PERFORMANCE ANALYSIS OF SYSTEM 75

Robert H. Astmann and Joseph S. Kaufman

Robert H. Astmann *is a supervisor in the Data Network Design Department of AT&T Bell Laboratories in Middletown, New Jersey. Mr. Astmann joined AT&T in 1973 and is currently responsible for software development for the Information Systems Network (ISN). Previously, he worked on performance measurement and software development for the System 75 project. Mr. Astmann has a B.S.E.E. from Clarkson University, Potsdam, New York, and an M.S.E.E. from Purdue University.* ***Joseph S. Kaufman*** *is a distinguished member of technical staff in the Performance Analysis Department of Bell Laboratories in Holmdel, New Jersey. Mr. Kaufman joined the company in 1973 and is involved in performance analysis studies on a variety of real-time systems. He has a B.E.E. from Pratt*
(continued on page 120)

Quantifying the real-time performance of a PBX (private branch exchange) such as the AT&T System 75 is crucial to its success. However, waiting until a prototype can be evaluated under realistic conditions may be too risky. Constructing predictive models of performance while a system is still in the development stage has the advantage of raising performance concerns early, when corrective design steps can be undertaken more easily.

This paper describes a performance analysis study of System 75 that began in the system design stage. We describe an analytic model that was used to determine stimulus response times of interest. Early results of the analytic modeling work were used to motivate architectural enhancements that were made while System 75 was still in the development stage. Most significantly, the analytic model was a factor in the decision to increase the advertised call capacity of System 75, which allowed the product to address the needs of a much larger market.

Along with the modeling and analysis work, we began a systematic program of laboratory measurements to obtain an accurate workload characterization. This crucial measurement activity and the manner in which it was integrated with the evolving modeling and analysis effort are also described.

Introduction

The System 75 is a digital voice/data communications system that was originally targeted to handle 400 voice stations, 100 data ter-

minals, and 100 trunks. It provides a rich set of voice, data, and networking features, CRT-based administration access, and sophisticated maintenance capabilities. System 75 was introduced early in 1984 and continues to evolve as a product. (See Panel 1 for a list of terms and acronyms used in this paper.) The performance measurement/modeling methodology outlined in this paper has been a key element in the evolution of System 75 and has resulted in significantly larger capacities than those required when the product was originally introduced.

The effort to understand the performance implications of design decisions was begun very early in the System 75 development cycle. One of the initial objectives of this study was to identify architectural refinements that could or should be made to improve performance. Another important objective was to put in place a performance tracking and reporting methodology. By providing constant feedback to the software developers, we hoped to build an awareness of performance issues. Finally, we wanted to develop an analytic model that could be used to determine the call capacity of the system, given the performance criteria established for the product.

Panel 1. Terms and Acronyms in This Paper

angel	a peripheral board control processor
archangel	the processor that mediates communications between the angels and the SPE
BHCR	busy hour call rate
CRT	cathode-ray tube
FCFS	first-come, first-served
GAMUT	a UNIX®-system-based testing tool to monitor and simulate message activity
MFET	multifunction electronic telephone
Oryx/Pecos	a message-based operating system that supports real-time applications
PBX	private branch exchange
RAM	random-access memory
ringtrip	when the called party goes off-hook
Sc_dr	switch control driver
SPE	switch processing element
spigot	a system call compiled into software for time-stamping of events in real time
TDM	time-division multiplexing

System 75 Architecture

The design of System 75, from both hardware and software points of view, emphasizes flexibility and extensibility. The January 1985 issue of the *AT&T Technical Journal* is dedicated to System 75 and discusses the architecture of the system in much greater detail than the summary that follows.[1]

Hardware Architecture. The System 75 hardware architecture is characterized by the use of "intelligent," microprocessor-controlled peripheral interface boards (e.g., trunk interfaces, voice/data terminal interfaces) that communicate with a central controller processor, referred to as the switch processing element (SPE). A time-division-multiplexed (TDM) bus is used for port-to-port voice and data connections, as well as for communications between the peripheral board processors (known as "angels") and the SPE. The mediation of communications between the angels and SPE is the job of another processor, called the "archangel."

