western union

Technical Review

MAGAZINE
WINTER 1968

INDEX

■ The purpose of the *Technical Review* is to present technological advances and their applications to communications.

■ The *Technical Review* is published by the Planning and Engineering Operation for management, supervisory and technical personnel in Western Union. It is issued quarterly in spring, summer, autumn and winter.

■ New Subscriptions — $10.00 per year including postage (4 issues)

Single Copies — $5.00 each

Back Issues available — 5.00 per copy (minimum order 2 copies)

■ Address all communications to:

Editor
Technical Review
Western Union
Mahwah, New Jersey 07430

WESTERN UNION
PLANNING & ENGINEERING OPERATION
82 McKEE DRIVE
MAHWAH, N. J. 07430

NEW YORK NEW YORK

FIRST CLASS MAIL
U. S. POSTAGE
PAID 30 CENTS
PERMIT NO. 697

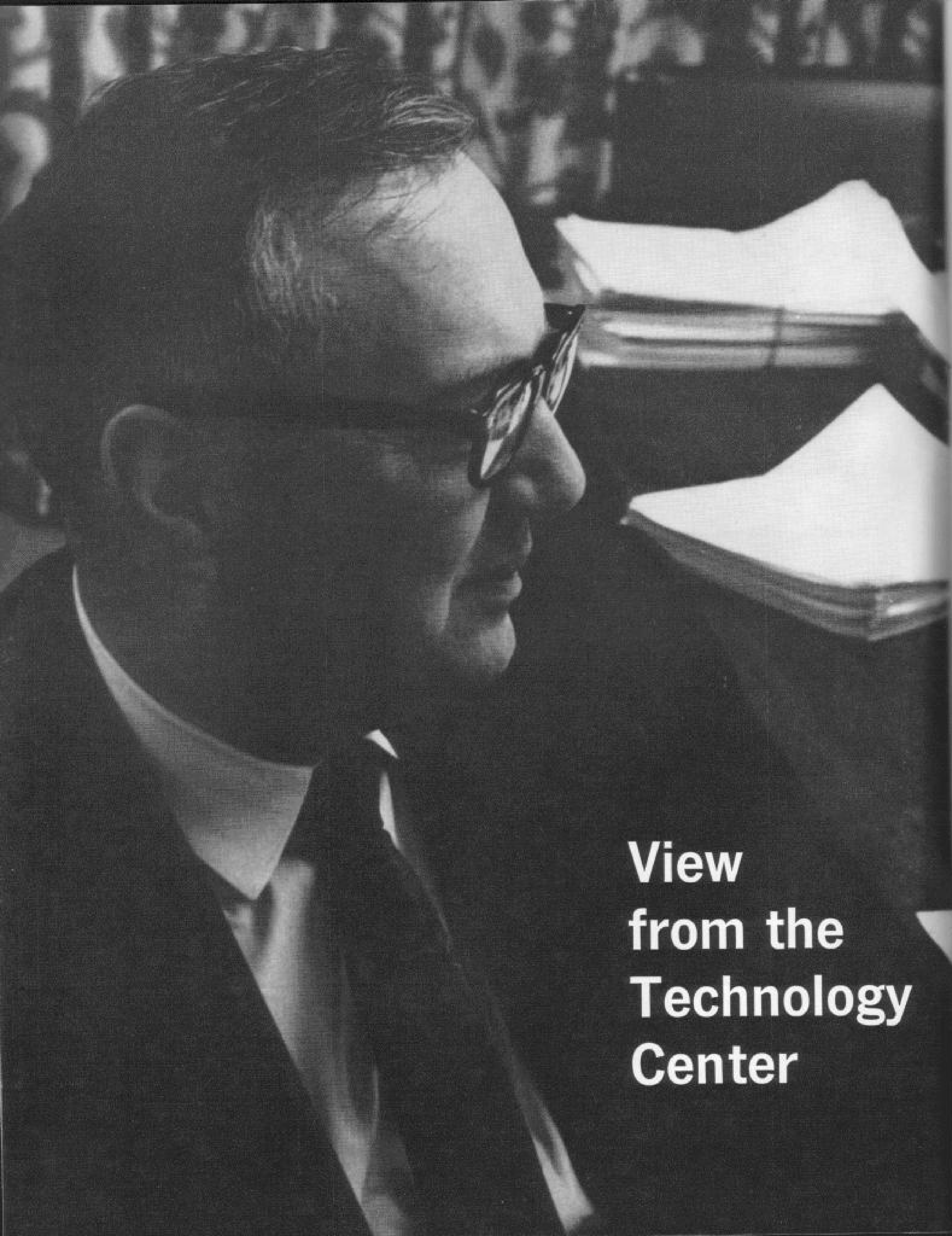
FIRST CLASS MAIL

James H. Haynes
1809 W. El Caminito
Phoenix, Ariz. 85021

western union
Technical Review
MAGAZINE
WINTER 1968

Cover: Western Union's new Technology Center at Mahwah, N. J. where systems and software check out is made and the simulation of second stage operation is implemented.

Special Credit: J. J. Luca, photo, Page 2; Lou Altieri, graphics



View from the Technology Center

By R. W. Hodgers, Jr., Vice President
Planning & Engineering Operation

It has been just one year since Western Union began the physical consolidation of the manpower and resources devoted to new services development and facilities modernization at the Western Union Technology Center in Mahwah, New Jersey. The Center, which is linked to the nationwide radio beam communications network by the tower pictured on the cover of this issue of TECHNICAL REVIEW, today houses several hundred specialized technical and professional support personnel.

Visitors who tour this new Company facility—groups ranging from Wall Street securities analysts to representatives of communications industries in countries as distant as Australia and Japan—express genuine appreciation for the opportunity to see a side of Western Union which is not familiar to many of our customers and friends.

I think the beginning of our second year of operation at the Technology Center is an appropriate time to describe briefly for the readers of TECHNICAL REVIEW the program and planning efforts underway here.

I believe that every reader of this publication is aware that the Company has been engaged for some time now in modernizing and expanding its facilities, and planning for new services, to meet the growing requirements for message and data communications. The high priority being given to this effort is based on the simple fact of the substantial opportunity represented by our industry's growth.

Studies of the volume of traffic handled over common carrier facilities indicate that there has been approximately a 500 per cent increase in number of words carried during the 12-year period between 1955 and 1967. Forecasts indicate that the industry will match that 500 per cent growth by 1972. The challenge we face in this industry is to maintain our current offerings, with improved speed and quality of service, while we develop the resources—both in increased facility capacity and in new services—to meet these exploding requirements.

Our traditional business at Western Union has been the information transportation business. We have been the moving company of the information industry, with the key characteristic being our acceptance of the total responsibility for a customer message, from the time it is received into our system until final delivery. We also have responsibility for the development, maintenance and operation of whatever facilities are needed to do this job.

With the introduction of Telex, we began to provide an information transmission service, wherein our role is to provide a path over which the customer sends his own message.

The commercial systems and government computerized communications systems, we have been providing for several years, give the customer the ability to handle both transportation and transmission functions.

That was where we stood before the Company's modernization programs were initiated. Since then, the Phase I Modernization Program has had several results: we have added Hot/Line and Broadband to our transmission services; we have designed and installed commercial computer information transportation/transmission systems and, most recently, Western Union has begun to offer computerized, shared services, the new breed of information transportation services.

There are three types of shared services in the Phase I Modernization Program: Telex Computer Communications Service (TCCS), Securities Industry Communications (SICOM), and INFO-COM™ for the general purpose user.

On January 1 of this year, the TCCS service was cut over. It is being expanded as rapidly as customers can be trained. SICOM will go into operation shortly. INFO-COM 100, with a 1200-terminal capacity—half of which already has been reserved—is now scheduled for cutover after mid-year.

In addition, using the TCCS capability, we have been able to introduce experimentally the sending of Public Message book messages directly into the computer, bypassing the input reperforator, and then automatically routing, by city and state address, through the output reperforator to the proper public office for final delivery.

™ Western Union Trade Mark

These initial new services all fall within the information transportation and transmission categories. They are coming on stream as the result of the first phase in the implementation of our total "Integrated Message/ Data System" concept.

In Phase II of the Modernization Program, the main thrust is modernization of the Public Message System, as well as expansion of INFO-COM into higher speed data services and automation of our internal data processing —particularly that associated with message billing and accounting. The Phase II planning also will take into account the outcome of our negotiations for acquisition of TWX.

The objective of Public Message System modernization in Phase II is to improve service while effecting cost savings. Computerization of our facility will allow us to attack PMS costs—those associated with cross office handling, reperforator operation and billing and accounting. In addition, the computerized network will give us a better basis for attacking the largest area of cost—terminal handling.

Establishing a computerized network, however, is only part of the PMS improvement effort. I want to mention two service improvement and cost reduction projects. First is improved telephone answering through centralization, already underway by the Public Office Operation. In the New England area, call diverters are now being used to automatically switch phone traffic from local offices—when these offices are busy or closed—to central office. Tie line centralization also is planned to permit operation of teleprinters and facsimile from greater distances at lower transmission costs. Several other service improvement projects will be carried out concurrently with the computer segment of the modernization effort.

As we progress through the various stages of our program to implement the "Integrated Message/Data System" concept, we are moving closer and closer to our eventual goal. This goal is to have total integration of Western Union systems and services, permitting our customers—whether in a public office, a Telex station, INFO-COM subscriber site or private terminal —to be able to communicate with each other and derive a useful service under control of the common, flexible facility we provide for them.

Modeling and Simulating the

To Simulate or Not???

The classical method of solving a scientific problem is to express its essential characteristics in terms of formal equations or in terms of a controlled laboratory experiment. This procedure is frequently referred to as constructing a "model" of the real problem.

Many problems can be formulated and solved entirely in terms of mathematical equations; but frequently complex system problems cannot be solved in this way. Consequently, they must be solved by an experimental method. This laboratory experimental method is often impossible or prohibitively expensive with large, complex systems. Another method, now popular in large systems, has been found to overcome some of these difficulties; it involves the development of a computer program which imitates the system under study in a simplified way. By this method called, "Simulation," system experiments can be carried out on the "model" represented by the computer program rather than on the system itself. It provides a method of testing, interpreting and evaluating the solutions to system problems for study by the analyst or designer.

GPSS

To facilitate the development of computer programs for simulation, many programming languages have been developed that are general purpose in application but contain many special features convenient to "simulation" work. An example of one of the most important and widely used of these "programming languages or systems" is the General Purpose Systems Simulator, GPSS. It was developed by Geoffrey Gordon of IBM and is particularly useful in preparing programs that simulate traffic-handling systems. The system to be simulated is generally broken down in terms of special "block diagrams." The user must be familiar with the rules by which these block diagrams are to be constructed. The output of the GPSS provides information on:

1. the volume of traffic flowing through sections of the system,
2. the distribution of transit times for the traffic flowing through sections of the system,
3. the average utilization of elements in the system,
4. the maximum and average queue lengths at selected points in the system.

Inputs to the Model

Various statistical and probability techniques may be introduced into the "model," many levels of priority may be assigned to selected units of traffic and complex logical decisions may be made during the simulation. The interdependence of the variables in the system, such as: queue lengths, input rates, and processing time can be simulated.

Simulation Languages

Some of the other languages developed for computer simulation are: SIMSCRIPT, SIMPAC, SIMULA, CL-1, GASP and DYNAMO. A description and comparison of the features and application of these languages is beyond the scope of this article. However, they may be found in current literature.

Criteria for Decision

A decision between using an analytical model and simulation requires a detailed understanding of the nature and characteristics of the complex problem to be solved. However, some general considerations should be followed before making such a decision.

If the problem is amenable to mathematical analysis and a relatively simple analytical model can be formulated, results can be obtained in much less time and at lower cost than with simulation. On the other hand, if too much sim-

ISCS-Phase I Processing Center

plification is required to "make the problem fit the model" the results will not be sufficiently accurate to provide useful information.

Simulation is not a panacea but rather a last resort. It is to be used for problems where no satisfactory mathematical solution can be formulated. This may occur because the problem is logically complex, non-linear, discrete, contains probabilistic elements or too many variables. In such a case, simulation can provide a model that more closely resembles the actual system, can include analytic submodels, provides flexibility of detail and results in the form of a time history with variable time increments. On the other hand, it does not provide "optimal" solutions, except by extensive parametric computer runs. It is costly in terms of computer and man time and requires highly skilled analysts and programmers. In addition, it generally requires voluminous input data and results are frequently difficult to verify.

