AT&T Bell Laboratories Technical Journal
Vol. 63, No. 9, November 1984
Printed in U.S.A.

1982/83 End Office Connection Study: ASPEN Data Acquisition System and Sampling Plan

By J. D. HEALY,* M. LAMPELL,* D. G. LEEPER,*
T. C. REDMAN,* and E. J. VLACICH*

(Manuscript received December 19, 1983)

To characterize the transmission performance of the public switched network, the Bell System conducted an intensive field measurement study of end-office-to-end-office connections from October 1982 to January 1983. A special multistage sampling plan and ASPEN, a flexible, automatic data acquisition system based on the *UNIX*™ operating system, were developed to evaluate the more than 10,000 transmission paths included in the study. This paper describes both the ASPEN measurement system and the sampling plan. A companion paper in this issue of the *AT&T Bell Laboratories Technical Journal* describes the network transmission performance results.

I. INTRODUCTION AND SUMMARY

The Bell System traditionally conducted field measurement studies of network transmission performance to evaluate existing administrative and maintenance procedures and to set objectives for new transmission systems and equipment.[1-11] The 1982/83 End Office Connection Study (EOCS) had additional goals arising from Computer Inquiry II and the 1984 divestiture of the Bell operating companies from AT&T. Specifically, the EOCS was intended to support allocation of performance objectives across segments of the network; to aid planners of new telecommunications systems, services, and terminal equipment;

* AT&T Bell Laboratories; present affiliation Bell Communications Research, Inc.

and to serve as a benchmark against which new system arrangements and services might be compared. The EOCS and allied studies now provide a database to support work on both the traditional and divestiture-related goals.

To conduct the EOCS, a software-controlled field measurement system called ASPEN (Automatic System for Performance Evaluation of the Network) was developed, a special sampling plan was prepared, and software tools were adapted to allow for rapid screening and analysis of the measurement data. This section presents an overview of ASPEN and the EOCS sampling plan. The accompanying paper describes the study results in detail, along with a description of the data analysis methods.[12]

1.1 The ASPEN Data Acquisition System

The most recent field measurement study comparable to the EOCS is the 1969/70 Connection Survey.[8] In that study, manually operated measurement equipment was carried from office to office for a period of one year, and a total of about 600 transmission paths were evaluated. (A given *test connection* offers two directions of transmission for testing, hence two *transmission paths.* In the 1969/70 connection survey, measurements were made on one transmission path per test connection.) Measurement data were entered manually on a terminal connected to a central computer. While this procedure worked well, a highly automated data acquisition system was deemed desirable for the EOCS for two fundamental reasons.

First, setting performance objectives requires accurate information about the tails of performance distributions, including distributions conditioned on end-office switch type, time of day, mileage band, and other criteria. It was estimated that a sample size of 10,000 transmission paths was needed to meet the study goals. The time and cost of manual data acquisition would have been prohibitive.

Second, a modifiable and reusable measurement tool appeared to be the most efficient way to meet future demand for field performance measurements of inter- and intraexchange network segments, as well as a host of planned new services. Progress during the 1970s in automatic measurement equipment and mini/microcomputer hardware and software had made the development of such a tool possible.

As Fig. 1 shows, the ASPEN application to the EOCS consisted of 20 remotely controlled instrumentation packages (called Remote Test Units or RTUs) connected to the line side of mainframes in selected Bell System end offices. Under the control of a host computer, the RTUs called one another over the public switched network just as actual customers would. Once a connection between two RTUs was

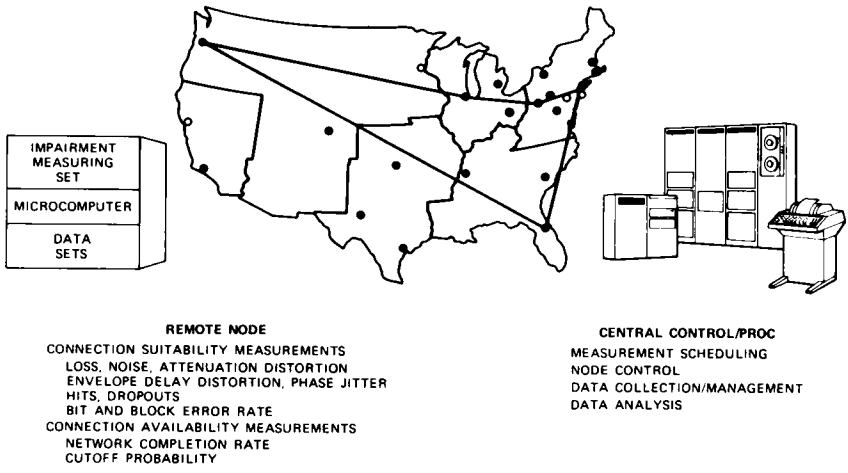

Fig. 1—ASPEN System: End Office Connection Study.

established, automatic measurement equipment in the RTUs evaluated more than 25 transmission parameters on that connection (see Table I).

The ASPEN structure offered several advantages over the 1969/70 approach, the most notable of which was speed. Evaluating more than 120 transmission paths per day, the equivalent of the 1969/70 data was gathered in five days as opposed to a full year. In all, over a period extending from October 1982 to January 1983, parameters were measured on more than 10,000 paths from over 7,000 test connections. (In the EOCS, approximately 3000 of the 7000 test connections were evaluated in both directions, while the remaining connections were evaluated in one direction only. Thus the total number of transmission paths evaluated was approximately $2(3000) + 4000 = 10{,}000$.)

A second advantage offered by the ASPEN structure was flexibility. As described in Sections II and III, the sequence of operations at an RTU was controlled by a microprocessor whose software was down loaded from the host computer. The measurement sequence could be, and was, changed as needed.

Finally, because the data screening and preliminary analysis ran concurrently with data acquisition, information was available to make changes in the data acquisition process while the study was still in progress. The analysis tools also made it possible to obtain final results quickly, despite the much larger volume of data.

A particularly challenging aspect of ASPEN development was the preparation of host computer software that could handle fault conditions gracefully. The ASPEN host computer for the EOCS was a

Table I—1982/83 End office connection study parameters measured

Voice and Voiceband Data	Insertion loss (1 kHz)
	Loss vs. frequency (30 freqs)
	C-message noise
	Call cutoffs
	Propagation delay
Voiceband Data	Signal-to-noise ratio
	C-notched noise
	3-kHz flat noise
	3-kHz notched noise
	Noise-to-ground
	Envelope delay vs. frequency (30 freqs)
	Peak-to-average ratio (P/AR)
	2nd-order intermodulation distortion
	3rd-order intermodulation distortion
	Phase jitter (2-300 Hz)
	Phase jitter (20-300 Hz)
	Amplitude jitter (2-300 Hz)
	Amplitude jitter (20-300 Hz)
	Impulse noise (6 thresholds)
	Gain hits (3 thresholds)
	Phase hits (3 thresholds)
	Dropouts
	Frequency shift
	1200-b/s bit/block error rate
	4800-b/s bit/block error rate

*VAX-11/780** computer running under the *UNIX* operating system. As described in Section III, ASPEN's host computer control software consisted of three distinct layers responsible for call management, connection supervision, and overall connection scheduling. This structure successfully handled the occasional dropped connection, intermittent RTU failure, or host computer service interruption (both scheduled and unscheduled). As a result, ASPEN could run virtually unattended on a daily basis.

