THE BELL SYSTEM TECHNICAL JOURNAL
Vol. 55, No. 7, September 1976
Printed in U.S.A.

Test Facility for a Message-Switching System

By W. E. BRACKER and E. R. SEARS

(Manuscript received October 8, 1975)

This paper describes the development and use of a test facility for a computer-based store-and-forward message-switching system. The test facility provides a close approximation to actual operating environments, allowing the system to be tested in a realistic environment. The facility provides a vehicle for many essential system studies including performance monitoring and measurement, load testing, certification, and validation of various simulation and forecasting models. A wide range of tools is used in this facility—hardware and software monitors, terminal and network simulators, and data reduction and analysis packages. One version of the system is used to drive and thus test another version. A language for creating a set of test data is defined and is used to present various parameters to the switching system.

I. INTRODUCTION

A comprehensive test environment is important in the development and maintenance of any large computer system. In particular, when dealing with a message-switching system, special attention must be given to message loading, traffic mixes, and network configuration variables. Two methods of controlling these variables to test the system are:

(*i*) The use of real terminals and operators to drive the system, or
(*ii*) Simulation of terminal and operator system interactions to exercise the system as though it were in live operation.

The second method was used in our approach because it leads to more comprehensive, more controlled, and easily repeatable tests.

We describe the concept of "using the system to test the system"—a test facility is created from an existing system configuration. The system described is BISCOM,* a large-scale, computer-based, store-and-forward message switcher.

* Business Information Systems Communication System

Early in the development of BISCOM, the need for a comprehensive facility was recognized. The important factor in the creation of the facility was the need for a controlled test environment—one in which the operation of BISCOM could be observed and measured.

II. BISCOM OVERVIEW

Figure 1 illustrates a typical BISCOM installation. For reliability reasons, BISCOM is configured as a dual system with two central processing units (CPUs), each CPU having a complete set of peripheral units. One system serves as the Active (primary) message switcher while the other system acts as the Recover (backup) system in case of a failure in the Active system. This division is purely logical; either of the two systems may be the primary or recovery system (the communication network interfaces are attached to the machine serving as the active message switcher). BISCOM is designed to operate on a Xerox* Sigma-5 computer with core memory sizes up to 128,000, 32-bit words.

During Active system operation, all significant events are logged as follows:

Message Journals

(*i*) All User Input Messages
(*ii*) All System Internally Generated Messages
(*iii*) Messages Fetched or Delivered From System Data Files.

Logging Functions

(*i*) Input Start of Message
(*ii*) Input End of Message
(*iii*) Output Start of Message
(*iv*) Output End of Message
(*v*) On-line Network Changes
(*vi*) Disk File Transaction Data
(*vii*) Periodic Terminal Handler Statistics
(*viii*) Disk Retry and Parity Error Statistics.

The complete text of all switched messages are journalized (recorded) on magnetic tape. These tapes are input to a set of programs that keep the Recover system up to date with the Active system (e.g., they record the current state of the data base, network configuration, undelivered messages, etc.).

In the event of an Active system hardware or software failure, the Recover system becomes the Active system. At the present time this

* Xerox is a trademark of the Xerox Corporation.