While the use of simulation techniques has risen dramatically over the past five years, a word of caution on this trend is in order. Although simulation does, in fact, significantly assist in the solution of many problems, its arbitrary use, rather than a balanced use of analytic and simulation techniques, is to be avoided. ■

Mr. B. Weitzer, Director of Systems Analysis of the Western Union Systems Projects Operation, is responsible for the design and performance analysis of ISCS Systems. His 17 yrs. diversified experience in computer systems has made him an authority in this field.

B. Weitzer

By F. R. Hanvey, Jr.

The following article describes the computer simulation of the Western Union ISCS Phase I Processing Center. The complexity of the PC was the major reason for electing the simulation approach for that study. It would have been possible to analyze only isolated parts of the system, or a very simplified model of it, had it been studied analytically. The simulation model, on the other hand, is a more accurate and comprehensive image of the PC. It provides a means of observing the entire system in operation, including the interface relations among separate parts of the PC.

Modernization Program

Western Union has embarked on a modernization program to redesign its plant facilities for expanding record communication services. This modernization is designed to improve the speed, quality and interchange capabilities of the present communication system facility. This program will ultimately provide an Integrated Message/Data System (IMDS) capable of full message switching interchange; the necessary new plant facility will be a base for extensions of existing services, for a new family of shared services, and will provide growth capability into totally new message/data service areas.

One of the first stages of this development of an integrated Message/Data System is called the ISCS-Phase I System, which consists of computer-based processing and communication centers providing nation-wide, rapid, automatic record message interchange on a store-and-forward basis for Telex-to-Telex and Telex-to-TWX and Telex-to-PMS subscribers. In addition, ISCS-Phase I provides an initial INFOCOM service, a standardized store-and-forward message switching service permitting subscribers to maintain an apparent private, computerized, message switch system, while using the Western Union communication plant for actual message transmission. The INFOCOM subscriber may relay information to the Telex, TWX and Public Message System through ISCS-Phase I.

The heart of the computing capability of the Phase I system is the UNIVAC 418 computer, which is utilized for both the Communication Center (CC); and the Processing Center (PC) functions. The 418 Communication Centers consist of UNIVAC 418 main frames with communication subsystems necessary to interface these with the communication lines. No peripherals are employed at the Communication Center for "on-line" message switching operations. The CC performs all of the repetitive communications functions associated with terminal servicing and line interplay and it also responds to message routing indicators to direct an input message to the proper PC for processing. The processing center (PC) is a UNIVAC 418 computer with a full line of peripheral devices, i.e. drums, tapes, etc. It performs all of the logical functions associated with the message, such as in-transit storage, routing, header validation, journaling, billing, etc. This article concerns itself with the modeling and the simulation of the ISCS-Phase I processing center.

Why Simulate?

One of the major requirements for orderly design and development of any complex real-time computer system is the ability to measure the design status and the performance of the system implied thereby. Additionally, one must be able to predict the impact of design decisions before those decisions are implemented in hardware or software. The typical techniques utilized to provide these required measurement and prediction capabilities include modeling, analysis and simulation. For many problems, the analytical approach may be found superior to the simulation approach, both in time required between problem statement and solution, and in terms of the insight provided by the analysis itself. This article will not attempt to compare simulation and analysis nor to defend simulation as a technique often superior to that of analysis. Instead, it will discuss simulation as a technique required in certain instances where analysis cannot be effectively employed for the entire system involved. This situation was found in the evaluation of the ISCS-Phase I PC performance characteristics.

Three conditions were found which prevented the analysis approach from being effectively employed to evaluate the entire PC performance. These were:

- Considerable complexity in the system model and the interrelationships implied thereby. This complexity is best exemplified by the fact that the Central Processing Unit (CPU) and the peripheral subsystem may both be utilized simultaneously in a time overlapping manner. As of this time, simulation is the only way to measure this overlap factor.

- Random characteristics of the traffic being presented to the processing center, both in terms of times of occurrence and traffic character—such as message length, percentage of header, multiple address factor, etc.

- The interdependence between system functions and status within the model, i.e. the waiting lines, or queues, which exist for one program are functionally related to, and may be entirely dependent upon, the time history of the queues existing for, and the processing accomplished by, another program.

The foregoing three characteristics induced us to utilize simulation as the major tool for performance evaluation and prediction in the Phase-I ISCS processing center. The potential ancillary benefits to this decision are discussed briefly below.

The model development required for simulation happens to be a useful tool for monitoring the system design progress and for displaying the impact of "system requirements" decisions. As the system design develops, so does the simulation model. The timing estimates on model elements, in many instances, can be viewed as "coding budgets" which the programmers may try to meet. In a healthy design environment, arguments will arise as to the merits of certain algorithms, file structures or program priorities. Simulation results of these modeled functions can be of great value in deciding between approaches. Thus, a major by-product of any system modeling exercise is that the detailed observations of the system, necessary to create the model, inevitably lead to a better understanding of the system itself, and usually to suggestions for improvement which would otherwise not become available. This has been the case in the Phase-I PC modeling and simulation reported upon in this article.

The Simulation Model

The starting point for a useful simulation model is a detailed flow diagram depicting all system functional operations, interaction paths and basic logic constraints. For a Processing Center as complicated as ISCS-I, such a diagram required a large wall for display and many volumes of explanatory material for understanding. Some insight into the complexity involved in the myriad PC operations may be read in another article in this issue of the TECHNICAL REVIEW dealing with system checkout and test aids (TELTRAN, et al) generated for the ISCS-I system. Suffice it to say that a PC functional flow chart was generated in the PC modeling process and that it was sufficiently complex that it is inappropriate to attempt an explication of it here.

As the modeling process continued, insight was gained into various aspects of system operation, such that certain model details could be omitted and others could be combined. The result was a PC simulation model compact enough to be handled via GPSS—a General Purpose System Simulation language developed by IBM and specifically tailored to traffic simulation. A simplified functional block diagram of the simulation model is shown in Figure 1. (Note that Fig. 1 is annotated with operationally oriented statements, plus some GPSS code oriented statements).

The GPSS code actually implemented for this model was developed by the Western Union Analytical Services a subsection of Systems Analysis. It turned out to be rather unusual in two interesting respects. First, the simulation of "n" minutes of traffic may be accomplished in significantly less than "n" minutes of IBM 7094 time. Most "complex" simulations require considerable more computer time to simulate a unit of real time. Secondly, the GPSS code was large and complex, as measured by nominal GPSS program sizes. This latter situation was discovered at the first Annual Conference on "Applications of Simulation Using GPSS" held in New York City on November 13-14, 1967 under the auspices of the IEEE Systems Science and Cybernetics Group. Some 500 GPSS users, devotees, and interested parties attended this conference. In one of the working sessions, a straw pole was taken on the sizes and complexities of the GPSS programs of which the members of the audience had any knowledge. In this straw pole the Western Union PC simulation was the third largest GPSS application identified. The fact that this GPSS code, which was both efficient and complex, was generated and checked out in an interval of approximately three months speaks as testimony to Western Union's capability in this area and the efficacy of the GPSS simulation approach.

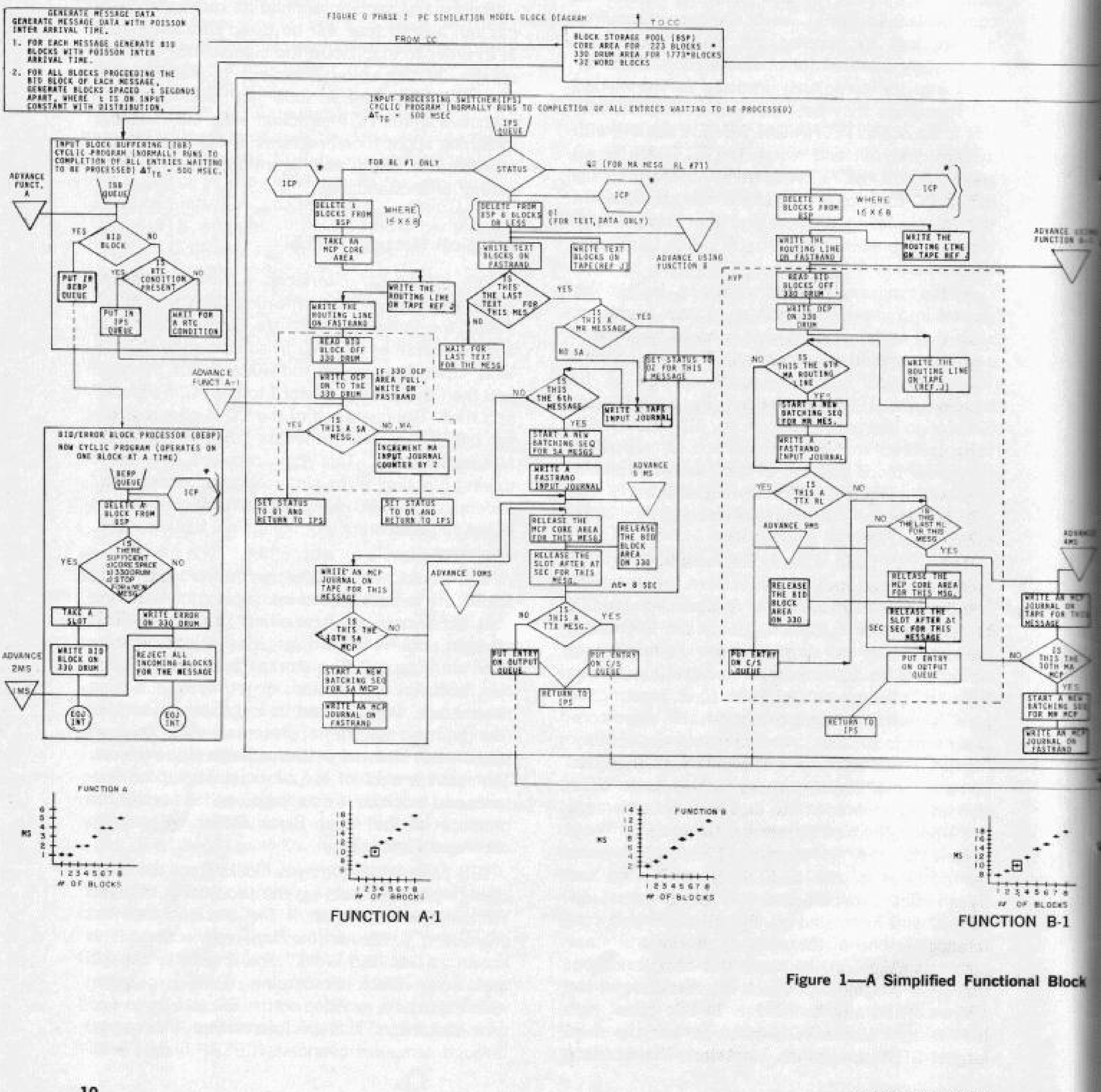
Before any specific simulation results are discussed, a brief tour will be taken through Figure 1, in order to emphasize the significance of these results. It should be noted, at this point, that Figure 1 does not include any details of the executive system's interaction with the message switching application programs. These interactions were omitted from the figure to make it more readily understandable.

A Typical Message Switch

The ISCS-I CC interfaces with the message input and output communication lines and multiplexes the individual message data streams into a form usable by the PC. In so doing the CC collects characters for an individual input message and then ships these forward to the PC, a "block" at a time. The operation of the CC is shown at the top left of Figure 1 in the box labelled "Generate Message Data." This box provides the basic driving function of the simulation, e.g. messages/sec input to the PC. Most of the significant results of the simulation are plotted against this msg/sec. argument.

The blocks which are delivered from the CC to the PC are initially placed into a common buffering pool of storage while awaiting processing. This storage pool is shown as the "Block Storage Pool" in Figure 1. The storage area set aside for this pool is one of the major system design parameters, since it must be large enough to hold (temporarily) all traffic presented to it; and as traffic rates rise, so do the storage requirements. The management of the block storage pool contents and block address allocations falls under the province of the Input Block Buffering program shown on Figure 1.

