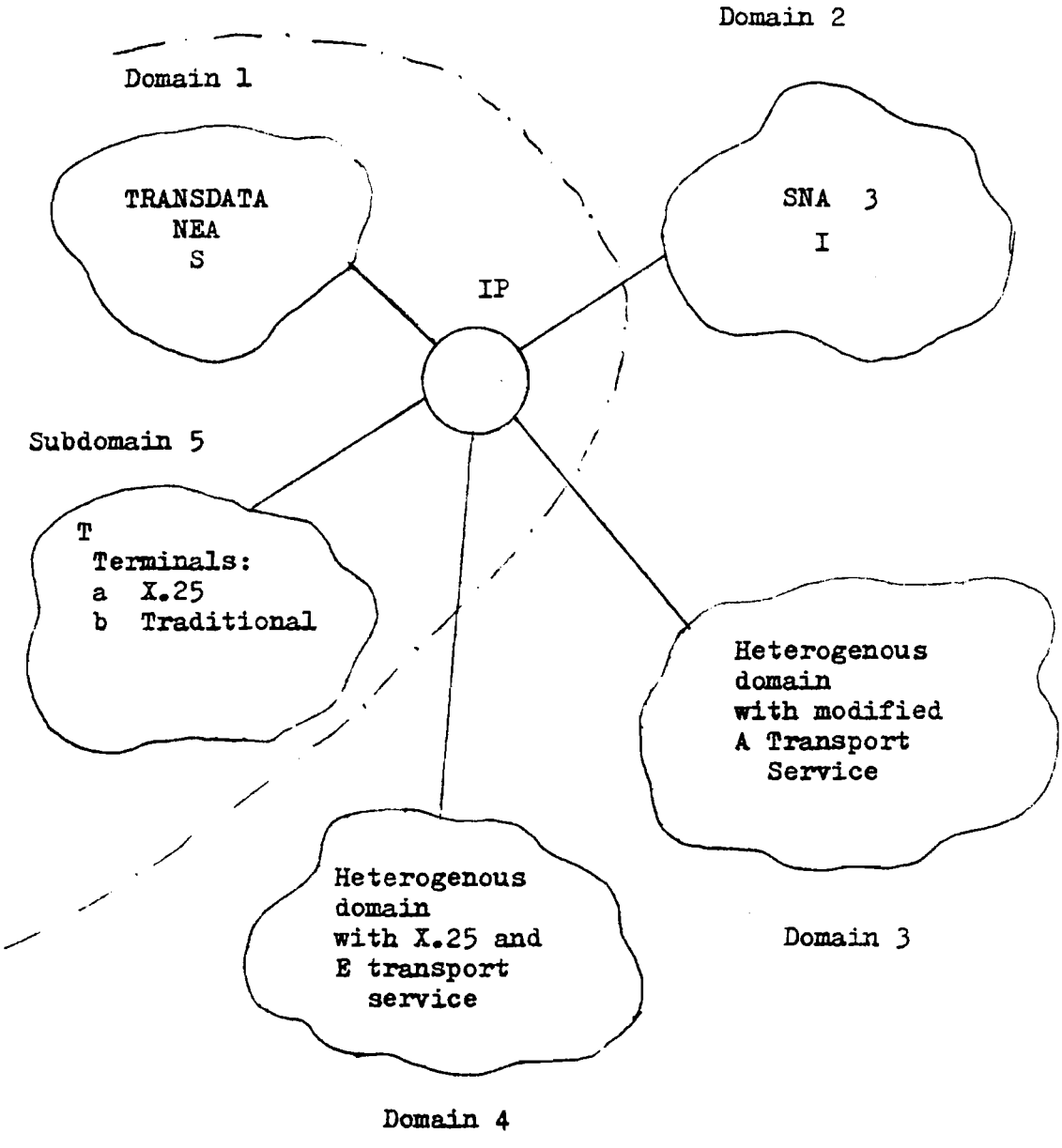


Figure 1. Domains constituting the network.