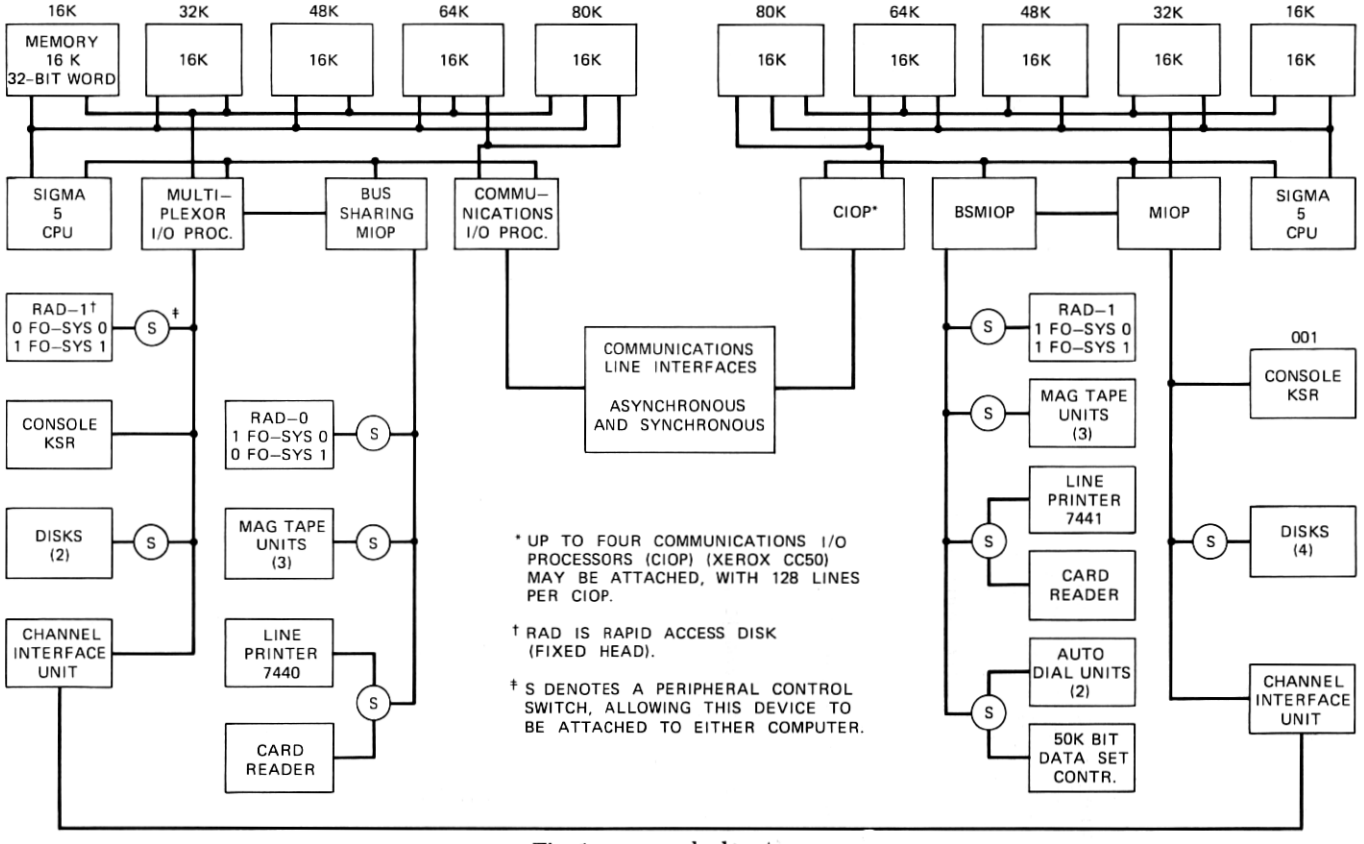

Fig. 1—BISCOM dual system.

process is manual, although testing of an automatic switch-over in case of failure is underway. In future versions of BISCOM, logs and message journals will be sent directly from the active to the backup system over a channel-interface unit.

BISCOM uses access arrangements including data-link control procedures that agree with those contained in Standard X3.28 (1971) published by the American National Standards Institute. These access arrangements include low and medium speeds, asynchronous and synchronous, respectively; switched and dedicated access lines; and remote computer access at speeds up to 50 kilobits/second.

Messages to be switched by BISCOM consist of a header and text; message framing is accomplished by three control characters: start of header (SOH), start of text (STX), and end of text (ETX). The system is text transparent in that only header information is scanned and validated, although illegal characters in the text are recognized. Information in the header specifies the actions BISCOM is to take with respect to the message. If the header is invalid or incomplete, delivery

Table I — Message mix and traffic loads for a typical busy hour

Type*	Input (Msg/Hour)	Length†	Deliveries‡			Output (Msg/Hour)
			Low	Med	Other	
POSM	125	100	1			125
SMST	1200	425				0
SMDL	200	500		1		200
SMFT	3000	25				3000
SMRD	700	440		1		700
SMRD§	500	440		3		1500
SMRD	100	440	3			300
SMRD	400	700	1			400
SMRD	600	700		1		600
SMRD	200	440	2			400
SMSD	1240	440		2	1	3720
SMSD	320	440		4		1280
SMSD‖	30	600	7			210
SMCL	125	25				0
SMRP#	625	425				0
Total Input	9365				Total Output	12,435