IBB periodically removes blocks from the block storage pool and sets up the processing required for the block involved. If the block is the first one for a particular message/connection, it is known as the "Bid Block" and it goes to the Bid and Error Block Processing (BEBP) program which acts as a monitor on the PC activity to prevent disastrous loss of information (messages) through computer overload. If BEBP finds the PC



FUNCTION A-1

FUNCTION B-1

Figure 1—A Simplified Functional Block

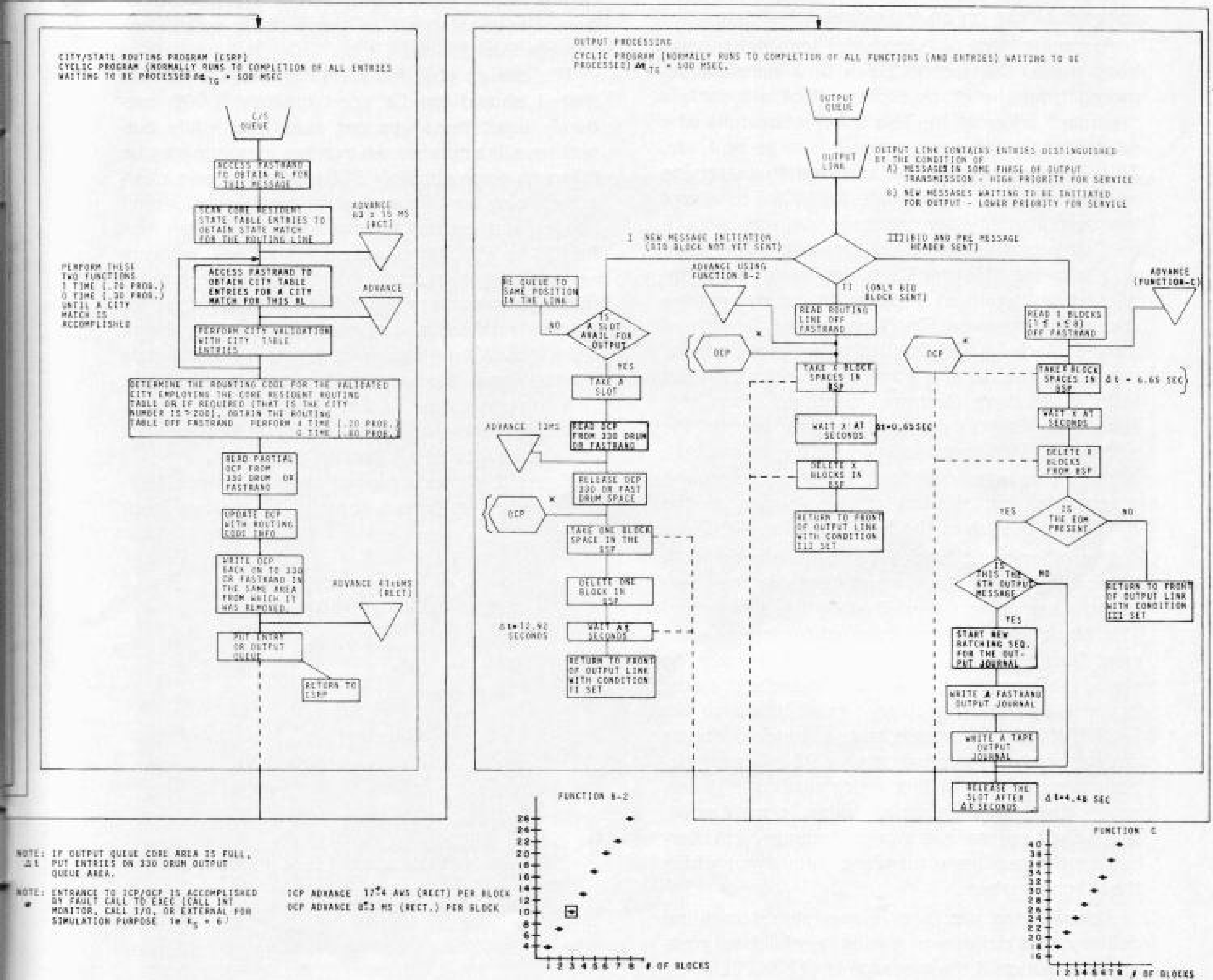


Diagram of ISCS—Phase I Simulation

too busy to accept new messages, it sends a message back to the CC directing that the CC disconnect the new message line represented by the Bid Block. Otherwise, BEBP directs the CC to connect the line and continue to send message blocks into the PC for processing. In any event, the Bid Block does not contain message oriented data—which will be found in the second and subsequent blocks on an individual connection.

Assuming that a successful connection has been made, the second block of a message removed from the Block Storage Pool will contain "Header" information. This information tells who originated the message, where it is to be sent, etc. This information must be analyzed to determine whether or not it is adequate for ISCS-I to accept the responsibility for message delivery with a negligible probability of lost messages. This analysis is called "Header Validation" and is accomplished in the Input Process Switcher program (the path shown on Fig. 1 as the left branch of the IPS block, i.e. IPS-00). If the message contains more than one delivery addressee for the common text, (a Multiple Address message), then this same analysis must be performed for each of the addresses. The bookkeeping involved in multiple address messages causes this analysis to require a separate (but similar) routine (shown as the right hand branch of the IPS block, i.e. IPS-02).

The other basic block type to be processed in IPS is the "Text" block, which contains the basic information being transmitted by the subscriber. This block need only be examined for format error conditions, and, if none are found, it may be stored while awaiting forwarding to the addressee. In the ISCS-I PC this storage takes place on the FASTRAND drum, a mass storage random access device, which has approximately a 92-milliseconds-per-access (either writing on or reading off) execution time. This numerical value is mentioned, primarily, because it is a critical number in determining the maximum msg/sec throughput which the PC can sustain.

Although the message is now stored awaiting delivery, that delivery may require additional routing information if the message is of the TELEX-to-PMS type. The processing to obtain this routing information is called "City/State Routing", a separate box for which is shown in Figure 1. The message delivery (from the PC) is accomplished by the "Output Processing" program. For this delivery, the message blocks are read from the FASTRAND drum and stored temporarily in the Block Storage Pool prior to going to the CC for delivery. This completes the basic loop of processing in the PC.

Block Storage Pool Sizing

The simulation provides a myriad of detailed results which were useful in the system evaluation. The most basic of these deals with Block Storage Pool Sizing.

The design goal for the ISCS-Phase I PC was that it should handle approximately 5,000 randomly input messages per peak hour while outputting a like number. An average message may be taken as approximately 300 alpha numeric characters long for this goal statement (including all header and control information). The block size for CC to PC intercommunications was set very early at 64 characters, 50 of which represent message characters and 14 of which are used for block identification and communications control. Thus, the PC might expect to see approximately 30,000 input blocks over the peak traffic hour. The basic question posed concerns the number of blocks that need to be set aside in the block storage pool to safely handle that load.

Figure 2 shows a plot of observed Block Storage Pool contents as a function of average input

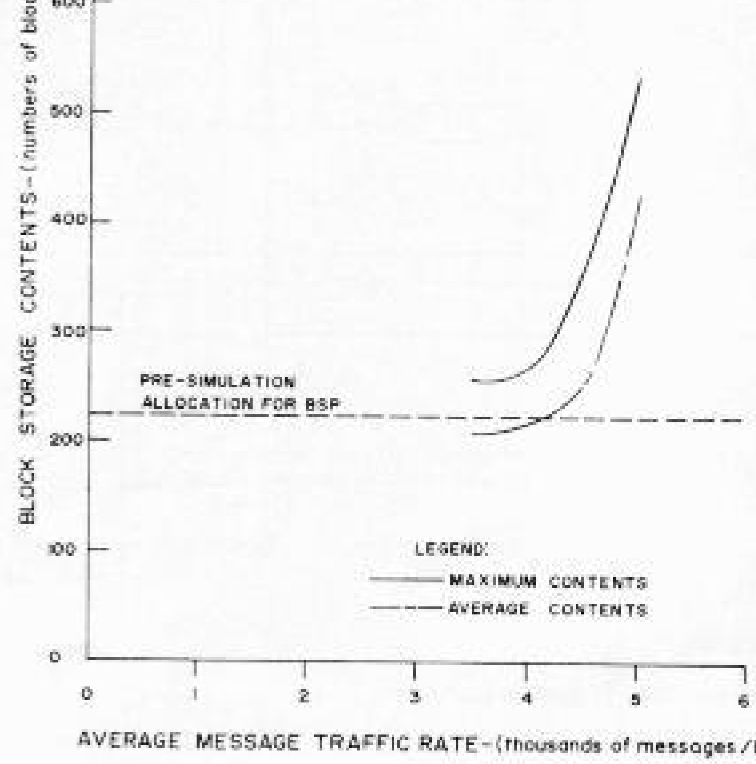


Figure 2—Plot of Observed Block Storage Pool vs. Average Input Traffic

traffic for the simulation runs. The blocks involve both input and output block buffering. Figure 2 points out that over 525 blocks must be set aside for this block buffering at the design goal traffic of 5000 messages per hour. This may be contrasted with the 223 blocks set aside for this purpose prior to the availability of the simulation results.

Also, it may be noted from the figure that, at the design goal, the rate of increase in BSP requirements for increasing traffic is quite steep. This is typical of the queues which exist in systems approaching maximum capacity. Thus, we conclude that the design goal can be achieved. Figure 2 induces us to look further into the specific element that is limiting the maximum traffic to approximately the design goal.

The foregoing question may be examined rather simply in the simulation, merely by changing individual timing elements and re-running the simulation. The resulting runs showed very clearly that the maximum PC capacity could be increased

Fig. 3—ISCS-Phase II System at the New Technology Center in Mahwah, N. J.

by a factor of approximately 1.5 if we could lower sufficiently the 92 millisecond average FASTRAND access time. Further examination showed why this is true.

As the PC programming is structured, the Input Process Switcher may be working on one, and **only one,** message block at a time. Also, if IPS makes an I/O call to FASTRAND, it suspends further processing until the I/O response is completed. Thus, we have input blocks waiting for service from IPS, and IPS is waiting for service from FASTRAND. Therefore, speeding up the average access to the drum would result in lower total waiting time, and, therefore, shorter waiting lines in the Block Storage Pool.

Armed with this information, we may recommend that future PCs, with higher design goals, be structured with more of a parallel multi-program environment, and that a large number of accesses to relatively slow peripherals be avoided. These suggestions are not particularly astounding in themselves, but they tend to carry more weight when the simulation justifies them with actual cases.

Epilogue and Prologue

The PC simulation proved to be an extremely valuable tool for evaluating the design of the IMDS-Phase I system. Specific recommendations forthcoming from that evaluation are now being implemented in the system. Even greater benefits are envisioned from simulation for the planned ISCS-Phase II system. This is partially true because of the increased magnitude of the message switching job, (86,000 msg/hr.-system), and partially true because we plan to obtain the benefits of sharing peripherals (drums, etc.) in a "Multi-processor" environment. The ISCS-Phase II system will employ high-speed 3rd generation UNIVAC 1108 central processors with dual channel peripheral access and with contention among the CPUs (Central Processing Units) for peripheral services. ISCS-Phase II computer system equipment is presently installed in the new Western Union Technology Center in Mahwah, N. J., shown in Figure 3. The added system complexity of ISCS-II may be handled in its entirety via the combination of simulation techniques and this computer equipment.

Acknowledgements

The author wishes to acknowledge the assistance of A. Spagnolo and A. Wadler who developed the model and analyzed the runs. He also wishes to give credit to H. Gabrieli who wrote the GPSS simulation and ran the program.

F. R. Hanvey, Jr., Manager of Systems Modeling and Evaluation in Systems Analysis and is responsible for applying the techniques of modeling and simulation to the evaluation of the ISCS—Phase I system and the design and performance of ISCS-II.

He joined Western Union in 1966 but previously had been with General Precision where he was concerned with inertial guidance systems. He served five years with the U.S. Air Force in the analysis and test design of inertial guidance systems.

Mr. Hanvey is a graduate of the United States Naval Academy class of 1957 and received his Master of Science degree from Massachusetts Institute of Technology in 1959.

Telex Computer Communications Service and Testing Techniques

by E. H. Primoff and F. C. Hilse

Part I

TCCS Hardware and Software

Part I—describes the hardware and software used in the TCCS programming system.
Part II—describes the testing techniques and the test tool TELTRAN which is covered in more detail.

The Telex Computer Communications Service, TCCS, previously known as ACTS, is a large, complex, computerized store-and-forward message switching system which connects several computer systems by means of high-speed data transmission lines. This service interfaces with the existing services: Telex, TWX, and PMS, all of which use different terminal equipment.