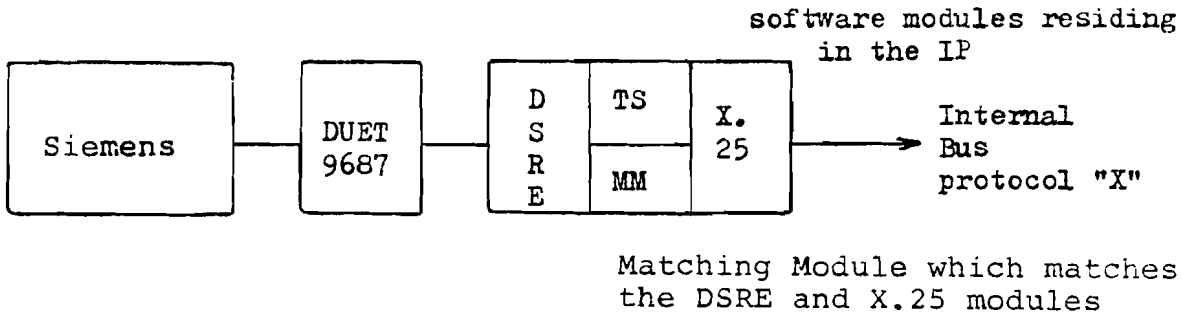


Figure 2. The Siemens connecting module.

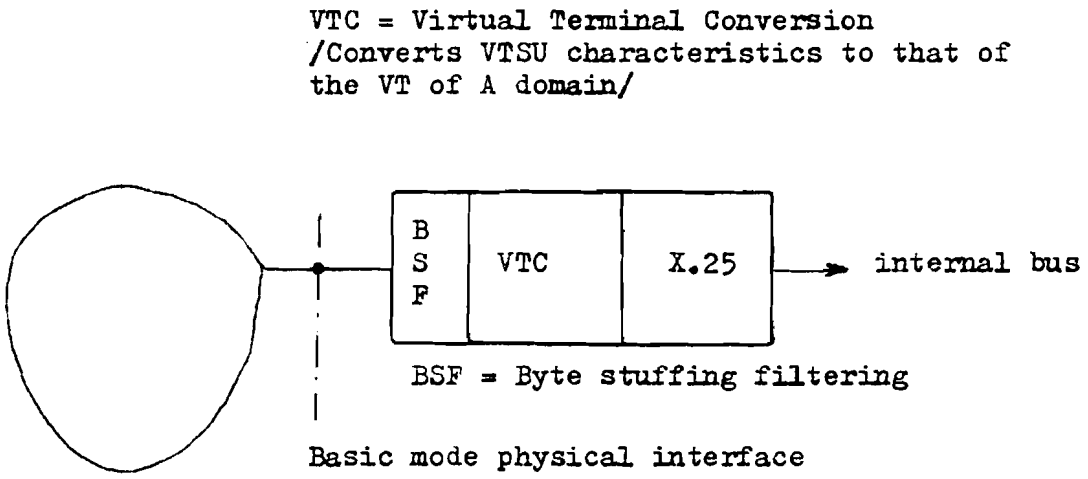


Figure 3. Connecting module of domain A.

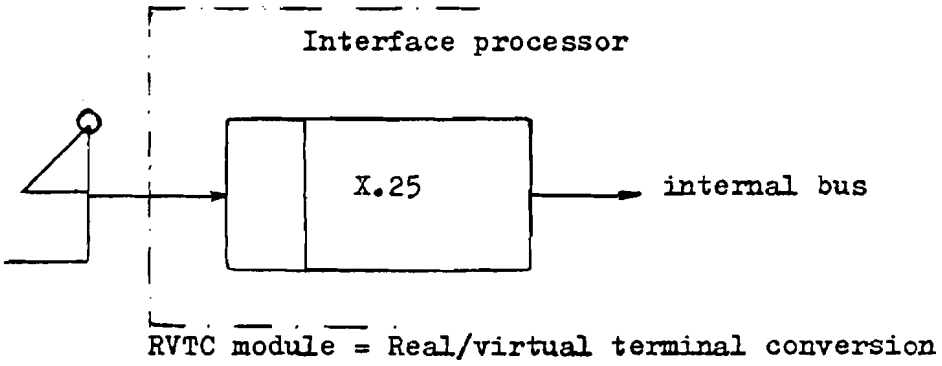


Figure 4. Connection of synchronous terminals.