* See Appendix A (POSM = straight switched).
† Bytes.
‡ Low—110 baud, 150 baud terminals; med—2400 baud terminals.
§ 500 messages in the SMF file will be replaced with 440-byte messages. The new messages are delivered to three medium-speed terminals.
‖ 30 messages of 600 bytes each are stored in the SMF file and delivered to seven low-speed terminals.
625 messages of 425 bytes each will replace an equal number of messages already in the SMF file.

will not be attempted and the originating station is notified. If the header is acceptable, the message is switched to the designated location(s) and/or stored in one of the system data files. Appendix A provides a description of the message syntax accepted by BISCOM.

A BISCOM message mix is determined by the types of messages input during a given hour, while message load is determined by the number of input and output messages processed during this period. Table I shows a typical busy-hour message mix and traffic load for a sample BISCOM installation. The BISCOM busy hour is defined as a fixed percentage of the theoretical upper limit of input and output messages that can be processed during a selected peak 60-minute period.

III. TEST FACILITY OVERVIEW

The test facility is comprised of three major components: terminal and network simulators, hardware monitors, and software monitors. The logical configuration is shown in Fig. 2.

The terminal and network simulators drive the Active system; a set of real communication lines in the Active system is attached to simulated terminals in the Driver. In conjunction with a test message format handler in the Driver, the simulators respond to polling and selection sequences from BISCOM, and transmit and receive traffic.

Hardware monitors are attached to the Active system to monitor physical resource utilization through test points that include the CPU hardware and peripheral subsystem. Output from the monitors is

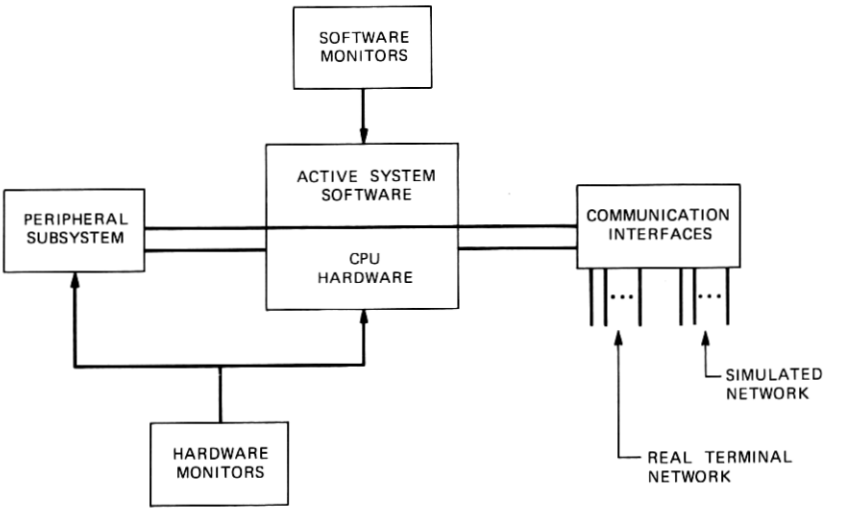

Fig. 2—The test facility.

stored on magnetic tape and forms input to a set of data-reduction routines.

The software monitors, which are incorporated in the code of both the Active system and the Driver, serve to measure system physical and logical resource utilization. The Active system outputs statistics every three minutes to a Reports terminal, and also logs significant system events onto magnetic tape; the Driver periodically outputs communication line profiles to a line printer.

IV. TEST FACILITY DETAILS

To examine the test facility in detail, we discuss

(*i*) Terminal, network, and traffic simulations.
(*ii*) Test messages.
(*iii*) Measurement tools.

4.1 Terminal, network, and traffic simulation

To create the Driver portion of the test environment, the normal recovery system is replaced with load Driver software, as shown in Fig. 3. The Active and Driver computers are connected by transmission lines that terminate in low- and medium-speed data sets. The network and corresponding communication lines are defined in both the Active and Driver systems by a network-generation program. This definition consists of core memory tables that characterize the lines in terms of speed, character code, and access arrangement.