The TCCS programming system is a real-time system having the following programming characteristics:

Critical response time—real time

Input rate—medium

Number of input types—low

Priorities—mixed*

Reliability—fail soft

*Mixed priority means—messages are queued on a first-in first-out basis within a priority category (i.e. rush or normal). Also, message switching proceeds with a finite number of tasks, each with an implied priority, for a given message.

Characteristics of TCCS

TCCS enlarges the capacity of Western Union's message switching services. In addition, it enhances billing functions, automatic record keeping and message switching of our Telex customers. Telex users may send messages to other services such as TWX and PMS.

The Telex Computer Communications Service interfaces with the three following networks, at different speeds:

a) With the Telex network at 50 baud with 7.5 unit character frame in baudot code;
b) With the TWX network at 110 baud with an 11 unit character frame in 8 level code (ASCII); and
c) With the Public Message System at 75 words per minute, with 7.42 unit start-stop 5 level code.

In TCCS, data transfer between CCs is at 2400 baud on full duplex lines.

Internal control of TCCS by Western Union requires four types of supervisory positions at each computer site.

a) IIP
At the Input Intercept Position ASR 35 sets transmit at 110 baud with an 11 unit char-

acter frame in 8 level code (ASCII). This position is used for message retrievals, amended headers and normal input messages.

b) OIP

At the Output Intercept Position the same type of ASR sets as above in (a) are used. The position provides printouts of accepted message headers with correctable errors and printouts of retrieved messages.

c) SAP

At the Supervisory Acknowledgment Position, ASR 28 sets are used to receive at 100 words per minute in baudot code. This position receives acceptance or rejection messages on input disconnects.

d) CLP

At the Communication Log Printer, Kleinschmidt RO devices are used to receive at 272.7 words per minute (300 baud). This position is used for recording system status reports and error indications.

TCCS Hardware

The TCCS Message Switching System has 12 computers, located at New York, Chicago, San Francisco, and Atlanta. As shown in Figure 1, at New York, Chicago, and San Francisco there are 3 computers, 1 CC, 1 PC and 1 Standby. At New York, there is an additional remote CC, shown as NYC2. Atlanta has only 2 computers, 1 CC and 1 Standby. The Standby at New York, Chicago, and San Francisco may be used as a CC or a PC.

The CC computer is a Central Processing Unit (CPU) with up to 5 multiplexers for handling the communication lines. Thus, the CC serves as a message concentrator.

The Central Processing Unit is a UNIVAC 418 computer having 65K 18-bit words of core memory storage and up to 16 Input/Output channels. The CPU includes a day clock and delta clock, one active index register, 12 bit addressing, and a 2 microsecond storage cycle time.

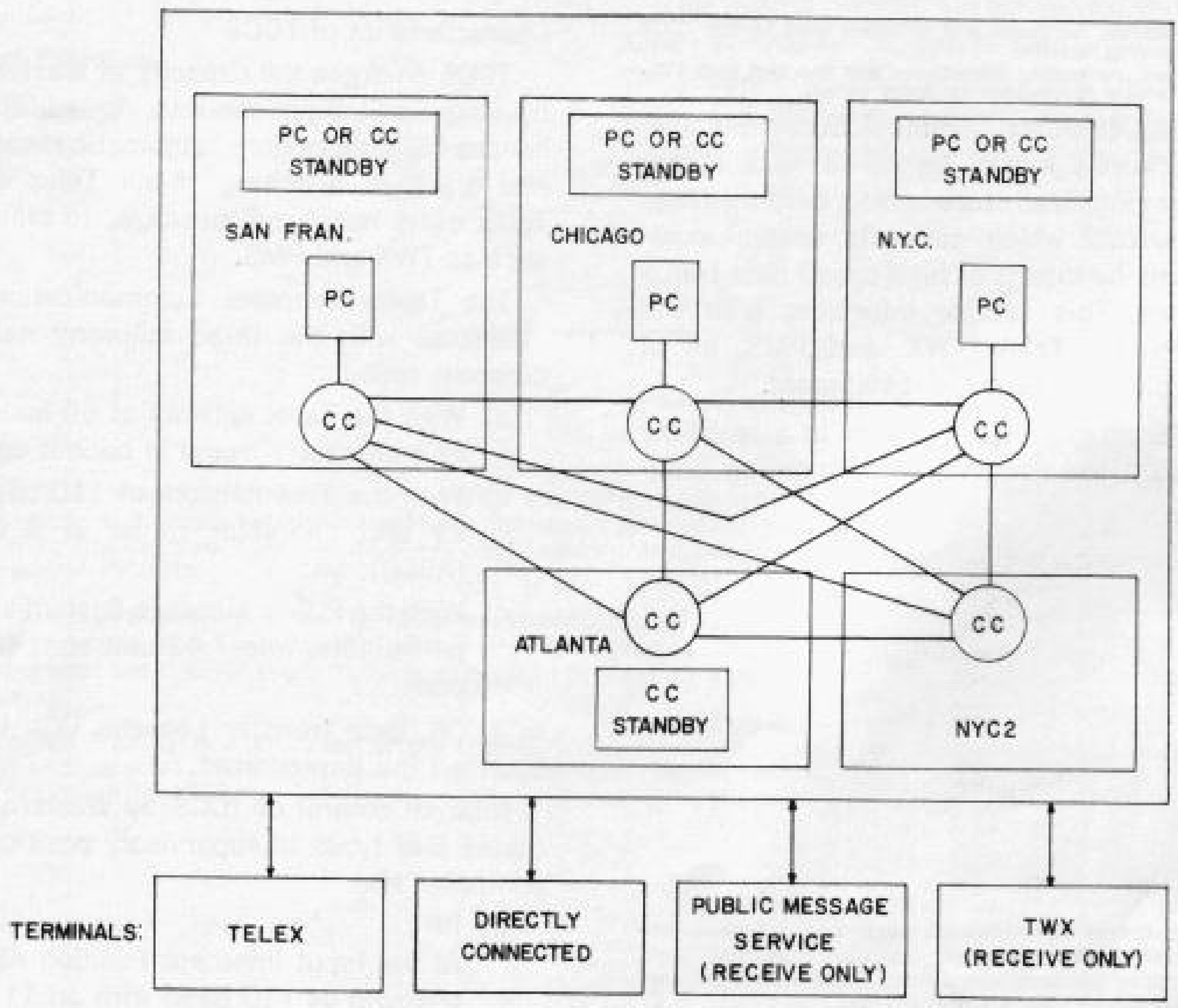


Fig. 1—TCCS Message Switching System

The PC computer comprises a CPU, six tapes of the VI C variety, a FASTRAND II and an FH 330 drum. 1004 subsystems were available for aid in program development and debugging.

All CCs are interconnected by full duplex 2400 baud lines. Each CC at New York, Chicago and San Francisco has a dedicated PC for Telex input traffic and a table of alternate routes in the event of line failure.

Character Codes

Various character codes are used in the TCCS. These are shown in Table I.

Table I

Code	Function
Baudot	Low speed—CLP, TLX, PMS, SAP
ASCII	Low speed—TWX, IIP, OIP
ASCII	High speed—CC to CC
XS-3	1004
Fieldata	418 console
Dense graphic binary	internal software control
Rotating Hamming	error detecting
Special internal BCD	Day Clock

Inputs/Outputs

The system inputs are Multiple Address, Single Address, or Multiple Message per connection. Multiple Message means more than one single address message in a given connection.

The system outputs are messages to the TELEX subscribers, to Public Message Service and TWX subscribers.

Additional outputs provide an historical record of message switching for on-line retrieval, off-line billing, and legal purposes. In order to do this, the TCCS maintains six journals plus statistical and recovery information.

a) The Input Journal is routing line oriented and provides a header record for every message accepted by the PC.

b) The Reference Journal provides a record of the header and text received by the PC.

c) The Error Journal provides a record of addressees of messages received at the PC but which have errors preventing routing and delivery.

d) The MCP Journal is message oriented and consists of the Message Control Packet, one for each single address, multiple address, and one for each multiple message on single connect.

e) The Output Journal identifies every message successfully transmitted by the PC. It is the principal source of billing information.

f) The Bid-Acknowledge Journal provides information about CC to PC Bids and PC Acknowledgements to the CC. This journal is connection oriented.

Message Formats for Input

A general description of Telex input single-address and multiple-address message formats implies processing breakpoints. Generally, fields within a logical line are delineated by spaces. Physical lines are terminated by the two characters (CR) (LF) or (LF) (CR).

A single address (SA) message has the following six basic segments

1) SOM line
2) Service line and primary routing for TLX, TWX
3) Secondary routing for TLX, TWX and primary routing for PMS
4) End of routing indicator (or beginning of text)
5) Text
6) End of Text indicator

In more detail, these six segments may be written as follows:

1) ZCZC OMN PR NL COLLECT CSDATE (CR) (LF)
2) SVC DIALCODE AB (CR) (LF) or SVC FREE (CR) (LF)
3) SR (CR) (LF)
4) BT (CR) (LF)
5) TEXT
6) NNNN

Lines 1 through 4 comprise the header.

Explanation

Line 1—ZCZC is the start of message sequence.

—OMN is an originator assigned message number. OMN is **optional** on one SA message per connect. It is required in multiple address (MA) messages and more than one SA message per connect.

—PR is an **optional** priority designator which may be R (rush) or N (normal) or C (rush for special system usage).

—NL (night letter) and collect are **optional** billing indicators.

—CSDATE is originators city and state with date and is used for PMS.

Line 2—SVC may be TLX, TWX, or PMS.
—DIALCODE and destination answerback (AB) are used for TLX and TWX. FREE is used for PMS and may be any user specified words (within certain format restrictions.)

Line 3—SR is secondary routing which is **optional** for TLX and TWX but required for PMS. In the latter case it contains the city and state of the destination.

Line 4—BT is the beginning of text sequence.

Line 6—NNNN is the end of message indicator.

For a multiple address message one must have a field containing the characters MA between the ZCZC and the OMN. An example of MA construction follows for N addresses:

1) First header
   Text
   NNNN
2) Second header
3) Third header
   .
   .
   .
4) Nth header
   NNNN

Some of the restrictions that are necessary for message construction are:

1) Header must not exceed 400 characters.
2) OMN's must be increasing numbers in successive headers, where required.
3) Text must not exceed 15000 characters.
4) NNNN, ZCZC, or BT may not appear inside the text.

How Messages Enter the System

The following steps indicate how a message is entered into the TCCS:

1. Push Connect Button on terminal.
2. Dial indicator light will light when connection to Telex is available.
3. Dial the computer system.
4. Computer pre-message ID is typed out.
5. Originator's AB is triggered and typed out.
6. Enter message from punched tape or type-in manually.
7. Acceptance (or rejection) message is typed out.

Software Subsystem

The three main subsystem functions at a computer node are:

a) CC processing
b) PC/CC interface
c) PC processing

CC Processing

The CC handles high speed traffic with all other computers in the system. It services input and output on terminal lines, checks input message character stream formats, converts characters when necessary for internal processing form, buffers input from low speed lines into blocks for high speed transmission and processes blocks for output to low speed lines.

Each CC queues up to 10 blocks for output to each of the other CC's. An acknowledgement waiting table containing Block Serial Number (BSN), block address, and time of transmission out is kept. Each 32 word block transmitted out must be acknowledged by the receiving CC. In addition, the BSN and Terminal Sequence Number (TSN) sequences must be maintained.

When blocks are not acknowledged, BSN's are not sequential, or TSN's are not sequential, the blocks are dropped and appropriate error action is taken.

Idle Blocks are sent to maintain high speed line activity and acknowledge prior blocks.

PC/CC Interface

At nodes with both CC and PC computers the CC is connected to the PC by a high speed core to core coupler. The PC/CC interplay function permits connectivity between input/output lines and the PC computer. Throughput control is supervised on both sides of the coupler. Information is passed to and from the PC in a standard communication block format used on high speed lines (see high speed processing). Blocks are used to transmit messages as well as intersystem control data. For example, there are error notification blocks and threshold status blocks.

PC Processing

The PC software is composed of two sets of programs.

1) One set contains the executive function and a group of routines known as the communication input and output control (CIOC) programs.
2) The other set constitutes the applications or message switching programs. These in turn are split into an input processing part and an output processing part.

The PC computer accepts buffered message blocks from the CC for message switching. Book

(MA) messages are partitioned by routing line (address) to create complete output messages for transmission. Messages are transformed (in the header, mainly) into a form acceptable to the TLX, TWX and PMS systems and buffered out of the PC at time intervals consistent with line speeds. For each message, journals are created by the PC for various off-line processes and recovery or retrieval when necessary.