Software Architecture. The System 75 software is written in C language and is organized around a real-time, event-driven operating system called Oryx/Pecos. Oryx/Pecos provides a prioritized process structure that uses messages for interprocess communication and synchronization. Call processing, database administration, and maintenance software all coexist in the Oryx/Pecos environment. The call-processing software architecture is characterized by a layered structure that models the logical decomposition of a call and its feature interactions. The lower layer is called the "resource layer," and is made up of the processes that manage the physical resources of the system (e.g., trunks, terminals, network time slots, and tone sources). The upper layer, the "application layer," does call-event sequencing and handles high-level feature interactions.

Call-event stimuli are detected initially by the angel processors, and are sent uplink to the SPE by way of the archangel, which places them in a first-come-first-served (FCFS) queue (called the "uplink buffer"). A polling scheme (described in the section on the uplink buffer delay model) is used to retrieve stimuli from the uplink buffer and transfer them to the SPE where call processing begins.

Workload Characterization

Real-time measurements of System 75 software were made using the "spigot" system.

Spigot Measurements of Stimuli. A spigot is a system call compiled into the software that outputs data at high speed. A special interface board collects and time stamps each spigot output record to the nearest 10 microseconds (μs); a minicomputer stores the spigot data on disk for the duration of an experiment (e.g., one complete call). These raw data are transferred at high speed to a VAX™ computer for data reduction and report generation. (VAX is a trademark of Digital Equipment Corporation.)

A limited number of "official" spigots were compiled into the software so that certain critical measurements could be repeated in a consistent manner as feature development proceeded. For example, spigots were put into the code at the points where the processing of all call-related stimuli begins and ends. As a result, a simple report can be generated for each type of call, giving the processing time of each stimulus making up the call, as well as the total call-handling time. In addition, because spigots were also placed in the Oryx/Pecos kernel, it was quite easy to produce a report listing the breakdown *by process* of time spent processing the call.

An environment was also set up to make it easy for individual software developers to install temporary, unofficial spigots in their code to diagnose real-time usage problems or test possible optimizations.

In addition to the single-user call-measurement approach described above, a capability to monitor the system while it is subjected to realistic calling loads was also developed. An existing UNIX®-system-based testing system called GAMUT[1] was configured to generate call stimuli at rates comparable to expected busy hour loads, while the spigot system was used to measure and report statistics of stimulus queueing delays.

Monitoring Measurements During Development. As feature software development proceeded, a new internal release package was produced every one to two months. After a new release was considered stable (i.e., basic calls could be made), a set of benchmark call measurements was made, condensed using the spigot data-reduction tools, and reported in a performance "newsletter." These activities were quite effective in encouraging developers to make optimizations in their code on an ongoing basis (as opposed to only at the end of the project). One significant result of this procedure was that the processing times of the benchmark calls being tracked decreased by an average of about 40 percent over the development cycle, even though a very large number of calling features were integrated into the software during this time.

Standard-Call-Mix Model. Different call types place significantly different real-time demands on the SPE. Therefore, for the purposes of tracking performance and obtaining preliminary capacity estimates, it was necessary to define a *standard call mix* that consisted of:

- m distinct call types, $c_1, \ldots, c_m$
- The probability, p_i, that call type c_i is used
- The number of hardware stimuli, n_i, associated with call type c_i
- The SPE processing time for each of the $\sum_{i=1}^{m} n_i$ distinct stimuli.

The m call types chosen were "bread-and-butter" type calls, widely used in many different environments (various types of outgoing, incoming, and intraswitch calls).