The Active network consists of a set of lines terminating in real terminals and simulated terminals; the Driver consists of a set of

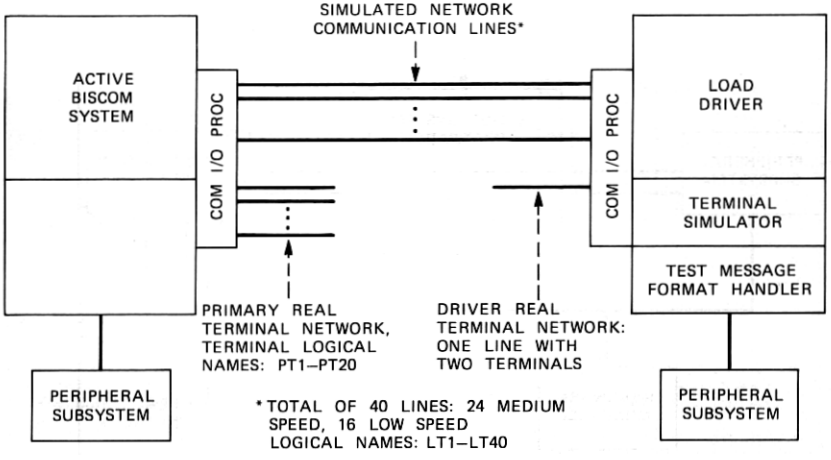

Fig. 3—Example of the active and load-driver configuration.

real terminals and terminal simulators. Active simulated terminals are connected to the Driver via low- and medium-speed data-set connections. During the course of a test run, traffic is received from a subset of driver terminals and directed to primary network real and simulated terminals and the various BISCOM data files.

To the Active system, the simulated terminals appear real—no software changes are made in the Active system to distinguish between the simulated and real portions of the network.

The Driver software represents a slightly modified BISCOM. Existing BISCOM software is used to perform support functions, such as message queuing, and special software is added that communicates with the primary system to generate a flow of traffic. This special software consists of two terminal simulators and a test-message format handler. One of the simulators handles medium-speed (2400–4800 bits/second, synchronous) communication lines; the other simulator handles low-speed (110 bits/second, asynchronous) communication lines. A Format Handler program intercepts test traffic queued to the Driver simulators and restructures the traffic for transfer to the Active system.

As far as the Driver software is concerned, simulators are identical to normal terminals. It is this concept which permits the BISCOM system itself to be employed as the Driver. When activated, a simulator makes the decision, when polled, whether to send a message, and, when selected, whether to receive a message. The simulators may also respond negatively to polls or selections for a predetermined length of time.

Driver test traffic input may come from one of two sources: (*i*) real terminals defined in the Driver network or (*ii*) a tape file. Messages that have been created by an off-line process and stored on the tape file may be sent by the Driver to the Active system and switched as if they were coming from actual terminals; of course, such messages must conform to valid BISCOM syntax. Prior to a test run, a tape is generated representing a specific load test; for example, it may be desirable to perform a 1-hour test representing a load of 40 percent of an hour's traffic mix for a particular installation. The necessary parameters are input to a PL/1 program (SIMGEN) which generates an appropriate test tape (see Fig. 4). Currently, a library of test tapes is maintained, representing many different loads, message mixes, and test durations.

4.2 Test messages

Message-load and message-mix data is jointly forecasted by a Bell System operating telephone company and Bell Laboratories. This data determines specific message inputs that are made available to the

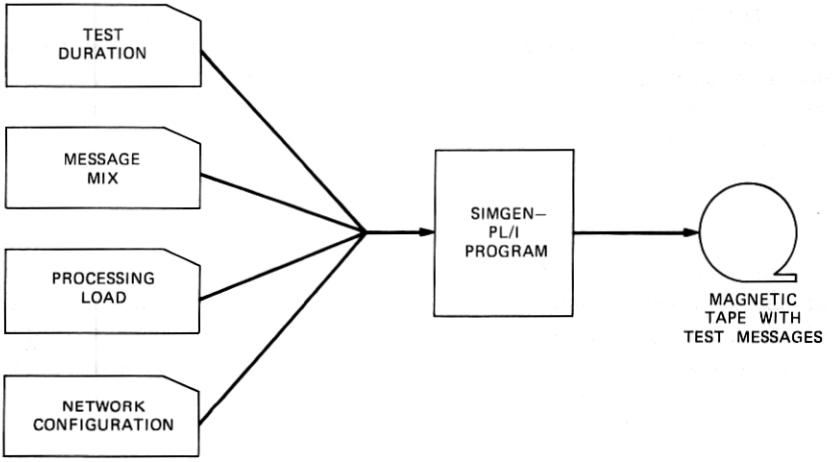

Fig. 4—Process of creating test-message tape.