The PC message switching programs operate under the control of a real time multiprogramming Western Union PC Executive, in a demand environment. The basic functions of the PC executive are Time Control, Input/Output Handling, Interrupt Processing, Interprogram Communication, and Process Schedule control. This system has a roadblock table (RBT), which may contain as many as 64 programs.

The roadblock table (RBT) is repeatedly scanned by the switcher (part of the executive) which enables execution of programs scheduled for operation. The latter is accomplished in one of two ways (not both for a specific program): 1) by means of a call which is accomplished by using a program fault instruction. The executive analyzes the call packet following the faulting location and sets a switch (in the appropriate slot of the RBT) for execution. When the switcher reaches that location during its scan of the RBT, the called program will be executed. The calling program will be left in a busy state unless an end of job indication is set in the call packet, 2) by means of the Time-To-Go (TTG) table maintained for the programs. Every 15 milliseconds (ms) there is a delta clock interrupt and, at that time, all time-to-go's are decremented by 15 ms. Whenever the TTG for a specific program goes nonpositive, the corresponding switch is set for execution in the RBT. This processor returns to the switcher when its time chores are done. When a program is interrupted, its parameters and place-to-re-enter are saved. Monitor and external function interrupts for the various I/O channels are handled by the executive interrupt processors. When a program calls for I/O service on a device, its request is entered on a queue assigned to that device. The I/O program called will, in turn, fault to the executive with a temporary "end-of-job" after initiating the I/O transfer enabling the switcher to execute other programs. I/O completion is indicated by an interrupt which is linked by channel number to the I/O call.

Only some of the programs on the RBT are initiated periodically. This is accomplished by maintaining a finite TTG value between executions. Other programs have effectively "infinite" TTGs.

The periodic programs include routines that control I/O block transmission across the coupler, error processing, city state (for PMS) routing, output queue processing, recovery information dumps and TELTRAN test data I/O.

First in the RBT (after the time control routine) are the I/O handlers which are followed by communication block I/O handling routines. The application (message switching programs) and test routines are next.

System Parameters

There are certain important system parameters that are used to control and trigger significant software functions.

The following list of some parameters is especially interesting for test functions:

1. CC trunk connect number (CCTCN).
2. Block sequence number (BSN).
3. Terminal sequence number (TSN).
4. Start of Message character (SOM).
5. Beginning of Text character (BT).
6. End of Message character (EOM).
7. End of Transmission character (EOT).
8. PC internal sequence number (PCISN).
9. Ready to Compute indicator (RTC).
10. Routing line sequential count (RLSC).
11. Error notification characters.
12. Time stamps.
13. Slot number.

The CCTCN establishes a unique identification for a connection of a trunk line to a CC. It is associated with each message on the connection throughout the CC and PC. The CCTCN contains the Julian date, the local CC identifier (to which the line is connected), the trunk or line number and a unique connection number assigned by the CC sequentially to connections.

The BSN is used for high speed sequential block transmission control relative to I/O transfer queues.

The TSN is used for high speed sequential block transmission control relative to an associated terminal connection.

The special 8 bit characters (SOM), (BT) and (EOM) are inserted in the message stream (at the CC) when the associated message character sequences occur on input. Character packing in the 32 word communication block is terminated (for BT and EOM) and a Ready-to-Compute bit is set in the control portion of the block. The latter is then queued up for transmission to the PC.

The EOT is a special character, similar to the above, used when transmission on an input line ceases legally.

A unique PCISN is assigned when each message **enters** the PC.

When the PC detects an RTC bit in an input block, a well defined message portion may be processed.

The RLSC is used for the control in the PC of multiple routing lines (such as for MA messages).

The Error Notification Characters are special 8 bit characters that appear in communication blocks to notify the other computer at a node of an error condition.

Time stamps are made in the CC at the occurrence of SOM, BT, and EOM. These are placed in two characters preceding the associated special characters. Time stamps are used in the computation of message billing by off-line programs.

The PC (in CIOC area) maintains a table of Slot Numbers for processing control associated with a line connection at the CC. This number is used as an associative pointer to data and control relating to a specific message portion (see RTC) on a unique connection.

In addition to the above parameters, the routing lines and originators' answerbacks (AB), are also important.

Message Transfer—Message Process Loop

TCCS message transfer is initiated when a TELEX subscriber terminal dials the computer requesting a connect. The TCCS transmits a premessage header and requests the subscriber's answerback.

While the subscriber terminal begins inputting characters, the CC transmits a Bid containing the CC trunk connect number to the appropriate PC. If a connect is received, the CC Low Speed Programs scan the input characters for the SOM indicator within 48 characters. If a connect is not received, the terminal is disconnected.

The SOM sequence is recorded (with the time and special SOM character) and the scan is continued for the BT condition. The characters are converted to ASCII, if not in ASCII, and packed 2 char/word.

The CC High Speed Programs transfer data in 32 word blocks (containing $\leq$ 50 characters) to the co-located PC over the ICC (inter-computer coupler) or to the appropriate PC via another CC.

Detection of the BT indicator, EOM indicator, and any following indicators will result in a RTC (Ready to Compute) Flag being set in the input stream (actually, in the control portion of the 32 word block).

The message is passed via communication Input/Output control programs in the PC to the Input Application Programs which perform the following functions:

a) validate the routing information.
b) record the routing and internal control information in the MCP (one for each message) and Output Control Packet (OCP) (one for each routing line).
c) Create journals.
d) Transmit an acceptance or rejection message to the subscriber terminal.
e) Enter reference to message on an output queue.

Periodically the Output Application programs scan a series of output queues and build a bid block for each output message. The CIOC programs transmit the Bid to the co-located CC. If the Bid is for a device associated with this co-located CC, a connection is established over the multiplexor lines. When a connection is established to the device, a connect block containing CC output trunk number and CC message sequence number is sent to the PC. The CC High Speed Programs transmit the Bid to another CC, if the bid specifies another CC.

The Output Application Programs record connect block information, read text from the FASTRAND, and give the text to CIOC. Assignment of BSN & TSN, and data transmission timing is done by CIOC.

When the data is delivered to the destination device, an acknowledgement block is sent to the PC.

Receipt of the acknowledgement by the PC causes an output journal entry, thus verifying that the message has been delivered. The message is retransmitted later as a suspected duplicate, if the acknowledgment is not received from the CC.

Part II

Testing and Debugging the TCCS via MSTT and TELTRAN

The two devices used successfully to test the TCCS-Phase I were the MSTT and TELTRAN. The Message Switching Test Tool is a tool which includes an environment remote terminal simulator. TELTRAN is a language and processor developed to test the TCCS. It prepares the test data in a form which can readily be interfaced with the MSTT.

The first stage of PC testing was debugging, prior to getting the TCCS system to cycle. After that, testing was concerned with monitoring and verifying the PC message processing loop (MPL). MPL tests were successively run through the stages of single thread, multi-thread, medium load and high load testing. For all of these tests normal processing and error processing were both exercised. During the runs, various test points such as important system parameters, message control packets (MCPs), output control packets (OCPs) and journal packets were examined. In simulator runs using TELTRAN input and the environment simulator, SAP, OIP and CLP output were recorded on the CC console typewriter.

One area in which the test simulator proved to be of great use was the introduction of simultaneous inputs in quantity. Some intricate timing problems were solved here.

It is characteristic of real time message switching systems that high loads tend to uncover problems not noticed during low load activity. Here the MSTT was significant.

The Message Switching Test Tool, MSTT, is used to test and exercise the PC from the CC side of the TCCS (i.e. across the coupler hook-up). The PC is more amenable to simulated testing than the CC, because the PC is farther removed from communication line interaction. In a sense the PC is not considered as "real time" as the CC by definition of the CC function.

The MSTT provided valuable timing data relevant to PC throughput capabilities. One advantage of the MSTT is that "one person" can exercise the PC system by simulating many simultaneous different input messages on different lines. All he needs is the CC-PC hardware configuration, PC software and the MSTT with TELTRAN derived input.

A library of Test Messages for the purpose of thorough system validation was generated. Some of these messages can be used for timing and throughput studies. Another set of messages was created for off-line billing tests, with different originators and different addressees. SOM line variations, text size variations, and all grades of service were included in the test.

One type of test run, Real Time Recovery, which was of great significance, interfaced very nicely with the full real time system. The PC, under control of the MSTT, read in TELTRAN test data with the output processing part of the MPL shut off. The resulting journal tapes were saved, and real time system **recovery** from tapes was successfully run with output turned on, thus providing significant recovery program tests.

The Phase I testing required the development of three basic areas.

a) MSTT
   1. PC test programs
   2. CC simulator (remote terminal simulator)

b) TELTRAN compiler

c) Post mortem analysis and edit programs.

MSTT

A general description of the operation of MSTT follows. In Figure 2 a set of snaps and dumps was incorporated into the message switching programs of the PC. Many different types of snaps are available. Some of these snaps were located

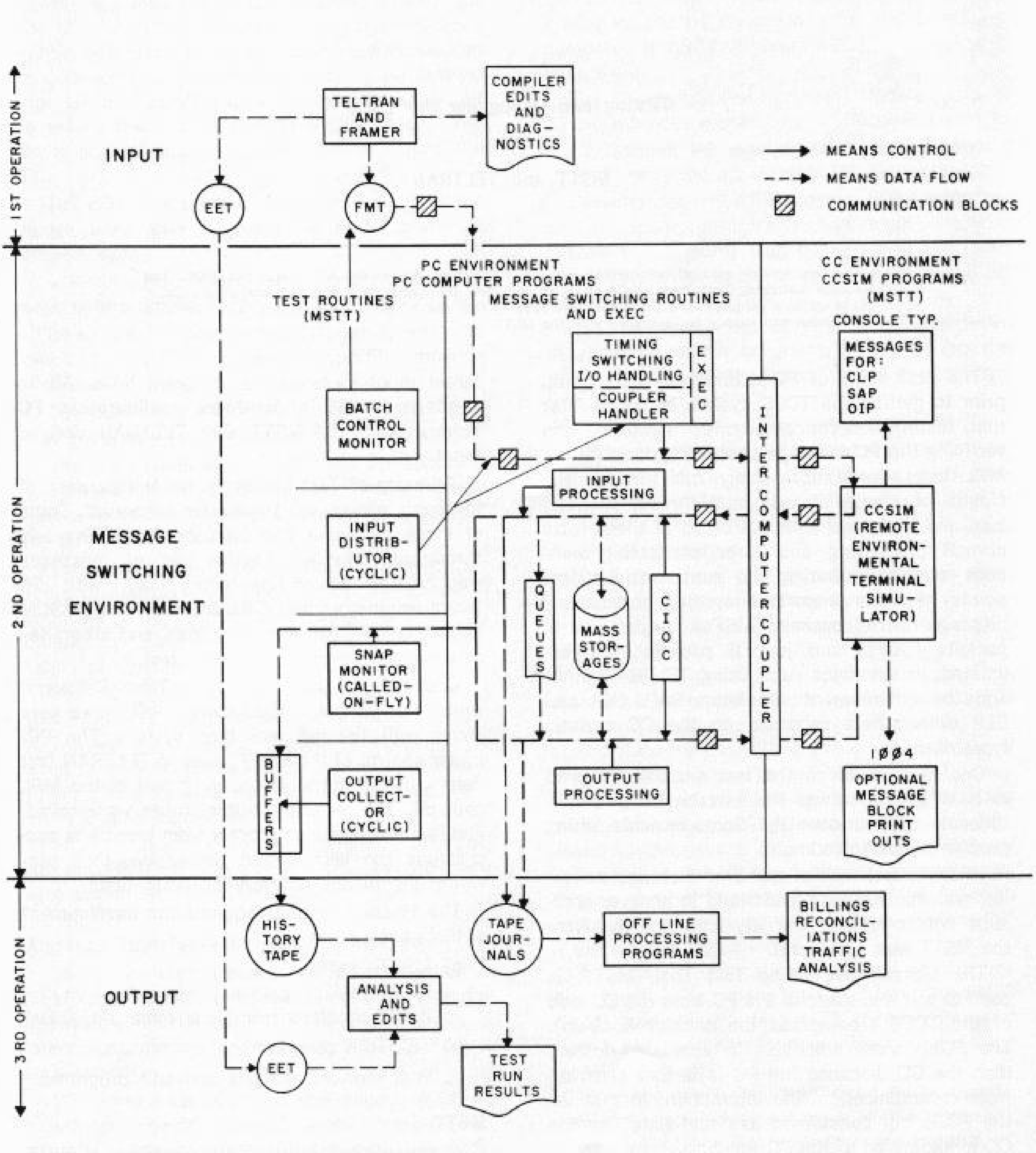


Fig. 2—Flow Chart Showing Operation of MSTT

at critical points of message processing. During a run, these snaps, along with many octal dumps are activated when the programs process a message, starting with the beginning of input processing and ending with the successful transmission of the message out of the system. These snaps and dumps are immediately moved to "safe" buffers to preserve their timeliness, in a real time environment.