In addition to providing a friendly environment for program development, the *UNIX* operating system offered data storage and analysis tools to permit data screening and analysis to run concurrently with data acquisition. A relational database package and the *S* statistical analysis package (created at AT&T Bell Laboratories) were a key part of the ASPEN system and the EOCS. Together with special C language data screening programs, these tools were used to ensure data integrity, to store the data in a compact, logical structure, and to allow rapid exploratory analysis of results. Because the data screening programs accepted data directly from the RTUs, manual handling of measurement data was completely eliminated.

1.2 The end office connection study sampling plan

The sampling plan included both RTU location (spatial) and test

* Trademark of Digital Equipment Corporation.

connection scheduling (temporal) components. As described in Section IV, RTU locations were selected to yield performance data representative of different mileage bands, end office switch technologies (*ESS*™, crossbar, and step-by-step switching equipment), and other strata. A unique aspect of the EOCS spatial sampling plan was that the sampled units, end offices, were quite different from the units being evaluated, namely, telephone calls.

With 20 RTUs deployed, there were 380 possible originating/terminating pairs over which calls could be placed. The EOCS temporal sampling plan, operating through the ASPEN Scheduler software, managed the selection of the RTU pairs. A unique feature of the temporal sampling plan algorithm was its ability to enhance the number of busy-hour connections and to handle gracefully the additions or deletions of RTUs caused by equipment problems.

1.3 Summary

The ASPEN approach to acquisition of network performance data proved highly successful; data were gathered more than 20 times as fast as by previous manually oriented methods. The ASPEN host computer software structure shows how a fault-tolerant automatic performance characterization system may be implemented, and the ASPEN spatial and temporal sampling plans show how locations may be chosen and measurements scheduled to evaluate a network or other telecommunications service.

Sections II and III provide a detailed description of the hardware and software design of ASPEN as it was applied to the EOCS. While the components used in ASPEN are explicitly listed, the hardware description is a generic one because there are many possible choices for RTU and host computer components. The software description in Section III is more explicit. While the actual code is not presented, the structure is described in some detail, with emphasis on fault-handling capabilities.

Section IV describes how the a priori study requirements and the opportunities offered by the fully automated ASPEN approach combined to create special sampling challenges. Also described is an iterative preselection step that made it possible to meet the need for stratification of results by mileage band and end-office switch technology (*ESS*, crossbar, or step-by-step switching equipment).

II. ASPEN HARDWARE SUBSYSTEMS

The ASPEN data acquisition system is composed of two major hardware subsystems: the remote test unit and the host computer. This section describes the hardware components used in each of these subsystems.

2.1 Remote test unit hardware

The Remote Test Unit (RTU) subsystem has five major hardware components: a microcomputer that controls the operation of the RTU, a remotely controlled transmission Impairment Measuring Set (IMS), "test" data sets used in conjunction with a bit and block error rate testing capability, and a switching matrix. Table II lists the specific equipment used for the EOCS, and Fig. 2 shows a complete RTU, ready for shipment and installation.

Figure 3 is a schematic representation of the ASPEN data acquisition system. The host computer and RTU microprocessor communicate by means of 1200-b/s full-duplex data sets operating over an ordinary dial-up connection through the public switched network. Using the switching matrix, the microprocessor connects the line under test, IMS, test data sets, Bit Error Rate Receiver (BERR), and Bit Error Rate Transmitter (BERT) in the various configurations required to measure the parameters listed in Table I.

The IMS measures all the parameters in Table I except bit/block error rate and propagation delay. Selection and testing of the IMS, an especially critical phase of the ASPEN project, was guided by AT&T PUB 41009, a technical reference on the evaluation of transmission impairment measurement equipment. The instrument chosen for ASPEN (see Table II) struck the compromise among performance, cost, and availability that was most appropriate for the EOCS. The

Table II—ASPEN remote test unit components

Transmission impairment measuring set
Hekimian Laboratories Inc. Model 3701 Communications Test System with EIA RS232C Remote Control Option

Microprocessor
Colorado Data Systems 53A Smart Hardware System*
Card Complement:
1. Zilog Z80 microprocessor, two RS232 input/output ports with buffered input and output
2. Three relay cards (ten relays per card)
3. Bit error rate receiver card
4. Bit error rate transmitter card
5. Counter card (four counters per card)

Data modems
1. For host-RTU communications:
   Western Electric 212AR
2. For bit and block error rate measurements:
   Western Electric 212AR
   Racal Vadic 3450
   Western Electric 208B-L1B
   Codex 5208R

Equipment case
Environmental Container Systems, Inc. fibercase enclosure (2 ft × 2 ft × 4 ft) with shock-mounted rack (see Fig. 2).

* Trademark of Colorado Data Systems.

Fig. 2—ASPEN RTU.

RTU instruments were remotely self-checked for accuracy throughout the data-gathering period.

As is common practice, ASPEN measures bit and block error rates by transmitting and receiving a repeated pseudorandom bit stream with the BERT and BERR equipment. The data sets used in the EOCS (see Table II) were widely used in the network and readily available at the time of the study. All data sets were pretested in the laboratory to ensure that none exhibited performance aberrations absent in other sets of the same type.