driver system. These inputs are associated with the simulated terminals, and the driver system sends these messages to BISCOM at a controlled rate. Conversely, the messages sent to the primary system will stimulate activity and responses (i.e., deliveries of messages to simulated and real terminals); therefore, the rate at which the driver accepts messages from BISCOM must also be controlled. By establishing this control on the transmission and receipt of messages associated with the primary system, a load capability is created.

4.2.1 Test message format

A BISCOM message has the format

(SOH) Header (STX) Text (ETX).

This syntax does not change in the load driver although message processing is modified by the terminal simulators and format handlers.

The basic BISCOM format is expanded within the text portion to form the syntax of the test message

$$(\text{SOH})\ \frac{\text{DH}}{\text{HEADER}} \qquad (\text{STX})\ \frac{\text{CS}}{\text{*POL p}\quad \text{*SEL s}}\ \frac{\text{PH}}{\text{*SOH HEADER}}$$
$$\frac{\text{PT}}{\text{*REP r *STX TEXT *PAD d}}\ (\text{ETX}).$$

The test message is broken into four major sections: DH (Driver Header), CS (Control Section), PH (Primary Header), and PT (Primary Text). These message sections (*i*) direct test messages to driver-simulated terminals, (*ii*) control the responses of these terminals to

BISCOM polling and selection, and (*iii*) define the text patterns sent when positive poll responses are desired. Examples of various test messages are given in Appendix B.

4.2.2 DH (driver header) section

This section designates a simulated terminal in the load-driver network. Normal BISCOM mechanisms are used to queue the test message to a simulated terminal; however, instead of its normal delivery of the message to a real terminal, BISCOM releases all control over the message and activates the appropriate terminal simulator. The simulator, as part of its operation, will call the test-message format handler which reconstructs the message for delivery to the primary system.

4.2.3 CS (control section)

The test-message control section controls the response of the simulated terminals to poll and select sequences from the primary system. By regulating the time between positive polling or selection responses, it is possible to present any desired message-transfer rate to the primary system.

The *POL p and *SEL s subsections specify that a delay is to be introduced before either a positive *polling* response or a positive *selection* response is given. Polling and selection response delays stay in effect until reset by another test-message control section.

4.2.4 PH (primary header) section

Entries in this field may represent any valid BISCOM header information, as shown in Appendix A. When a terminal simulator is ready to give a positive response to a primary polling sequence, the *SOH will be converted to the (SOH) control character. The header has no processing performed on it and must correspond to valid primary network terminals and/or data-base operations.

4.2.5 PT (primary text) section

This section defines the text portion of the test message. As in the case of *SOH, the test message format handler converts the *STX to the (STX) control character and performs other operations on the TEXT subsection if specified by the *REP subsection. *REP r operates on the PT section and on the PH section simultaneously, *r* specifying how many times this particular message is to be repeated before the simulator releases it and goes to the next test message. The *REP subsection is also used in conjunction with a special character '#', which is used for sequence numbering.

The TEXT section may be alphanumeric strings of any length. *PAD d specifies how many repetitions of a 20-character "filler" sequence are to follow the TEXT subsection. In this manner, messages of varying lengths may be transferred to the Active system without the need to supply all the characters in the test message.

4.3 Measurement tools

The measurement tools available in the BISCOM test facility may be grouped into three categories:

(*i*) Hardware monitors
(*ii*) Software monitors
(*iii*) Logging analysis programs.

The use of these tools along with their advantages and shortcomings are discussed in the next section.