PC Test Programs

When the test buffers are full, the PC test programs dump them onto a history tape (post mortem information). Thus, the sequence of events can be reviewed along with dumps of message control packets, etc., for integrity after the run is completed. Each snap is accompanied by such data items as the message line number, snap time, a mnemonic identifier, the program that called for the snap and the PC internal sequence number, when appropriate.

While Figure 2 illustrates the breakdown of the operating use of the MSTT into Input, Message Switching Environment and Output, it emphasizes the interaction of test data with a Message Switching Environment.

In the Message Switching Environment one of the PC test programs, the batch control monitor (BCM), is used to select the desired test batch on the FMT (Tape Control). After BCM initializes for a run, another PC test program, the Input Distributor, periodically reads one frame from the FMT into the PC every 5 seconds (the amount of time required to process one block on each line). The Input Distributor sends these blocks over the coupler to the CC remote terminal simulator (CCSIM) where the blocks undergo some final transformations. Then CCSIM transmits the blocks to the PC as if they had resulted from real CC system interaction with the input lines. CCSIM communicates thereafter with the PC in essentially the same way as the real CC programs. Messages transiting CCSIM may be printed out upon user option. When the Input Distributor senses an end of batch mark on the FMT, message input ceases for this batch. At the end of all activity for this batch, the batch control monitor writes an end of batch mark on the history tape. At the end of any batch runs, the journal tapes may be finalized and saved for off-line analysis.

In the third operation Output (off-system) the history tape and EET are used to analyze run results. EET is the Expected Events Tape.

CC Simulator

In the second operation, the CC Simulator, CCSIM, is used as an environment and remote terminal simulator.[2] Its function is to provide a dynamic interface with the PC from the point of putting a bid block on queue for transmission across the coupler. Test data is read in every 5 seconds from the FMT magnetic tape by the MSTT input distributor (IPDIST) which resides in the PC. IPDIST transmits test data blocks to the CC computer over the coupler. CCSIM receives these blocks and reblocks the data according to the appearance of special characters such as BT and EOM. Dynamic parameters such as the CC trunk connect number, time stamps, RTC indicators and BSN's are inserted. Thus, after some final transformations, the blocks are shipped back to the PC as if their contents had originated from live terminals. The CCSIM must interact with all blocks sent to the CC by the PC system. This includes messages destined for trunk lines from output processing in the PC.

A CCSIM option prints all incoming and outgoing blocks on the 1004. This is useful during low load runs for debugging.

Messages destined for CLP, SAP or OIP are printed on the CC console typewriter.

When the CC capacity (i.e. queues) is about to be exceeded, CCSIM informs the input distributor in the PC to stop test data input.

On any given "line," characters appearing between (EOT) and ZCZC are discarded. This provides an interarrival delay capability which is utilized by TELTRAN.

TELTRAN

TELTRAN is a versatile compiler language used to generate test data consisting of messages and associated control information. It is a language and processor developed to test the W. U. TCCS Phase I. It prepares test data for the MSTT. The input to the CC Simulator shown in operation 1, Figure 2, is obtained via TELTRAN.

TELTRAN generates messages on magnetic tapes in a suitable form to be used in testing significant portions of store and forward message switching systems. One of the output tapes can be used to generate paper tape input for testing a total live system from its various terminals.

Input

Input to TELTRAN is in a special source language on cards or on tape. Each job is processed as a batch with a source listing provided with error comments if errors are detected. The output of a run is a Raw Message Tape (RMT) and an Expected Events Tape (EET optional). The RMT contains all the messages generated for the current batch in serial order.

Each message on the RMT appears as a series of communication blocks. The first block simulates the bid block and contains various origination parameters which are specified by the user for each message. Message characters are contained in sequence in the following blocks. These blocks contain, in addition, user specified control information such as the originating line number. The blocks represent essentially the pre-simulated non-dynamic information that would normally be constructed by a "live" CC computer. Also these blocks are 32 words long and contain control information along with message data in packed ASCII format.

The format followed is essentially that required by the PC-CC communications programs. When one requires message data from many lines simultaneously another program (Framer), sorts the blocks (by line number) on the RMT to provide up to one block per line. A logical frame is defined as a set of from 1 to N (total # lines into computer) communications blocks. The result of this run is a framed message tape (FMT). On reading one logical frame of data, one block (when it exists) from each of the N lines is available immediately. In addition to the three tapes mentioned (EET, FMT, RMT), various edits of the message tapes are available. The RMT can be used for subsystem check out and for creating bulk punched paper tape (Baudot or ASCII) for live terminal input tests. The FMT should be used for MSTT operations, throughput studies and load testing.

A more detailed diagram of operation 1, of Figure 2, is illustrated in Figure 3.

The TELTRAN programs were written for a 65K (core) Univac 418 with 1004 and tapes. A 330 drum is useful but not essential. The framing program requires a 32K (core) Univac 418 with 1004, tapes and a FASTRAND. TELTRAN was written in 418 FORTRAN and ART (assembly language).

TELTRAN Processor (the Compiler)

TELTRAN is a one pass compiler that operates in batch mode. The user can compile and execute as many batches as he desires in one run. The compiler analyzes input as a continuous stream of characters. This syntax analysis is driven by a Turing Table with less than 50 active entries. Quite thorough error analysis of the user's source code is provided. The analyzed (parsed) source code is handed over to the semantics routine which generates object code, coordinates symbol tables and compiles a data dictionary which is to be referenced by the object code during execution, as shown in Figure 3.

If the batch has no errors at the end of compilation, it is executed interpretively so as to manufacture the RMT and EET. Errors detected during execution are logged for the user, to aid him in producing error-free test data.

Characteristics of TELTRAN

Since most programmers are familiar with FORTRAN, any other language with FORTRAN-like statements is simple to comprehend. FORTRAN is meant to provide the user with easy I/O control mixed with an ability to manipulate formulas. Here the problem of dealing with messages and message segments required that TELTRAN should give the user the capability to manipulate character strings of variable length. In addition, he had to build tables of character strings so that he could use the indexing features available in FORTRAN. With these available, only the introduction of DO loops was needed to provide the user with the ability to generate large quantities of messages easily.

In addition to normal message characters, certain special ASCII characters occur in the communications blocks. These characters are extensions of the ASCII alphabet, hence, a whole set of reserved names are required so that the user may insert these characters in the message stream if he so desires. Finally, an option is provided that enables one to specify, for each message, a string of data characters that can be used for automatic post mortem checking. This information goes out on the "expected events" tape, which can be saved for the post mortem run. The manner in which such post mortem analysis is carried out is up to the user and depends upon the particular system. The user must provide the analysis program and insert the necessary

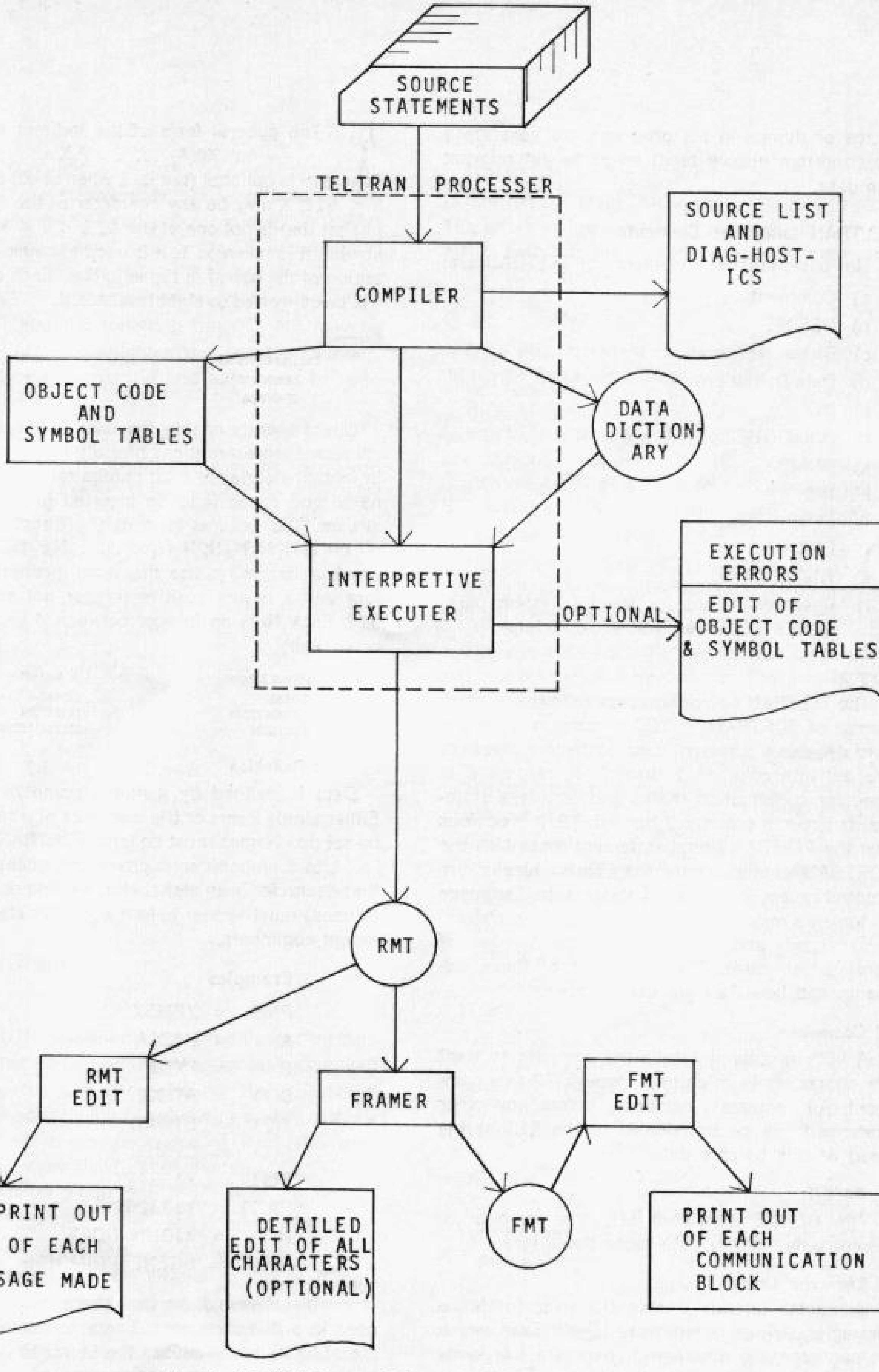


Fig. 3—Detailed Diagram of Operation 1 shown in Figure 2

snaps or dumps in his programs (to generate a post mortem history tape) so as to get relevant run data.

TELTRAN Language Elements

The basic language elements of TELTRAN are:

a) Comment
b) DEFINE
c) Character literals
d) Data Definition
e) DO
f) CONTINUE
g) DUMP
h) BID
i) STOP
j) END
k) FINIS
l) Reserved names for special System parameters (extended ASCII characters)

Format

The TELTRAN source language follows the gross format of FORTRAN. A "C" in column 1 on the card denotes a comment card, statement numbers are put into columns 1 through 5, column 6 is used for continuation marks and language statements occur in columns 7 through 72. It is obvious that the TELTRAN literal statement resembles the FORTRAN format statement. Thus, blanks are ignored except inside literal statements. Language statements may be spaced apart (between characters) at will and continued on any number of continuation cards. The definitions of these elements and how they are used follows:

a) Comment

A "C" in column 1 tells the compiler to treat the characters in columns 7 through 72 as a comment. All comments occurring before any other statement will be transferred to the EET at the head of this batch's data.

b) DEFINE

The DEFINE statement sets the compiler to accept data and/or data table definitions.

c) Character Literals

Character Literals enable the user to define character strings of arbitrary length from which he can compose messages. There are two kinds of literals—indirect and direct.