ASPEN uses an extension of the error rate measurement technique

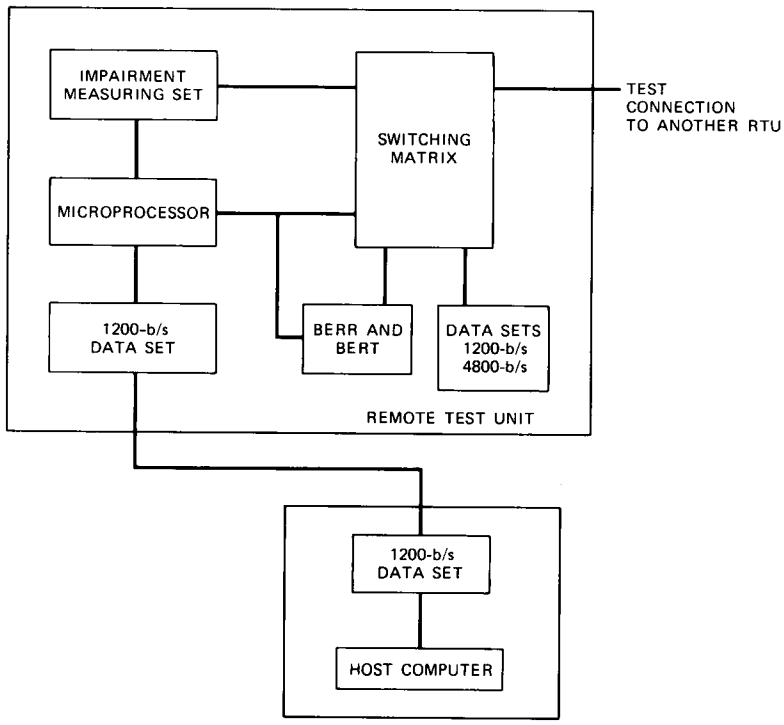

Fig. 3—Schematic representation of the ASPEN instrumentation.

to measure round trip propagation delay. As Fig. 4 shows, an error is deliberately injected into the transmitted bit stream at the near-end BERT. The bit stream containing the error is transmitted by a 1200-b/s full-duplex data set, and the error is detected by the far-end BERR. This event is used to trigger injection of an error into the far-end BERT, which is transmitted by the data sets back to the near end BERR. The elapsed time between near-end error injection at the BERT and detection at the BERR (minus a predetermined constant to allow for data set delays) is the round trip propagation delay.

2.2 Host computer hardware

The ASPEN system host computer consists of a mini- or microcomputer capable of running the *UNIX* operating system. Table III lists the host computer hardware used in the EOCS. The number of input/output (I/O) ports and amount of primary memory (i.e., random access) required depend on how many RTUs are to be controlled simultaneously. In addition, enough secondary memory (i.e., disk subsystems) is required to support the installation of a database large enough to store the data to be collected. If the RTUs are not connected

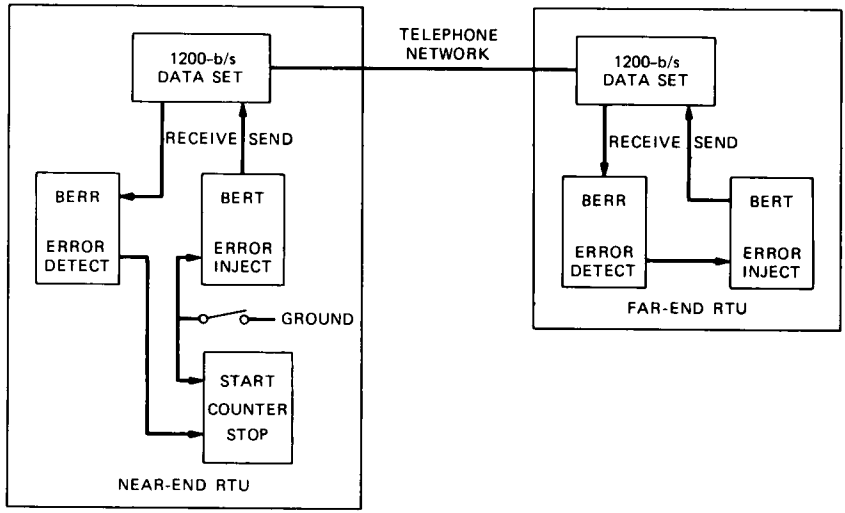

Fig. 4—Round-trip propagation delay measurement.

Table III—ASPEN/end office connection study host hardware

Components	Equipment Used
CPU	*VAX-11/780* with battery backup and *DEC* floating point accelerator
Primary memory	3.75 M-bytes random access memory
Secondary memory	Three *DEC* RMO5 removable disk drives, 256M bytes each
Tape backup	One *DEC* TU77 high-speed tape drive
I/O ports	Four *DEC* DZ11 8-channel asynchronous multiplexers, providing 32 ports, assisted by two *DEC* KMC11B auxiliary microprocessors
Dial-out	Four *DEC* DN11-DA Automatic Calling Unit (ACU) Controllers, and four WE 557A ACU-sharing arrangements.

directly to the host computer, Automatic Calling Units (ACUs) or their equivalent must be present to dial up all the RTUs.

III. ASPEN SOFTWARE SUBSYSTEMS

In the early planning stages, a decision was made to utilize the *UNIX* operating system for all aspects of the study. The *UNIX* system provided an excellent software environment for such assorted tasks as formulation and testing of statistical plans, development of ASPEN system software, collection and analysis of data, and generation of reports.

3.1 Remote test unit software subsystem

The RTU software communicates with the host computer while it simultaneously controls the RTU hardware. The RTU contains an

operating system that accepts down-loaded programs and commands from the host computer, executes subroutines in these programs, and exchanges data with the host.

The RTU software locally controls the various hardware functions by dividing the program into subroutines. Each subroutine contains an option that allows two or more subroutines to be linked.

In the EOCS the RTUs measured performance parameters of Direct Distance Dialing (DDD) connections. This involved the use of the RTUs in pairs. Because the RTU memory is limited, a separate program was needed for the originating end and the terminating end of the connection. Therefore, two RTUs testing a particular DDD connection had complementary subroutines that were initiated by "start" commands from the host computer. The subroutine timing was designed to be robust enough to accommodate a plus-or-minus 3-second error in the individual starting times of the complementary subroutines.

The complementary subroutine functions used in a typical EOCS sequence are shown in Table IV.

3.2 RTU-host interface

The interaction between the host computer and the RTU is based on the concept of a software module. A *module* is defined as a set of instructions at the host computer that causes execution of a subroutine at the RTU, and passes the RTU information necessary to execute the subroutine. There exists a one-to-one correspondence between

Table IV—Subroutine functions from typical measurement sequence

Originating RTU	Terminating RTU
Dial RTU test connection to far end	Answer
Measure analog parameters	Send test tones
Send data to host	Send data to host
Send test tones	Measure analog parameters
Send data to host	Send data to host
Dial far-end RTU reference connection	Answer
Measure envelope delay	Send test tones
Send test tones	Measure envelope delay
Send data to host	Send data to host
Connect 1200-b/s data set	Connect 1200-b/s data set
Measure propagation delay	Establish error return loop
Send data to host	—
Measure bit/block error rate	Measure bit/block error rate
Send data to host	Send data to host
Connect 4800-b/s data set	Connect 4800-b/s data set
Transmit 4800-b/s data	Measure bit/block error rate
—	Send data to host
Measure bit/block error rate	Transmit 4800-b/s data
Send data to host	—

host computer modules and RTU subroutines. The concept of modules will be covered more thoroughly in a later section.