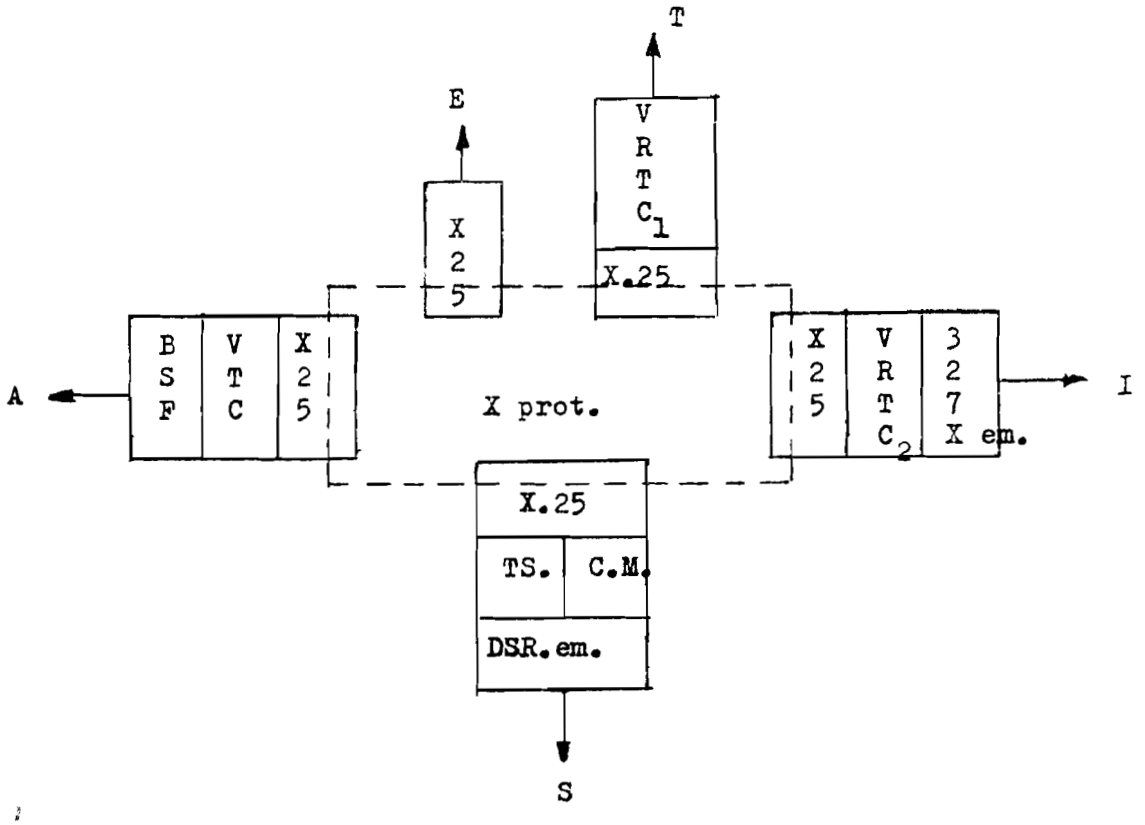


Figure 5. Software structure of the interface processor.

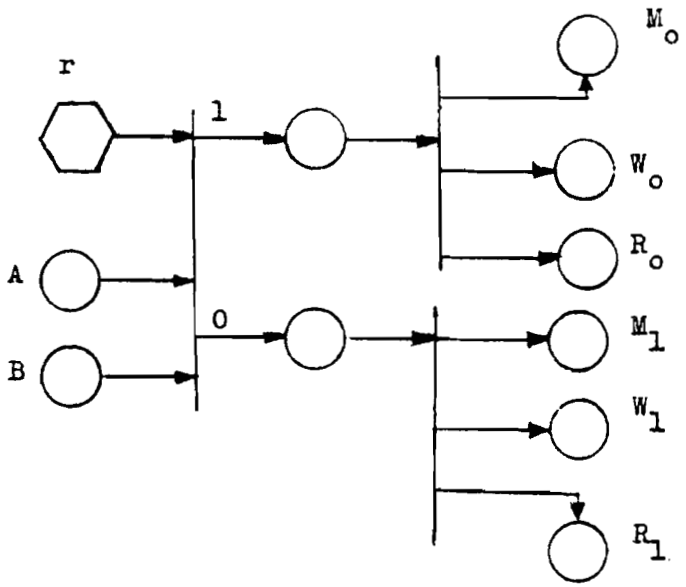


Figure 6. Danthine's primitive.