The general form of the indirect literal is

$$m'\ XA_1A_2\ .\ .\ .\ .\ .\ A_kX$$

(where m is optional (but $\geqq 1$ when used) and $1 \leq k \leq 512$. X may be any character of the ASCII alphabet that is not one of the A_i, $1 \leq i \leq k$. If m is absent, it is taken as 1. It is used to cause the repetition of the literal m times in line. Each of the A_i will be converted to eight level ASCII.

Examples

Indirect Literal	Result
'XABC2X	ABC2
2'AXYZA	XYZXYZ

Direct literals enable the user to create ASCII character representations himself. He may specify, in fact, any series of 8-bit configurations, give it a name and cause it to be inserted in a message stream. The general form of the direct literal is

$$m'\ \phi\, N_1\, N_2\, N_3\, N_4\, N_5\, N_6\ .\ .\ .\ .\ .\ N_{3k-2}\, N_{3k-1}\, N_{3k}$$

where ϕ is zero, m has the same meaning as before and k is any positive integer not exceeding 512. Each N_i is an integer between ϕ and 7 (i.e. octal digit)

Direct Literal	Result
2'0132	132132
'0143677554	143677554
2'0111234	111234111234

d) Data Definition

Data is defined by giving a name to literals. Either single items or the elements of a table may be set up. Names must be legal FORTRAN names, i.e., 1 to 6 alphanumeric characters such that the first character is an alphabetic. All data definitions (if used) must appear before any other statements except comments.

Examples

```
PMS    = 'XPMSX
A      = 2'AZCA
B      = 4'XNX
S(1)   = 'ATLXA
S(2)   = 'BTWXB
S(3)   = 'CPMSC
RL(1)  = 'X12242X
RL(2)  = 'Y12325Y
N(1)   = 'XJOHN DOEX
N(2)   = 'BGENE PRIMOFFB
```

It will be seen later that these items may be used in a BID statement. There, a reference to a literal by its name causes the literal to be placed in line in the message being created.

e) DO

The DO statement is used to cause the repetition of the message generation statements that lie between the DO and the designated statement number. The general form of the DO statement is

DO N1 N2 = K1, K2.

where N1 is the statement number of a CONTINUE. statement following the DO. N2 may be any legal FORTRAN name that is used as an index, K1 must be less than K2 and both must be positive integers.

f) CONTINUE

The CONTINUE. statement with its statement number in columns 1-5 delimits the TELTRAN code segment which was set up for repetition by a DO statement. Any number of DO loops may be nested as long as each has a unique continue statement.

Example

```
      DO  25  INDEX  =  1,  10.
      .
      .
      .
      DO  15  J  =  3,  6.
      .
      .
      .
15    CONTINUE.
      .
      .
      .
25    CONTINUE.
```

g) DUMP

The DUMP statement specifies an expected data stream that should be triggered by the associated message (the one generated by the BID statement immediately following) during an actual store and forward message switching run. The general form of the DUMP statement is

DUMP (A^1_1 A^1_2 A^1_3 A^1_4, ... , A^k_1 A^k_2 A^k_3 A^k_4).

Here, k may be any positive integer and the number of characters in each group between (, or , , or ,) may be between 1 and 4. The effect of this statement is to place, with a sequential message number and user designated line number (from the BID statement) the character stream A^1_1 . . . . A^k_4 in ASCII onto the EET.

Examples

DUMP (1).

DUMP (BEBP, HVAL, MCP, OCP, MACK, Q, FWD).

The order of appearance of these quartets in an actual test run will not in general be preserved in the TCCS system.

h) BID

Each BID statement causes the generation of a message. Its general form is

BID (M, , , , , . . . ,).

M may be any integer between 1 and N (the number of input lines) or a legal TELTRAN name that is also the index of a DO loop that surrounds the BID statement. It is here that each message is assigned a line number.

The next five items are origination parameters (such as the originators AB and terminal type) that are packed into the bid block for this message.

The elements following the last bid block parameter are separated by commas and may be any item from the following list:

Item	Example
Special reserved name	SOM
Indirect literal	2'AZCA
Direct literal	1'φ317
Non-indexed data name	ANAMEL
Constant indexed data name	ROUTE (3)
Variable indexed data name	RL (INDEX)
Index of a DO loop	INDEX

i) STOP

The STOP statement should only be used with card input to TELTRAN (card to tape to TELTRAN is optional). The STOP card halts the program so as to give the operator a chance to refill the 1004 hopper.

j) END

The END. statement ends the current message batch. An end-of-batch mark is placed on the EET and RMT.

k) FINIS

The FINIS. statement ends the current series of batches by placing an end-of-batches mark on the EET and RMT.

l) Special Reserved Names

The following list contains the special reserved names used for Western Union's Phase I application. Each name, when encountered in a BID statement causes the insertion of an appropriate eight level code.

Item	Meaning
1) SOM	Start of message
2) BT	Beginning of Text
3) EOM	End of Message
4) ERROR	The preceding character is an error code
5) DISCON	Disconnect
6) ACK	Acknowledge
7) ATI	Accept terminal input
8) CONN	Connect
9) CR	Carriage Return
10) LF	Line Feed
11) FIGS	Figures shift
12) LTRS	Letters shift
13) EOT	End of Transmission
14) WRU	Who are you
15) E1	First of 32 error codes
.	
.	
.	
E32	Last of 32 error codes

Advantages of TELTRAN

TELTRAN, currently being used for the TCCS-Phase I, has many advantages for testing in the communications and time-sharing computer industries. It is flexible enough to be used with other message switching systems. It is modular in design. The following elements are easy to change for a variety of applications:

The encoding of the message character set

The special reserved names

Error codes

Control information (in BID and DATA blocks)

The format of the output blocks

The contents and existence of bid blocks

The input language is similar to FORTRAN and has application in message switching. The output of TELTRAN is transparently simple in format and therefore is easily interfaced with the areas to be tested. Although the language is simple, its structure is quite versatile and limitless in its application. For example, the content of "messages" may be any data string. TELTRAN is a tool for generating bulk test data to service programming needs as well as message switching requirements in time shared systems.

Applications

A most promising application of TELTRAN is remote terminal simulation. TELTRAN can be interfaced readily with both Outer Remote Terminal Simulators (ORTS) and Inner Remote Terminal Simulators (IRTS). An ORTS simulates line traffic from a computer hooked back-to-back with a front end line servicing and buffering computer, and is used to check out an entire computer node in a message switching system. An IRTS resides in the computer node itself, for example just before the point of handing over buffered message data blocks to message switching routines. Thus, the IRTS is used to check out message switching functions.

By properly constructing message batches with TELTRAN the user can set up timing and traffic analysis runs. Header lengths and text lengths can be defined in indexable tables which represent a desired distribution of sizes. A series of runs could give a fair approximation to random sampling. We have had plans, in fact, to incorporate a random number generator within TELTRAN to give us the ability to sample given distributions randomly such as message interarrival time and message holding time. It is now possible to do the same thing with message interarrival delays by running a series of tests or by using the TELTRAN Random Input Line Loader (TRILL).

Any characters that appear on a "line", after an EOT (end of transmission character) and before the ZCZC, are discarded by CCSIM during a test run. Every block of characters (50 per block) discarded represents a real time delay of 5 seconds (a teletype line rate is 10 chars/sec.) One simply defines an indexable delay table and refers to it as the last item in each BID statement. For example

```
DO   1   I = 1,  12
BID (1, . . . ., EOM, EOT, DELAY1 (I) ).
BID (2, . . . ., EOM, EOT, DELAY2 (I) ).
1   CONTINUE.
```

In the MSTT, CCSIM performs the functions of an IRTS. TELTRAN and the MSTT were designed as early as September 1966 and have been operating successfully together since January 1967. Currently the use of terminal simulators has become a popularly accepted tool for developing communications systems software.

The MSTT runs in simulation mode which is considerably faster than real time. It is also highly controlled and therefore provides a useful tool in traffic analysis.

Potential for TELTRAN

The TELTRAN processor uses up to date techniques in table driven syntax analysis. The processor is functionally modularized and the language is written mostly in basic FORTRAN.

TELTRAN may be converted to run on almost any other computer. Finally, the closed loop testing capability provided by the DUMP statement can be a powerful controlled debugging tool in the hands of a resourceful user.

References

1. Real-time Systems in Perspective, J. D. Aron, IBM Systems Journal, Volume 6, Number 1, 1967.
2. Testing real-time system programs, M. G. Ginzberg, IBM Systems Journal, Volume 4, Number 1, 1965.
3. Information Service Computer System, John A. Hunt, Western Union Technical Review, Volume 21, Number 2, April, 1967.
4. Information Communications Service, Henry P. Bechtold, Western Union Technical Review, Volume 21, Number 2, April, 1967.

Acknowledgments

Bud Pine was instrumental in developing the TELTRAN concept with the authors. His valuable contributions and support made the project possible. We also wish to thank Robert Wilkins and Richard Bandat for their helpful suggestions in the structure of the language as well as compiler techniques. In the application of TELTRAN to Phase I, we recognize the assistance of Richard Burkett, especially for CCSIM, Jerald Greenberg, Phil Laplante, and Gordon Sanborn.

The authors wish to acknowledge the directing of Mr. James O'Reilly for his guidance in this article, which describes the hardware and software environment.

EUGENE H. PRIMOFF is a consultant to Western Union in test design and implementation of TCCS. His 12 years experience in mathematical applications on computers, operations research, simulation, software development and implementation is prelude to his present capacity as President of Worldwide Computer Services Inc.

Mr. Primoff has been involved with the Sage system test programs queuing models using GPSS II, command and control system design for the Air Force, Sort/Merge program systems, design of several assemblers and compliers as well as many communication system studies.

FRED C. HILSE is a programmer in the Planning and Engineering Operation and is most familiar with the test design of TELTRAN. He has written training documents for TCCS Training Presentation and has debugged and tested many CC programs.

He has been involved in the redesign of the PC Block Storage Pool and its implementation.

Mr. Hilse received his B.S. degree in Economics from Rutgers in 1964 and his M.S. degree in Economics and Statistics from the same university in 1966.

Our Customer Says:

National Distillers' order entry system uses a Western Union interface

by Philip J. Winterbauer

Corporate Communications Manager—
NATIONAL DISTILLERS and CHEMICAL CORPORATION

Western Union designed a multi-purpose interface to make the Model #28 teleprinter compatible with the McGraw-Edison "Address Memory Unit" Model 1000A. This interface is a very important link in the National Distillers' Liquor Order Entry System, as it opened the door to computerized order processing.

Our order entry system produces orders, invoices, reports, summaries ,and a report of orders received against quotas. All this is accomplished with minimum effort on the part of our order entry clerks as a result of the successful coupling of the above-mentioned units. Fig. 1 is a flow chart showing the steps in programming our order entry system.

The National Distillers order system components are;—a Western Union #111/115 private wire teletype system; McGraw-Edison "Address Memory Units"; an IBM 360/30 system and the Western Union specially designed interface, Magnetic Tape Control 12562-B. This is National Distillers' first application utilizing a magnetic tape storage unit with an ASR set.

The Tape Control unit (interface) consists of a control panel plus some associated electronic circuitry housed in a box. It is placed on a shelf to the right of the Model #28 ASR unit as shown in Fig. 2.

The interface allows order clerks to store five-channel programmed alpha-numeric information in the McGraw-Edison "AMU" and then play it back on "on command" to create input tapes for transmission to our computer. It takes telegraph line signal information, in standard Baudot code and records it on magnetic tape.