From the point of view of the host computer, *executing* a module means conversing with the RTU microprocessor, exchanging information with it, and starting the execution of a subroutine at the RTU. Related tasks include sending and retrieving data from the RTU and, in general, executing any of the RTU built-in operating system commands. The RTU microprocessor operating system gives the host immediate feedback about the disposition of a command at the RTU. This feedback consists both of echo checks to ensure the integrity of commands transmitted to the RTU, as well as status checks to make sure that the subroutines have been executed correctly. Whenever a subroutine finishes executing, the RTU generates a status symbol, which verifies that the subroutine is completed.

The host computer also synchronizes the two RTUs engaged in testing a transmission path. By commanding execution of modules at both RTUs simultaneously, any critical timing relationships between corresponding subroutines at the different RTUs can be maintained.

Another mechanism used in the host-RTU interaction is the generation of checksums. Whenever a program is down loaded from the host computer to the RTU, the RTU generates a unique checksum, which is used by the host computer to verify the integrity of the program. In addition, transmission of data from the RTU is implemented in an error-free fashion by the use of parity checks and character counts, with retransmissions in case of errors.

The simplest way for the host and the RTU to communicate is by maintaining a communication link open between them for the duration of a measurement sequence (e.g., a dial-up connection). However, with more sophisticated programs available for the RTUs, the need for this communication link diminishes, and it suffices to poll the RTUs periodically to retrieve data from them or to reinitialize them. This significantly reduces transmission costs.

3.3 Host computer software

The remainder of this section contains detailed descriptions of the structure and the major components of the ASPEN host software. Wherever necessary, specific references to the EOCS implementation of the host software are made, but the software structure is presented in a generic form. For ease of readability, program names are shown in typewriter face and file references are presented in italics. The index n represents the number of physical RTUs in the ASPEN study. As Fig. 5 shows, there are three principal layers of ASPEN control and communication software:

1. Layer 1 contains n `call` programs, each providing a connection

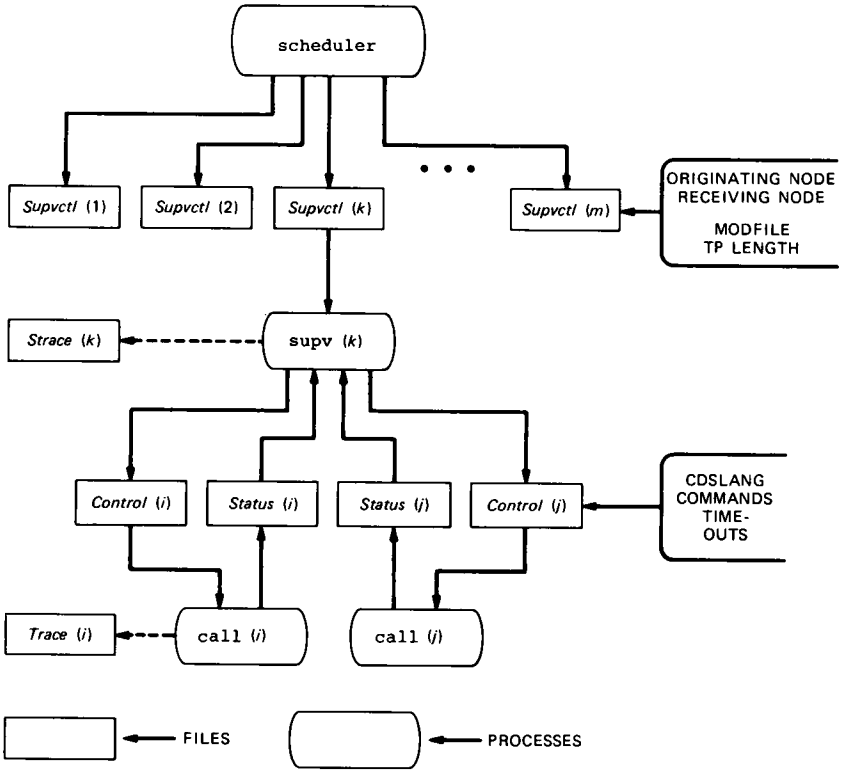

Fig. 5—ASPEN host software configuration.

from the host computer to one RTU, as well as a library of I/O functions used to interact with it.

2. Layer 2 contains $n/2$ `supv` programs, each responsible for controlling and synchronizing one pair of RTUs, by interacting with the subordinate pair of `call` programs.

3. Layer 3 consists of a single `scheduler` program, responsible for implementing the testing schedule for the study, as well as for dealing with the real-time constraints of the host computer system.

3.3.1 Call

`Call` is the base software layer and backbone of the ASPEN system. It provides communication with the RTU in a robust, error-tolerant fashion. The `call` program is based on a standard *UNIX* system utility, the `cu` (call *UNIX*) command. The `cu` command is a general-purpose software tool that enables the user to establish a telephone connection from the host computer to a remote computer system (based on the *UNIX* system or some other system). The `cu` command

provides a full-duplex environment to the user by splitting itself into two parts, one part for each direction of transmission. It manages an interactive conversation with the remote system and allows file transfers in either direction. The `call` program differs from `cu` in that it operates in half-duplex mode, requiring only a single process to run.* It is also designed specifically to converse with an RTU,† and it is not interactive, but takes its commands from a command file. A benefit of a system such as the *UNIX* system is that resources spent developing a single program can be exploited in multiple invocations of the same program. For the EOCS, 20 invocations of the `call` program ran simultaneously.

One `call` process exists for each working RTU in the ASPEN system. Upon its execution, `call` establishes a control link between the host and the RTU.‡ It then checks that the connection quality exceeds a minimum standard, and enters a quiescent state in which it waits for a signal to proceed. Each `call` process is assigned an index, which is used thereafter to identify it to various files with which it interacts. For example, process `call`(*i*)notifies the operator of its status by depositing information in the file *Trace*(*i*). Similarly, `call`(*i*) will reads its instructions from the file *Control*(*i*). There are four I/O files each `call` process interacts with: *Control*, *Data*, *Status* and *Trace*.

The *Control* file contains the instructions to be executed at any given time. In the case of the EOCS, these individual instructions are written in a format denoted CDSLANG, or CDS Language, an intermediate interpretive language based on the built-in commands of the CDS 53A hardware controller, the microprocessor-based controller used in each RTU. However, the `call` program isolates this language in such a way that other formats could be designed to control different RTUs. A collection of CDSLANG instructions, bound together to perform a discrete function, is denoted a *module*. Modules, which are stored as simple *UNIX* system files, are named according to the function they cause the RTU to perform. For example, a module named *Reset* might logically accomplish the software resetting of a component of the RTU.