4.3.1 Hardware monitors

COMRESS Corporation's Dynaprobe® hardware monitors, models 7900 and 8000, are currently used in the test facility. These monitors are used to collect data concerning system hardware component activities; for example, they measure the number of seeks, reads, and writes performed by the disk subsystem; the amount of disk controller and disk pack utilizations; the amount of Central Processor utilization; and the amount of magnetic tape controller activity. The hardware monitors also have the capability of monitoring instruction-execution activity as a function of instruction address, making possible the production of "maps" of instruction activity by memory location (see Fig. 5). This capability has been particularly useful in locating anomalous program loops and in selecting program modules as candidates for code optimization.

A hardware monitor does not perturb the system being measured and thus does not bias the measurements made. In addition, hardware monitors have a certain amount of flexibility and provide a way to make measurements that are not readily attainable by other means, e.g., measuring instantaneous CPU utilization, core usage, and interrupt enable/disable times.

The data gathered by the hardware monitor is used in the validation of various simulation and forecasting models of BISCOM. There are, however, many software-related measurements that cannot be obtained by this method. For example, it is difficult to measure buffer utilizations, message processing times, and task activations. Software monitors must be used to measure this activity.

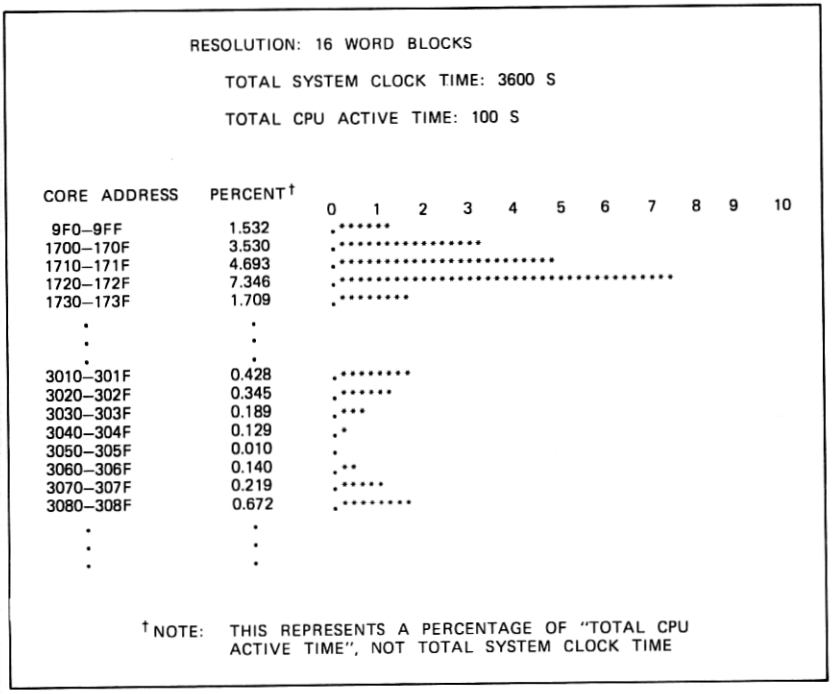

Fig. 5—Typical reduced output of hardware monitor.

4.3.2 Software monitors

There is limited software monitoring capability built into the Active system and Driver. The Driver software contains a statistics-gathering routine that periodically prints statistics on the network interactions with the Active BISCOM system. There is also an Active system software monitor routine that collects data on buffer pool utilizations, queue lengths for various processing tasks, and the number of messages entering and leaving the system during specified time intervals.

Significant system events are also logged on magnetic tape for use by the Recover system. In the test environment, these log tapes serve as inputs to an analysis program. This analysis program outputs statistics on message volumes and switch time (Table II); future plans call for reporting data-base activity and network performance through expanded system logging. Such logging is part of normal BISCOM operation, and the test facility makes use of already available data.

Two of the major advantages of software monitors over hardware monitors are the wider range of system attributes that can be measured and the greater flexibility in the control of data collection. One

Table II — Typical log analysis output

Message Type	Number of Messages	Average Resp-Time (s)	Average Chars In	Average Chars Out	Messages Delivered/ Cleared
POSM*	5	0.06	89.00	89.00	5
SMST	68	0.57	422.29	0.00	0
SMSD	64	1.89	424.87	424.87	152
SMRP	25	0.72	415.40	0.00	0
SMRD	103	1.32	530.04	530.04	146
SMCL	3	0.82	25.00	0.00	3
SMFT	129	0.52	25.00	516.25	129
SMDL	6	0.87	28.00	563.66	6

Window times: 02:55.34–03:00.34; window length: 5.00 minutes
No. messages = 403; 99.3% load; 100% input-message load = 4860 msgs/hr

* Straight switched messages.

of the disadvantages is that a software monitor interacts with normal system operation and thus uses some of the system's resources. Another disadvantage is that special data-reduction-and-analysis software must be developed for a software monitor.