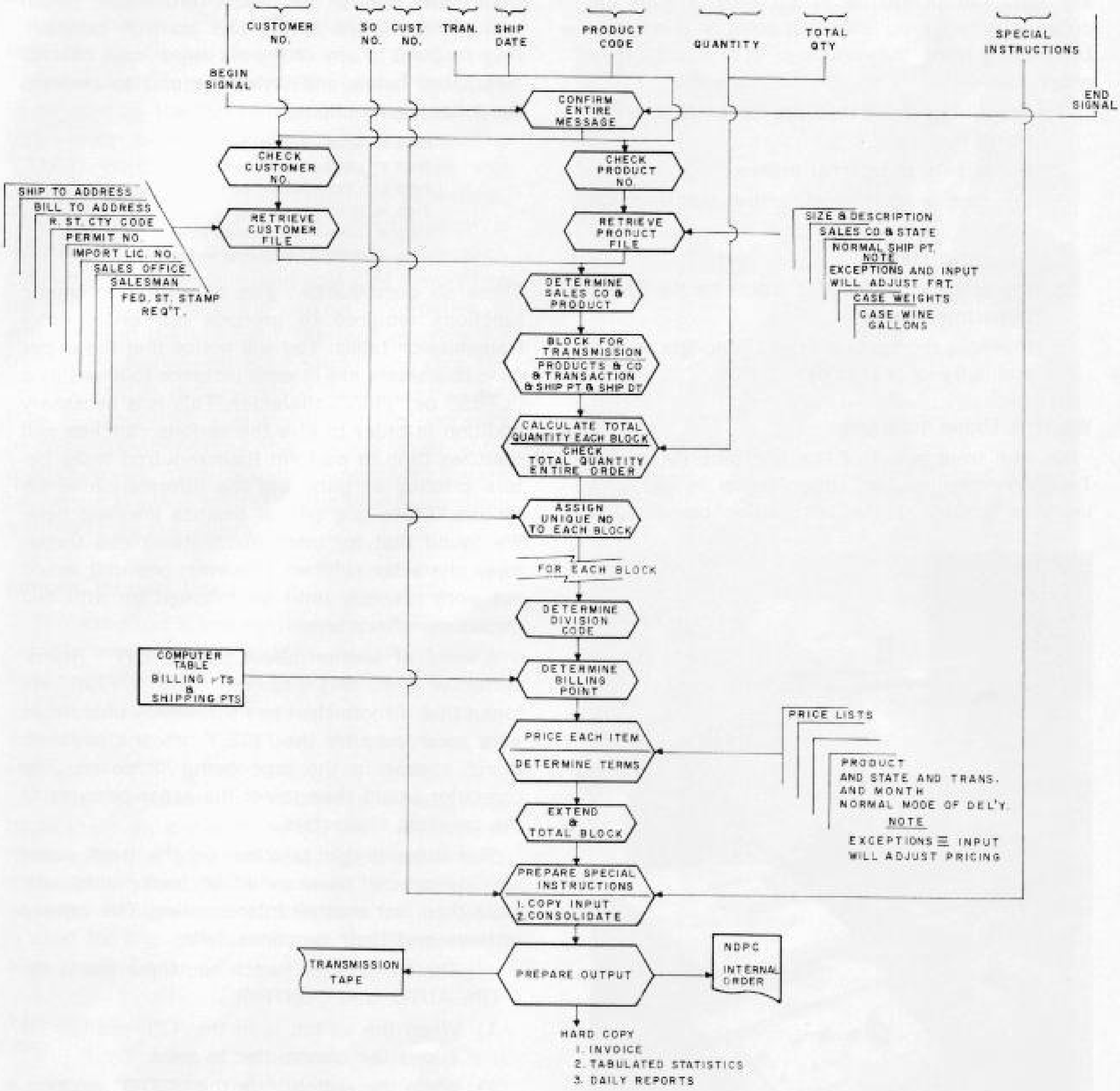


Figure 1. Flow Chart Order Entry System

Computer

The flow chart, illustrated in Fig. 1 defines the steps required to process an order in the computer.

First, the computer edits the input and rejects any data not presented in accordance with the order entry program. From that point on it isolates, by shipping point, the various parts of a customer's order and;—

1. Preprints sets of invoices for each shipping point,
2. Prints sets of internal orders,
3. Punches a 5-channel order transmission tape,
4. Tabulates statistics,
5. Prints a summary list of orders for the Credit Department, and
6. Provides the Central Traffic Manager with a summary of all orders

Western Union Interface

On the front panel of the interface (Magnetic Tape Control) housing, shown below in Fig. 2, a knob is located at the left, which couples the 28ASR set with the "AMU" in either a 'record' or 'playback' mode; it will also put the 28ASR set 'on line,' connecting it with the teletype circuit.

Figure 2. Front Panel of Magnetic Tape Control Unit. Unit is located on shelf so that the AMU may be placed on top of it.

Three toggle switches, located across the upper front panel, control the 28ASR transmitter, punch and printer. When set in the 'control' position, they respond to any of the six upper case characters listed below and order the unit to perform the following functions:

FIGS C LTRS—Printer "on"
FIGS F LTRS—Printer "off"
FIGS L LTRS—Punch "on"
FIGS N LTRS—Punch "off"
FIGS K LTRS—Intermediate stop
FIGS V FIGS—Transmitter or AMU "off."

These six combinations give us all the automatic functions required to produce our order entry transmission tapes. You will notice that the upper case characters are in each instance followed by a "LTRS" or "FIGS" character. This is a necessary addition in order to give the various clutches and switches time to perform their required tasks before printing or punching the information which follows. There is a critical balance involved here. We found that for most installations one throwaway character sufficed. However, one unit would not work properly until we followed up with two throwaway characters.

A word of caution about the "X OFF" (transmitter or AMU off) function, FIGS V FIGS. We found that if more than two throwaway characters were recorded after the FIGS V, ghost characters would appear in the tape being prepared. The computer would then reject the order because of the spurious characters.

The three toggle switches on the front panel are the special devices which make this unit more than just another intercoupling. The various settings and their functions follow.

The transmitter switch has three positions: ON, AUTO, and CONTROL.

1) When the switch is in the 'ON' position, it allows the transmitter to send.
2) When the switch is in the 'AUTO' position, it allows the transmitter to respond to selection from the control station "on line."
3) When the switch is in the 'CTL' (control) position, the transmitter is controlled by the three buttons immediately below. The buttons are labelled STEP, STOP, and START.

This is the feature that provides our National Distiller's order entry clerks with the ability to edit a bad tape quickly and accurately. With the knob set for 'PLAYBACK' and the toggle switches for the printer and punch set 'ON' and the toggle for the transmitter set for 'CTL' (control), a second tape can be quickly prepared.

The button labelled 'STEP' allows the tape to be read by the transmitter one character at a time—each time the button is depressed. The 'START' button allows the tape to read and duplicate until the 'STOP' button is depressed. Then, using the 'STEP' button until the error is reached, you step the tape through while each character is being duplicated on a new tape. When the error is reached, the correct information is entered from the keyboard, the tape reset in the transmitter to the next good section and then the 'START' button is pushed and the balance of the tape is duplicated. This eliminates the necessity of completely repeating the original order entry procedure,—a most valuable time-saving contribution made by the engineers who were responsible for producing the interface.

Address Memory Unit

The AMU has a locking device similar to voice recorders. When in the 'record' mode, it is locked in and a red caution light glows on its console.

The AMU has a storage capacity of 80 characters per line and 1000 lines per unit. Thus the storage of 80,000 characters is possible.

For fast selection of a customer or product playback there is a switch which will advance or reverse the magnetic tape at a high speed. A complete scan of the 1,000 lines takes less than eight seconds. A knob on the right side of the AMU is used for manual alignment of the desired item after the fast scan.

Advantages

The significance of this innovation is that it is a permanent record and replaces any other type of card or tape filing system, in much less space.

Simplicity of operation was also a major factor when we were considering the change over from our old system. This is borne out by the reply given by Mrs. Elizabeth Seward, our Eastern division order entry clerk shown in Fig. 3 when asked directly for her opinion. "It's simply marvelous. I processed an order the old way and it took 47 minutes. Then I did it the new way and it took only seven minutes. It's so simple, it is amazing."

Figure 3. **Mrs. Elizabeth Seward of National Distillers shown operating the W.U. Interface in the Order Entry System described by Mr. P. J. Winterbauer, author of this article.**

Successful Installation

In the development of the multi-purpose interface, Western Union engineers contributed greatly to the successful application of the #28ASR set to magnetic tape recording devices for punching raw data. It played a considerable role in the development and success of National Distillers' much improved order entry system.

Teams of systems men are already working on the further utilization of this basic system to provide for perpetual inventory, production scheduling and further applications as each of these is realized.

BOOK REVIEW

"The Anatomy of a Compiler," by John A. N. Lee published by Reinhold Publishing Corp., subsidiary of Chapman-Reinhold, Inc., 430 Park Ave., New York, N. Y. 10022, 275 pages.

The Anatomy of a Compiler is a "how to" book which fills some significant voids in the computer programming literature. It is a tutorial book intended for the professional programmer. The subject of this book, the anatomy (and construction) of a compiler, is of great general interest in the programming community.

The usual media for the presentation of new developments in computer programming—"Communications of the ACM" and "Datamation," for example—are frequently either overly esoteric or, because of space limitations, insufficiently comprehensive. Dr. Lee, head of the Computer Science Program at the University of Massachusetts, manages to overcome both of these objections.

This is not a book for the interested individual with no previous professional background in programming. Nor is it a primer for one who wishes to learn FORTRAN or BASIC. Rather, as the title implies, it presents the basic principles underlying the construction of a compiler and illustrates how they are applied. A reasonably competent working knowledge of at least one compiler language is a definite prerequisite for the reader.

Using the development of the FORTRAN language as a framework for his discussion, Dr. Lee traces the evolutionary path from language definition, to grammar, to symbol definition, and ultimately, to the generation of a machine code. Specific techniques such as Polish string notation, and the generation of machine codes from symbols thus recorded, are covered.

Dr. Lee's exposition of his material is by no means at the "Dick and Jane" level. It is nevertheless clear and highly professional. "The Anatomy of a Compiler" summarizes and organizes for the programmer much useful information which has previously been buried in the periodical literature, the proceedings of various symposia, or available only by word of mouth.

R. L. DuJack

New Subscription Rates for 1968

Starting with the 1968 issues, new subscription rates for the *TECHNICAL REVIEW* will be $10 per year.

Modernization Program

Hodgers, R. W., Jr.: View from the Technology Center
Western Union TECHNICAL REVIEW, Vol. 22, No. 1 (Winter 1968)
pp. 2-5

This article emphasizes the goals and objectives of the new Western Union Technology Center and points up the new types of shared services in the Phase I Modernization Program.

Computer
Simulation Techniques

Weitzer, B.: To Simulate or Not

Western Union TECHNICAL REVIEW, Vol. 22, No. 1 (Winter 1968)
pp. 6-7

Because large complex system problems cannot always be solved by mathematical equations, another method called Simulation has become popular in interpreting and evaluating solutions for study by the design engineer. Experiments are carried out on a "model" represented by the computer program.

This article points out that the arbitrary use of simulation is to be avoided. Instead a balanced use of analytic and simulation techniques is the most ideal.

ISCS
Simulation

Hanvey, Jr., F. R.: Modeling and Simulating the
ISCS—Phase I Processing Center

Western Union TECHNICAL REVIEW, Vol. 22, No. 1 (Winter 1968)
pp. 7 to 14

This article describes the computer simulation of a Western Union ISCS Phase I Processing Center. The decision to simulate was based on the complexity of the Processing Center.

The simulation model provides a means of observing the entire system in operation.

TCCS
Compilers
Testing Techniques

Primoff, E. H. and Hilse, F. C.: TCCS and Testing Techniques

Part I—Hardware and Software

Part II—Testing and Debugging
via MSTT and TELTRAN

Western Union TECHNICAL REVIEW, Vol. 22, No. 1, (Winter 1968)
pp. 15 to 29

The Telex Computer Communications Service, TCCS, interfaces with Telex, TWX and TMS. It is a real-time system. This article describes the characteristics of the service, the hardware and software used to test the processing center.

A description of the operation of the Message Switching Test Tool, MSTT, and the characteristics of the compiler language, TELTRAN, is included.

THESE ABSTRACT CARDS MAY BE CUT OUT AND PASTED ON LIBRARY CARDS FOR FILING.

New Technology Center

–A Showplace for Visitors

Increasing interest in Western Union's application of the advancing technology to communications is evident from a quick look at the recent list of visitors to the Technology Center at Mahwah, New Jersey. They have come from far off places, and as near as New York's Wall Street.

People are hearing about the Company's advanced work in message/data communications—and reading about it in the pages of TECHNICAL REVIEW—and are requesting that the Center be included in their itinerary.

For example:

Recently, a representative of the Postmaster General's Department of Australia spent a day at the Center with engineering, systems and planning personnel;

A contingent from Japan's Kokusai Denshin Denwa Company followed shortly after;

Nearly a hundred securities and financial analysts from New York City participated in a full day's orientation dealing with Western Union developments, including demonstrations of some of the new shared communications services;

The Company's Board of Directors got a first hand look at the Mahwah facility during their January 23rd board meeting, held at the Center for the first time.

Western Union's Technology Center is, indeed, proving to be an interesting place to visit.