---

* The *UNIX* operating system is a multitasking operating system, wherein a process is an image of a software program, loaded in core memory, with its own data and stack segments and a possibly shared text segment. Multiple invocations of a single program can result in multiple processes resident in core memory. Since the system operates in half-duplex mode, fewer processes reside in core memory, and a substantial reduction in CPU load can be achieved.

† For the EOCS, the `call` program contained code specific to the CDS 53A hardware system controller.

‡ The control link for the EOCS consisted of a telephone connection using 1200-b/s modems.

To execute a module, that is, get a `call` process to execute the instructions comprising the module, it must be copied into the *Control* file, and the `call` program must be signaled to begin. The `call` program will wake up and begin executing the CDSLANG instructions until it successfully completes all of them, or until one fails. A failure may be due to transmission difficulties between the host and the RTU, or to RTU hardware problems. The outcome of the module execution is reported to both the *Trace* file and the *Status* file. More than 50 modules were developed for the EOCS, using as building blocks more than 30 CDSLANG instructions.

A *Trace* file exists to store the current status of each `call` process whenever it changes. By utilizing an appropriate display program, an operator may monitor RTU actions.

The *Data* file is a transient template file used to deposit, on a temporary basis, data that are retrieved from the RTU. From this file data are transferred to other, more permanent, files that are time-stamped to reflect the date and time of acquisition.

The *Status* file is the I/O channel between the `call` program and the next higher layer, the `supv` program. This file holds information on the success/failure of modules treated as whole entities. No information is available as to which instruction within a module failed, merely that the module as a whole has failed. In addition, the *Status* file is used to report problems associated with the control link between the RTU and the host (e.g., a dropped telephone line). The `call` program itself is designed to take care of such situations and redial dropped connections, but the `supv` program makes sure that the second `call` program is suspended until the control link is reestablished.

3.3.2 Supv

The `supv` program constitutes the intermediate software layer in the ASPEN system. Its role is to supervise the operation of the two `call` processes beneath it, guiding them throughout the execution of a prearranged sequence of modules. The `supv` follows instructions contained in the *modfile*, a file specifying an ordered sequence of modules plus logical rules to follow in case the predefined sequence does not proceed normally. The program is able to monitor the actions of the two RTUs by analyzing the status information provided by `call` processes. The success of the `supv` may be quantified by the number of modules it guides the `call` processes through, within a specified time frame.

In the case of the EOCS, a typical modfile specified a collection of 30 modules to be executed sequentially, which would:

1. Down load programs into each of two RTUs

2. Establish a test connection between the two RTUs
3. Make a sequence of analog measurements on the test connection
4. Perform bit and block error rate measurements on the test connection
5. Transfer measurement data to the host computer
6. Reset the RTU hardware and release the test connection.

Upon invocation, the `supv` processes, one for each pair of `call` processes, would begin to execute the modules named in the modfile. One by one, a module would be copied into the *Control* files corresponding to the two `call` processes it was supervising, and the `call` processes would be awakened. `Supv` would then wait for the status of the module execution to be appended to the *Status* files. If execution was successful, `supv` would move on to the next module prescribed in the modfile. Otherwise, a set of logical rules specified in the modfile would direct `supv` through a sequence of actions ranging from repeating the module to executing an alternate set of modules, or aborting the whole modfile. To prevent endless loops, each module has a maximum allowable execution time, or "time-out" constraint. For the EOCS, eight principal modfiles controlled the measurement sequencing. New modfiles are easily developed, a flexibility which makes it possible to change the direction of a measurement study or to generate special-purpose substudies. When operating all RTUs, 10 `supv` processes supervised 20 `call` processes simultaneously, with data collection taking place on 10 test connections.

The `supv` processes take control of the `call` processes at the beginning of each new time period, defined in the next section. Each `supv` process is also assigned an index upon invocation, and only then finds out which two `call` processes it is to supervise. This information is conveyed by means of the *Supvctl* files. The *Strace* files also exist to store diagnostics from the `supv`.

Much as the success of the `supv` program is based on the outcome of individual modules, the progress of the `scheduler` program is based on the success or failure of whole modfiles. In the example above, a successful execution of the modfile would mean that the two RTUs involved successfully executed all the modules specified by the modfile on a test connection. A failed modfile execution would imply that a connection between the two particular RTUs needs to be rescheduled by the `scheduler`, since it has failed.

3.3.3 Scheduler

The `scheduler` program keeps track of the status of the entire measurement system. In contrast to `call` and `supv`, the `scheduler` program is not well defined; its implementation will vary depending on the criteria specified by a statistical sampling plan. It can range

from a simple control program, which repeatedly schedules a connection between the same set of RTUs, to a complex set of procedures, which govern a large number of RTUs, assuring that certain combinations of tests occur at certain times of the day, with special consideration for a subset of important RTUs. For the EOCS, the `scheduler` implemented a scheduling algorithm in accordance with the EOCS sampling plan. The aim was to ensure a uniform distribution of the number of measurements of each of the 380 possible RTU pair transmission paths (with 20 RTUs), while maximizing the number of RTU pair connections established during the busy hour(s) at the originating RTU location (nominally 10:00 to 11:00 a.m. and 2:00 to 3:00 p.m., local time).

In addition, the `scheduler` is responsible for making sure the study can accommodate whatever maintenance procedures exist for the host computer. The `scheduler` design provided that RTUs would not be active during a one-hour period per day scheduled a week in advance for maintenance of the host computer and stored in a file readable by the `scheduler`. By agreement, the computer servicing time occurred within a five-hour period before 5 a.m. daily.

The `scheduler` also requires robustness; it must provide adequate procedures for recovering from unexpected system downtime cleanly. For the EOCS, the `scheduler` consisted of a table- and list-driven C program that worked in combination with several smaller shell scripts. (The shell, a high-level programming language and command interpreter, is a fundamental component of the *UNIX* operating system.) For less complex studies, simpler implementations of the `scheduler` are possible entirely in shell control language.

For the EOCS, the concept of a run and a time period were developed. A *time period* consists of the time necessary for a pair of `call` processes to execute the modules specified by the modfile under the control of the `supv` process, approximately 2.5 hours for the EOCS. A *run* consists of a full cycle of time periods such that all possible RTU pair combinations are tested. For the EOCS a run was 38 time periods with 10 simultaneous RTU-RTU connections per time period. The `scheduler` used the beginning of each time period to resynchronize the scheduling algorithm, compute the connections for the next time period, manipulate lists and tables necessary to implement the busy-hour requirement of the sampling algorithm, and deal with RTUs that were out of service. As soon as the list of connections for the next time period was available, the `scheduler` would deposit the information into each of $n/2$ files, labeled *Supvctl*(1) through *Supvctl*(m), where n is the number of RTUs in the study and $m = n/2$. These files constitute the I/O channel to the intermediate `supv` level described above.