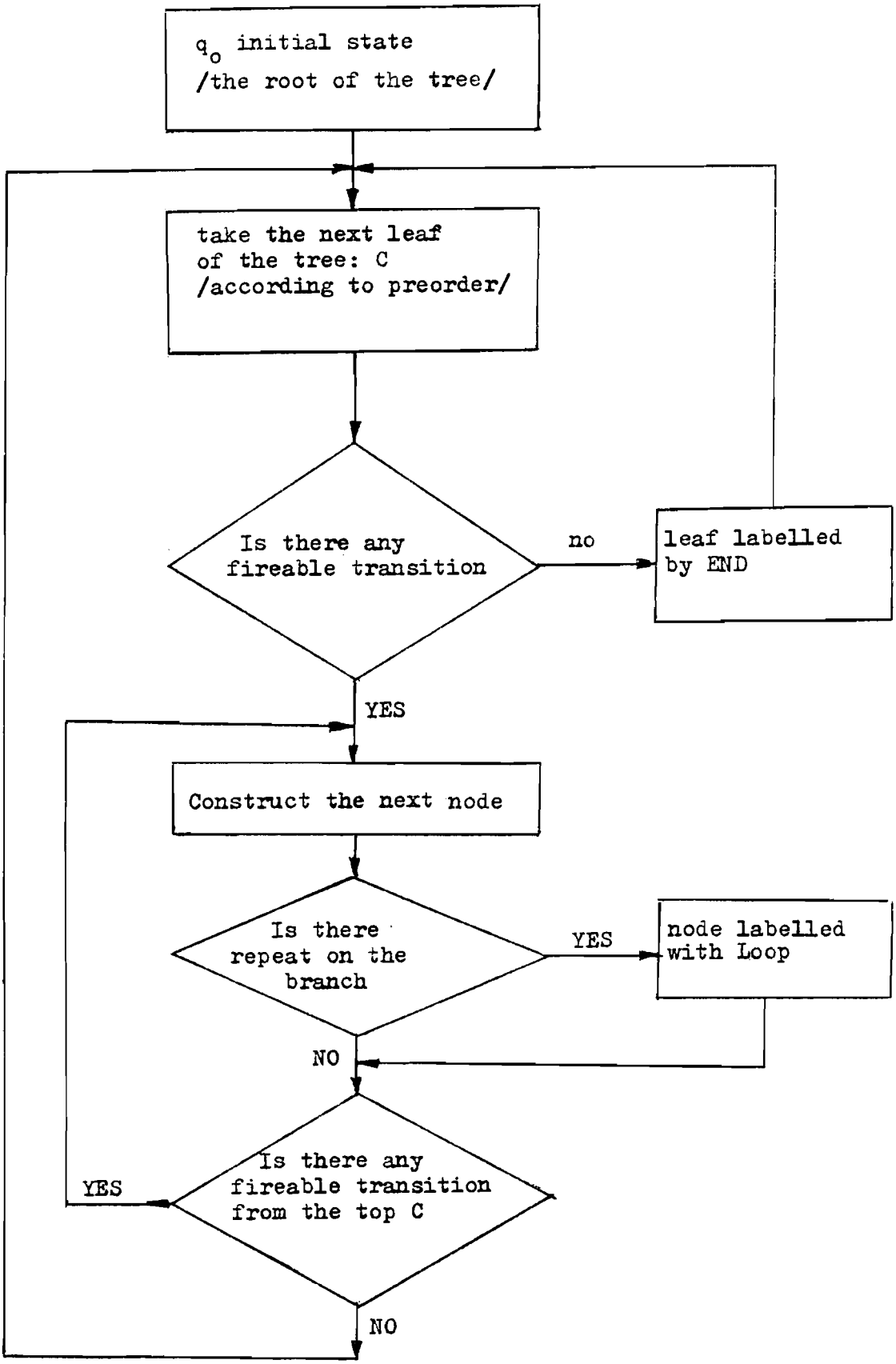


Figure 7. A sketched scheme of the reachability tree construction.