Note that one use made of the hardware monitor is to estimate the impact of the software monitor operation on normal system processing; in particular, it is possible to determine how much CPU time the software monitor is using.

V. SOFTWARE TEST ACTIVITY AND SYSTEM CERTIFICATION

The tools comprising the test facility are being employed for software test purposes as well as for performance measurement and evaluation studies. The most heavily used aspect of the test-facility tool is the load-test capability. Whereas a load test is performed on *each release* of BISCOM prior to site installation, a measurement-and-evaluation effort is only applied to *selected releases* of the system. There are two major reasons for this:

(*i*) Data collection, reduction, and analysis is a costly process,
(*ii*) Software changes on a per-release basis usually do not warrant a complete measurement-and-evaluation effort. However, those releases that contain performance-affecting changes have full-scale measurement-and-evaluation procedures applied prior to site distribution.

The load test is the culmination of prerelease test activity on a given set of software changes. Prior to the load test, the software changes

are tested by the change originator for their functional correctness. The changes are then integrated into the BISCOM software, and the entire system is subjected to a standard test series. The final stage in the test procedure is the application of the load test.

As discussed in Section I, the load test has essentially three parameters: message mix, message load, and network. The message mix and related network configuration cause the software to react in different ways depending upon the processing requirements on a specific message type (designated by the message-header information). A message that requires a store into the data base has different requirements than a message that is switched directly from terminal to terminal. Likewise, messages that access different data files in the BISCOM system have different processing requirements. The message-load parameter allows an approximation to the message-processing load expected to be experienced in an actual operating environment. This processing load is jointly forecasted by a Bell System operating telephone company and Bell Laboratories. Beyond these parameters, system dependent variables can be stressed in a manner designed to cause system actions that are not normally met in a simple test. Typical variables stressed are buffer exhaustion and buffer boundary conditions.

The load test is the final test that is applied before release of the software product. Data analysis is performed and compared with the analysis of previously released systems in a search for any discrepancies that might point to a design flaw. This analysis also ensures that the system does not regress (from previously released systems) in load or functional capability.

After the software system is released, the load test is again used to exercise the released system on a continuous basis in search of design flaws or dormant software bugs. Exercising the system on a continuous basis involves activating the system and switching messages for several consecutive hours.

5.1 Test results

Test results to date can be classified into two categories, namely,

(*i*) Uncovering of software problems.
(*ii*) Certification of the system-load capacity.

The first category of test results includes an entire range of software problems. A typical example in this category was uncovered during system shutdown, the time during which the system cleans up traffic in progress, secures its data base, and transmits "good night" messages

to the terminal users. The system performed this function adequately when not under load; however, when shutdown was attempted in a loaded environment, it did not function properly. This condition was corrected before it occurred in a real environment.

Another example of category (*i*) is the saving of machine states when high-priority tasks preempt current system activity. The software associated with "state" saving and restoring operated correctly until stressed in a load-test environment. This condition was also corrected. Violations of reentrancy conditions in the data-management functions were also uncovered and corrected in the laboratory environment.

We also found that the load-test facility is capable of exercising the various hardware components to such an extent that it can cause the hardware (disk units, tape units, etc.) to fail and thus force the system to exercise its retry logic. This is highly important, since many hardware conditions and status responses are very difficult to create during normal system operation.

The second category of test results relates to the throughput or capacity of the system. The point has already been made that system capacity is dependent upon the three variables: message type and size, message load, and network configuration. The laboratory has a close approximation to these variables as they relate to several live installations. In this environment, it has been possible to certify the capability of the system to process future traffic loads for these installations. This has been very useful information for site planners, particularly as it affects plans for routing more traffic through the system.

VI. ACKNOWLEDGMENT

The work presented here has been the result of the efforts of many people associated with the BISCOM project. The authors wish to acknowledge all who have contributed to the design, development, and maintenance of this test facility.