IV. END OFFICE CONNECTION STUDY SAMPLING PLAN

As we noted in the Introduction, the use of ASPEN technology has engendered new statistical problems involving sampling and data analysis. In this section we examine sampling issues somewhat generally. For the EOCS, sampling involves two components: choice of RTU locations and scheduling of calls in time. They are treated separately in this paper since they involve spatial and temporal considerations, respectively. Furthermore, the preselection method and the scheduling algorithm used to resolve these issues are general tools that may be used separately in other applications. Data analysis issues are examined in the companion paper.[12]

The problem involving selection of RTU locations arises because sampling methods required to support the new measurement methodology do not fit the classical sampling context. In the classical context experimental units are items on which measurements are made, sampling units are items that could be included in the sample, and sampling and experimental units are the same. For the EOCS, experimental units are connections, defined by ordered pairs of end office switching machines. The sampling units are end office buildings and they are *not the same* as experimental units. The situation was further complicated since study goals required adequate representation of certain strata in the sample, and stratification variables are defined in terms of both experimental and sampling units. For example, strata defined by airline mileage (a property of a pair of end office switching machines and hence defined in terms of experimental units) and by originating switch type (defined in terms of sampling units) were required.

A new method of sampling, herein called the preselection method, provides a general method for sampling networks under representation constraints. The method, an extension of the method of snowball sampling,[13] is the subject of Section 4.1.

As Section 4.2 describes, the scheduling problem involves determining when to place calls between the various RTU pairs. An algorithm was developed that provides two outputs: a synchronous, clocked schedule of RTU pairs to be in conversation at any time, and a list of the end office switches to which RTUs are to be connected. (Where end offices contained more than one type of switch, the EOCS sampling plan specified connections using each switch type.) It also provides for stratification of calls by busy versus nonbusy hour.

4.1 Remote test unit location sampling

With 20 RTUs allocated to the EOCS* at least 380 basic experi-

* Examination of data from Ref. 7 suggests that one deployment of 20 RTUs would be sufficient to achieve the goals of the study.

Table V—Sample representation requirements for the end office connection study sampling plan

Dimension	Stratum	Requirement
Mileage	0–360 miles*	At least 50 RTU pairs
	360–720 miles	At least 80 RTU pairs
	720–1320 miles	At least 100 RTU pairs
	1320 + miles	At least 110 RTU pairs
Originating switch type	*ESS* switching equipment	At least 6 switches
	Crossbar	At least 6 switches
	Step-by-step	At least 6 switches, 2 of which are Community Dial Offices (CDOs)
Facility	Satellite	At least 12 RTU pairs
	Terrestrial	Remainder
Region	Long Lines region†	3 RTUs per Long Lines region plus 2 in New York City

* Airline miles

† At the time of the EOCS sampling, Long Lines (now AT&T Communications) was divided into six administrative regions.

mental units are defined by originating-terminating pairs.* These experimental units cannot correspond to sampling units since obtaining a sample of 380 experimental units by selecting 380 office pairs would lead to the use of (up to) 760 RTUs. Thus, sampling units must be buildings that house end office switches, and the experimental and sampling units do not correspond.

As we noted, the sample was required to adequately represent various strata. Strata definitions and representation requirements were derived using 1969/70 Connection Survey data.[7] The most important strata variables are airline mileage and originating switch type. Airline mileage serves as an easily determined surrogate for route mileage, which is known to affect many parameters and is not easy to measure. Impulse noise is associated with step-by-step switches. In addition, strata were defined by geographic regions and by whether telephone connections between two locations could be carried by a satellite. Strata definitions and representation requirements are given in Table V.

The preselection method, used in drawing a sample that meets the representation requirements noted above, is diagramed in Fig. 6. It features three steps:

1. Creation of clusters of experimental units and definition of selection probabilities.

2. The preselection step, the output of which is a set of clusters, guaranteed to produce an acceptable sample.

* Many Bell operating company buildings house more than one switch, and the RTUs were designed for connection to up to three separate switches. Since experimental units are defined using switches rather than buildings, there were more than 380 such units in the EOCS. Multiple-switch buildings play a key role in sampling, as we will discuss.

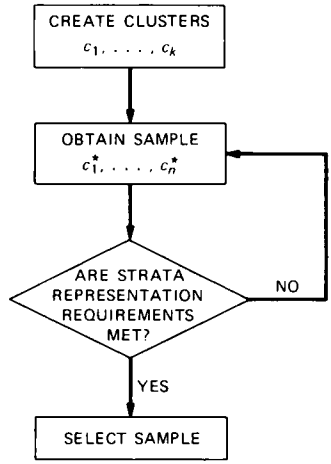

Fig. 6—The preselection method.

3. The selection step, in which the actual sample of experimental units is drawn.

The preselection method is quite general. Creation of clusters admits great latitude. Selection probabilities need not be equal, and at each step sampling can be with or without replacement, with or without stratification, and so on.

For the EOCS, clusters were created through definition of subregions, areas of about 10,000 square miles each, in each Long Lines (now AT&T Communications) administrative region. Clusters included all Bell System buildings within a given subregion. Selection probabilities were based on the number of subscriber lines served by a building and the type(s) of switches it housed. Three weights, one per switch type, were defined for each building:

$$w(b, s) = \begin{cases} T_b = \text{number of lines served by building } b, \text{ if switch type } s \text{ is housed in building } b, \\ 0, \text{ if } s \text{ is not in } b, \text{ where } s = ESS, \text{ crossbar, step-by-step switching equipment.} \end{cases}$$

Selection probabilities were defined through normalization of weights throughout. They favor inclusion of buildings that house more than one switch type, because such buildings lead to comparisons of switch type performance that are not confounded with other variables. Subregion weights were calculated by summing over all buildings located within the subregion.

In preselection, subregions were sampled according to the following algorithm:

1. Twenty artificial units, six each labeled *ESS*, crossbar, and step switching equipment, and two unlabeled, were defined. Starting in the Northeast, an artificial unit was drawn without replacement and a subregion sampled, again without replacement, using the subregion selection probabilities that correspond to the (switch) label of the artificial unit.
2. Sampling proceeded in this manner until the appropriate number of subregions had been sampled in each region. There was one exception to this rule: The two subregions in which Chicago and Orlando are located were included in the sample with probability one to help meet the satellite representation criterion (Table V). The label associated with the first artificial unit drawn in the Midwest and Southern regions was assigned to the Chicago and Orlando subregions, respectively.
3. Except for the mileage criteria, the sampling scheme ensures that all representation requirements are met. These criteria were checked by assuming that the selected buildings would be located at the center of their respective subregions and by computing the resultant distances. If a sample of subregions did not satisfy these representation criteria, that sample would be discarded and steps 1 through 3 repeated.