REFERENCES

[1] Nutt, G.J. (1972) The Formulation and Application of Evaluation Nets. National Information Service PB-214858.

[2] Danthine, A., and J. Bremer (1975) Définition, Réprésentation et Simulation de Protocols dans un Contexte Réseaux. AIM, Liege, Miniordinateurs.

[3] Zimmermann, H. (1975) The CYCLADES End-to-End Protocol. Pages 7-21 - 7-26, Proceedings of the Fourth Data Communication Symposium, Quebec, Canada, IEEE 751001-7 DATA.

[4] Postel, J.B. (1974) A Graph Model Analysis of a Computer Communication Protocol. UCLA ENG 7410 ARPA Contract No. DAHC 15-69-C-0285, January.

[5] Nutt, G.J. (1973) Some Evaluation Net Macro Structures. Computer Science No. 73-01-07. Washington University.

[6] Sunshine, C.A. (1976) Survey of Communication Protocol Verification Techniques. Pages 24 - 26, Proceedings of the Computer Networks Symposium, NBS, Gaithersburg, Maryland, November.

[7] Margitics, I. (1978) Final Report for Scholarship on the CYCLADES Computer Network. Doc. CYCLADES GAL 530, September.

FINAL LIST OF PARTICIPANTS

ING. G. ANDREONI
CREI - Centro Rete Europea di
Informatica
P.za Leonardo da Vinci 7
Milan
Italy

DR. J. A. BARCHANSKI
Institute of Engineering Cybernetics
Technical University of Wroclaw
ul. Wybrzeze Wyspianskiego 27
Wroclaw
Poland

DR. M. BAZEWICZ
Institute of Engineering Cybernetics
ul. Wybrzeze Wyspianskiego 27
Wroclaw
Poland

ING. A. BELLONI
CREI - Centro Rete Europea di
Informatica
P.za Leonardo da Vinci 7
Milan
Italy

ING. M. BOZZETTI
Ing. C. Olivetti e C., SpA - Centro Studi
Via G. Jervis 77
10015 Ivrea (To)
Italy

DR. F. CANESCHI
CNUCE - Istituto del Consiglio Nazionale
delle Ricerche
36 Via Santa Maria
I-56100 Pisa
Italy

PROF. ING. S. CASALE
Istituto Elettrotecnico della Faculta'
di Ingegneria
Universita' di Catania
V.le Andrea Doria 6
Catania
Italy

MR. MICHEL DEPARIS
EIN - European Informatics Network
c/o National Physical Laboratory
Teddington, Middlesex TW11 OLW
England
United Kingdom

PROF. ING. A. FARO
Istituto Elettrotecnico della
Faculta' di Ingegneria
Universita' di Catania
V.le Andrea Doria 6
Catania
Italy

MR. G. KACIN
EIN - European Informatics Network
c/o National Physical Laboratory
Teddington, Middlesex TW11 OLW
England
United Kingdom

DR. J. KOCSIS
Computer and Automation Institute of the
Hungarian Academy of Sciences
XIII. Victor Hugo 18 - 22
1132 Budapest
Hungary

DR. I. MARGITICS
Coordination Institute for Computer
Technique
Martinelli ter. 8
H-1052 Budapest
Hungary

ING. G. SCOLLO
Istituto Elettrotecnico della
Faculta' di Ingegneria
Universita' di Catania
V.le Andrea Doria 6
Catania
Italy

DR. P. ZUPA
CSATA - Centro Studi Applicati sulle
Tecnologie Avanzate
Via Amendola 173
70126 Bari
Italy

A. BUTRIMENKO

A. LABADI

A. PETRENKO

I. SEBESTYEN

U. SICHRA

M. ZIAK

IIASA
Schlossplatz 1
2361 Laxenburg
Austria