APPENDIX A

Message Syntax

*A.1 Terminal-to-Terminal (Straight Switch)**

Messages may be sent between terminals, or to and from the message storage files. The general format of each message consists of heading and text sections framed by delimiters. The heading must be correctly structured, whereas the text section is "transparent" to BISCOM.

* This message type does not cause access of any system data files and is switched directly from an originating terminal to destination terminal or terminals.

(SOH) DAC1 DAC2 . . . DACn (STX) TEXT (ETX), where:

(1) DACi is any 1–8 alphanumeric terminal logical name.
(2) The DACs must be separated by blanks.
(3) The total number of characters between (SOH) and (ETX) must be less than 256.
(4) The total number of characters between (SOH) and (ETX) must be less than 6400.
(5) TEXT may be any string that does not include certain control characters [e.g., (SOH), (ETX), etc.].

A.2 Special Message File (SMF)

(SOH) SM**. DAC1 . . . DACn/MSGID/PAGE/DATE CONTROL MM-DD (STX) Text (ETX)

where ** indicates one of the following process codes:

ST—Store	FT—Fetch
SD—Store and Deliver	DL—Deliver
RP—Replace	CL—Clear
RD—Replace and Deliver	

MSGID (Message ID)

Associates a storage and retrieval key with the message in the data base, any alphanumeric string from 1 to 15 characters.

PAGE

Any number between 1 and 99; the last page is denoted by the suffix L (e.g., 10L); associates a multipage message with a single-message ID.

DATE CONTROL

D—Deliver on MM (month), DD (day) date, current year
P—Purge the message on the MM, DD date
C—Deliver and purge the message on MM, DD date.

Example:

Store a message in the SMF of one page under the key ID100, deliver this message to terminals 37ASR1, SAND1; purge the message on July 11 of the current year.

(SOH) SMSD. 37ASR1 SAND1/ID100/1L/P 7-11 (STX) TEXT (ETX)

Fetch the above message:

(SOH) SMFT. /ID100 (STX) (ETX) (The message will be delivered to the terminal initiating the fetch.)

The SMF is the most heavily used system data file. Other system files are: Authorized, Resend, Batch, Paging, and Error. They have syntax similar to the SMF.

APPENDIX B

Test Message Examples

Communication lines in a BISCOM system normally terminate in more than one terminal; however, it is assumed for purposes of this discussion that each line terminates in only one terminal with the logical names given in Fig. 3. Consider the following test messages queued to driver-simulated terminals; the # character specifies a field used by the driver for message-sequence numbering.

INPUT TO DRIVER TERMINAL LT1:

(SOH) LT1 (STX) *REP 100 *POL 350 *SEL 100 LT2 PT1 *STX Test A### (ETX).

Sent to BISCOM From Simulated Terminal LT1:

(SOH) LT2 PT1 (STX) Test A001 (ETX)
⋮ 350 ms between messages sent to BISCOM
(SOH) LT2 PT1 (STX) Test A100 (ETX).

Delivered to Simulated Terminal LT2:

(STX) Test A001 (ETX)
⋮ 100 ms between deliveries
(STX) Test A100 (ETX)

Delivered to Primary Real Terminal PT1:

(STX) Test A001 (ETX)
⋮
(STX) Test A100 (ETX).

The following example shows how a test message may be used for data base operations:

INPUT TO DRIVER TERMINAL LT1:

(SOH) LT1 (STX) *REP 9 *SOH SMSD. RT1 LT#/BTL## *STX TEST ## (ETX):

Sent to BISCOM from Simulated Terminal LT1:

(SOH) SMSD . RT1 LT1/BTL01 (STX) TEST01 (ETX)
⋮
(SOH) SMSD . RT1 LT9/BTL09 (STX) TEST09 (ETX).

Stored in Primary System Special Message File (SMF):

TEST01 Stored under key BTL01

⋮

TEST09 Stored under key BTL09.

Delivered to Primary Real Terminal RT1:

(STX) TEST01 (ETX)

⋮

(STX) TEST09 (ETX).

Delivered to Simulated Terminals LT1–LT9:

(STX) TEST01 (ETX) to LT1

⋮

(STX) TEST09 (ETX) to LT9.