Once the sample of subregions was accepted by the preselection step, the selection step was used to produce an actual sample of buildings. Building selection probabilities corresponding to the switch types assigned each region were used in this step.

Locations of sampled buildings are shown in Fig. 7.

4.2 Test connection scheduling algorithm

When RTUs are in place and operating, the ASPEN system is capable of collecting data nearly continuously. Calls placed between RTUs simulate calls that could be placed by customers; they are not sampled from that population, however. This has implications for data analysis.

For the EOCS the following goals were adopted:

1. Minimize measurement equipment idle time
2. Maximize the number of busy-hour calls placed and ensure that both busy- and nonbusy-hour calls are made between all RTU pairs
3. Place and receive calls through all switches connected to each RTU with equal frequency
4. Be robust to RTU failure.

The scheduling algorithm described herein works whenever n, the number of RTUs, is even, though it is described here as used in the EOCS ($n = 20$).

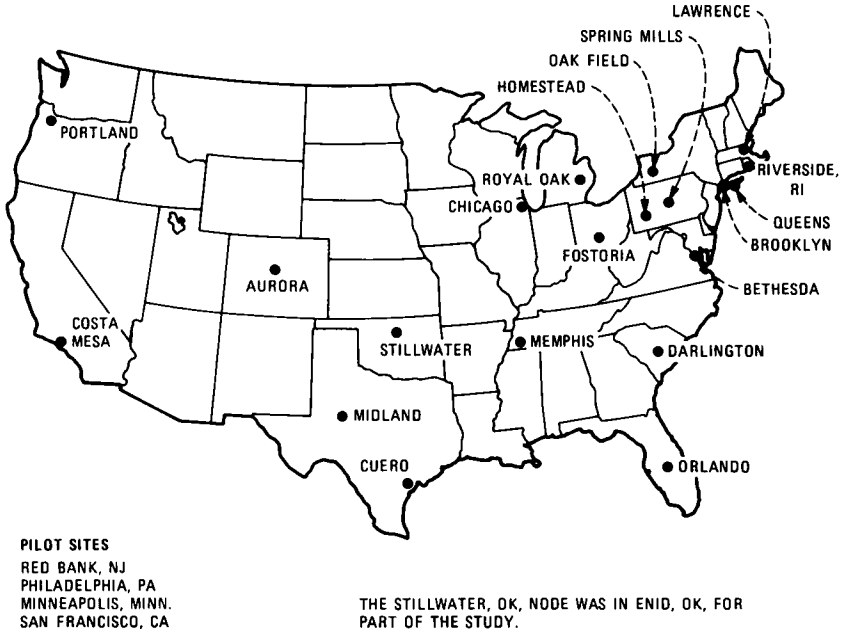

Fig. 7—Measurement equipment locations for end office connection study.

The algorithm specifies a synchronous, clocked schedule. That is, at the beginning of a measurement sequence, ten connections (of the 380 possible) are established and a predetermined span of time (the time period) is allotted for completion of test sequences. Thirty-eight time periods define a run, during which a call on each ordered RTU pair is established once and only once. The basic elements of the algorithm are:

1. The RTU pair table—This table consists of all 380 possible ordered pairs of 20 symbols, grouped in 38 sets, with each symbol used once and only once in each set. When the scheduling algorithm is implemented, symbols will be assigned to RTUs and sets to time periods. Construction of this table is discussed below.

2. The busy-hour table—This table gives the busy hour for each RTU and, as the experiment proceeds, the number of busy-hour connections established to date.

3. The switch definition table—This table specifies the switches through which calls are established for each RTU pair based on the run number.

At the start of each run:

1. RTUs are randomly assigned to symbols in the RTU pair table to provide 38 sets of 10 RTU pairs each.

2. Each of the 38 sets is assigned a time period. First, when the busy-hour table is being used, sets associated with RTU pairs on which few busy-hour calls have been placed are preferentially assigned time periods that correspond to the busy-hour for those pairs. Eventually, such assignments can no longer be made and the remaining sets are randomly assigned time periods.

3. Finally, the switches through which RTU pairs are to make each connection are determined by consulting the switch definition table.

Thus far, the algorithm provides a complete schedule of calls, but provides no method of protecting itself from intermittent RTU failure. To do so, the algorithm specifies that at the start of each time period lists be made of out-of-service RTUs and idled RTUs (mates of RTUs out of service). Connections that cannot be established, including designated switch pairs, are then added to a "calls missed" list. Finally, idled RTUs are used to establish previously missed connections.

The *RTU pair table* is the cornerstone of the scheduling algorithm since it establishes an efficient mechanism for specifying study test connections among the various RTU pairs. Creation of such a table is nontrivial, and so we describe the method used for the EOCS. This method starts with a Latin square,[14] but is not completely general in that Latin squares exist that do not yield an RTU pair table. The authors speculate that restriction to a special class of Latin squares would lead to a method of full generality.

The method is described below for the general n-RTU case, n even, and is illustrated using $n = 4$ throughout.

1. Start with a Latin square of size n, using symbols 0, ..., n-1. Attach row and column letters in such a way that zeroes are (i, i) entries.

EXAMPLE: For a four-RTU experiment, there are twelve RTU pairs and two pairs can be connected in each time period. Six time periods will be required to connect all RTUs in all possible pair combinations. An augmented Latin square is:

	a	b	c	d
a	0	2	3	1
d	1	3	2	0
b	2	0	1	3
c	3	1	0	2

2. Group the (row, column) ordered pairs that correspond to each nonzero entry of the Latin square.

EXAMPLE (continued): For the entry 1 we obtain:

(a, d), (d, a), (b, c), and (c, b).

3. Split each group into two subgroups such that each symbol

appears in each subgroup once and only once. This yields the RTU pair table.

EXAMPLE (continued): For the group above we obtain:

$$\{(a, d), (b, c)\} \text{ and } \{(d, a), (c, b)\}.$$

Each bracketed set corresponds to RTU pairs over which measurements will be made during one of the six time periods in each run of the four-RTU experiment. Note that these groups need not be unique.

V. ACKNOWLEDGMENTS

Many individuals contributed to the success of both the ASPEN Data Acquisition System and the End-Office Connection Study. We would especially like to thank Alan Furtado and Jon Palmer for their tireless efforts in installing and maintaining the remote test units, Jay MacMaster for his elegant physical design and construction of the RTUs, Louis Iacona for his vigilant management of the Study database, Karel Ehrlich for his efficient design of the ASPEN scheduler software, and Bob Sturges for his vital assistance in design and construction of RTU support circuitry.

Michele Carey, Han-Tee Chen, Fred Descloux, Jim Ingle, and Kun Park, the authors of the companion paper, provided the screening algorithms and rapid analyses of pilot and full study data that permitted the high level of data integrity achieved in this study. We were especially fortunate to have Jim Ingle take charge of the testing and acceptance of transmission impairment measuring sets for ASPEN, an invaluable contribution to a critical phase of the project.

We also wish to thank Paul Auer for his help in shaping the ASPEN system and in designing the transient parameter measurement procedures, Blan Godfrey for his far-sighted support of the original ASPEN concept, and Pete Lopiparo for his steady confidence in and support of the ASPEN project team. We gratefully acknowledge the cooperation and contributions of Bell operating company central office personnel and the many valuable consultations with our colleagues at AT&T Bell Laboratories. This was in every sense a Bell System study.

REFERENCES

1. A. A. Alexander, R. M. Gryb, and D. W. Nast, "Capabilities of the Telephone Network for Data Transmission," B.S.T.J., *39*, No. 3 (May 1960), pp. 431–76.
2. I. Nasell, "The 1962 Survey of Noise and Loss on Toll Connections," B.S.T.J., *43*, No. 2 (March 1964), pp. 697–718.
3. J. H. Fennick and I. Nasell, "The 1963 Survey of Impulse Noise on Bell System Carrier Facilities," IEEE Trans. Commun. Technology, *COM-14*, No. 4 (August 1966), pp. 520–4.
4. I. Nasell, "Some Transmission Characteristics of Bell System Toll Connections," B.S.T.J., *47*, No. 6 (July–August 1968), pp. 1001–18.
5. I. Nasell, C. R. Ellison, Jr., and R. Holmstrom, "The Transmission Performance of Bell System Intertoll Trunks," B.S.T.J., *47*, No. 8 (October 1968), pp. 1561–1613.

6. P. A. Gresh, "Physical and Transmission Characteristics of Customer Loop Plant," B.S.T.J., *48,* No. 10 (December 1969), pp. 3337–86.
7. F. P. Duffy and T. W. Thatcher, Jr., "Analog Transmission Performance on the Switched Telecommunications Network," B.S.T.J., *50,* No. 4 (April 1971), pp. 1311–47.
8. M. D. Balkovic, H. W. Klancer, S. W. Klare, and W. G. McGruther, "High-Speed Voiceband Data Transmission Performance on the Switched Telecommunications Network," B.S.T.J., *50,* No. 4 (April 1971), pp. 1349–84.
9. H. C. Fleming and R. M. Hutchinson, Jr., "Low-Speed Voiceband Data Transmission Performance on the Switched Telecommunications Network," B.S.T.J., *50,* No. 4 (April 1971), pp. 1385–1405.
10. J. E. Kessler, "The Transmission Performance of Bell System Toll Connecting Trunks," B.S.T.J., *50,* No. 8 (October 1971), pp. 2741–77.
11. F. P. Duffy, G. K. McNees, I. Nasell, and T. W. Thatcher, Jr., "Echo Performance of Toll Telephone Connections in the United States," B.S.T.J., *54,* No. 2 (February 1975), pp. 209–43.
12. M. B. Carey, H.-T. Chen, A. D. Descloux, J. F. Ingle, and K. I. Park, "1982/83 End-Office Connection Study, Analog and Voiceband Data Performance on the Public Switched Network," AT&T Bell Lab. Tech. J., this issue.
13. L. A. Goodman, "Snowball Sampling," Ann. Math. Statist., *32* (March 1961), pp. 148–70.
14. O. Kempthorne, *The Design and Analysis of Experiments,* Huntington, NY: Robert E. Krieger Publishing Co., Inc. 1952.

AUTHORS

John Healy, B.S. (Mathematics), 1971, St. Bonaventure University; Ph.D. (Mathematical Statistics), 1976, Purdue University, AT&T Bell Laboratories, 1976–1983. From 1976 through 1983 at Bell Laboratories, he held various technical and supervisory positions in the areas of statistics, reliability, and network measurements. His research has been published in the Reliability and Maintainability Symposium Proceedings, Bell System Technical Journal, the Journal of the American Statistical Association, Psychometrika, and the Journal of Multivariate Analysis. He is currently the District Manager of the Reliability and Maintainability Methods District in the Quality Assurance Technology Center at Bell Communications Research.

Maurice Lampell, B.A. (Physics) and B.S. (Electrical Engineering), 1979, Boston University; M.S. (Electrical Engineering), 1980, Stanford University, AT&T Bell Laboratories, 1980–1983. Present affiliation Bell Communications Research, Inc. Mr. Lampell has been involved with the development of automated data acquisition systems running under the *UNIX* operating system. He is currently working on network characterization planning, and is developing a framework for the transfer of data acquisition expertise to the Bell operating companies.

David G. Leeper, B.S. (Electrical Engineering), 1969, Washington University; M. Eng. (Electrical Engineering), 1970, Cornell University; Ph.D. (Electrical Engineering), 1977, University of Pennsylvania. AT&T Bell Laboratories, 1969–1983. Present affiliation Bell Communications Research, Inc. From 1969 to 1978 Mr. Leeper was involved in design, engineering, and performance measurements on new and existing digital transmission systems. From 1978 to 1983 he supervised design and construction of computerized performance measurement systems and their application in network performance field measurement studies. He is currently Division Manager, Exchange Network Services Planning.

Thomas C. Redman, B.S. (Mathematics) 1976, Northwestern University; M.S., Ph.D. (Statistics), Florida State University, in 1978 and 1980, respectively; AT&T Bell Laboratories, 1980–1983. Present affiliation Bell Communications Research, Inc. Mr. Redman has worked on the planning, execution, and statistical aspects of (telephone) network characterization studies. He is now with Bell Communications Research and continues to work on similar problems. Member, ASA.

Edward J. Vlacich, B.S. (Electrical Engineering), 1964, Fairleigh Dickinson University; M.S. (Statistics), 1970, Rutgers University; AT&T Bell Laboratories, 1961–1983. Present affiliation Bell Communications Research, Inc. Mr. Vlacich has been involved in the field instrumentation systems used for characterizations of the telephone network. He helped design ASPEN, which is used to characterize transmission impairments associated with data transmission. He is currently working to incorporate state-of-the-art measurement techniques into new survey equipment.