If t_i denotes the sum of the stimulus processing times associated with call type c_i (e.g., t_i is the total SPE real time expended on a type c_i call), then

$$\bar{t} = \sum_{i=1}^{m} p_i t_i \qquad (1)$$

is the average call processing time associated with the standard call mix. Moreover, if one assumes a total calling rate of λ_c calls per second, then

$$\rho = \lambda_c \bar{t} \qquad (2)$$

is the SPE utilization (average fraction of time the SPE is busy processing calls) associated with calling rate λ_c (e.g., there are $\lambda_c p_i$ calls of type c_i, each of which consumes t_i seconds of real time). It is important to note that the SPE utilization ρ as a function of the "load" λ_c (specified in calls

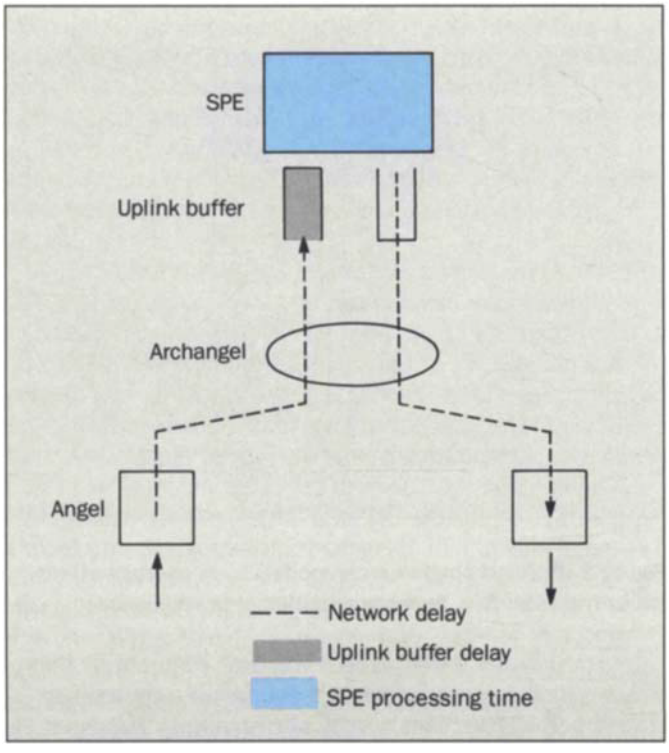

Figure 1. Components of stimulus response time.

per second) is known once the workload characterization (contained in the above list) is specified.

Performance Measures

Although a "call" is often thought of as the basic unit being processed by a switch, the switch actually processes the stimuli associated with each call (e.g., off-hook, digits, ringtrip). (Ringtrip refers to the called party going off-hook.) The delay in generating appropriate responses to stimuli must typically be very small—for example, the delay associated with dial-tone in response to off-hook, or speech path cut-through in response to ringtrip. Although differences may exist in the precise definition of a response-time criterion, there is little question that a stringent criterion on the tail of the stimulus-response-time distribution is far more meaningful than a simple, average response-time criterion. Thus, at one time, for example, the 99th percentile dial-tone delay criterion was 3 seconds.

If τ_d denotes the dial-tone delay, then this criterion on the tail of the dial-tone response-time distribution can be written as:

$$P(\tau_d > 3 \text{ seconds}) \leq 0.01$$

Given the various response-time criteria, the real-time call capacity of the switch, $\lambda_{c,\max}$ (calls per second), may be defined as the maximum calling rate for which none of the criteria is violated. The busy hour call rate (BHCR) is a frequently cited call capacity measure. This is simply $\lambda_{c,\max}$ scaled to an hourly measure:

$$BHCR = 3600 \cdot \lambda_{c,\max}$$

Note that call capacity determination has three necessary ingredients:

1. A workload characterization model that defines a standard call and specifies how much real-time work a "standard" call brings into the SPE.
2. A set of performance criteria on stimulus response times.
3. A model that can predict stimulus response times accurately.

In the absence of one or more of these essential ingredients, simple-minded approaches (which mask the lack of a detailed analysis) are often found in the literature. Thus, the busy hour call rate may be rewritten as:

$$BHCR = \frac{3600 \cdot \rho_{\max}}{\bar{t}}$$

where $\rho_{\max}$ corresponds to $\lambda_{c,\max}$ in equation (2) and $\bar{t}$ is defined by equation (1) ($\bar{t}$ is in seconds). In the approaches mentioned above, $\rho_{\max}$ is often chosen to be a reasonable occupancy (for example, $\rho_{\max} = 0.50$). Of course, guessing $\rho_{\max}$ is equivalent to guessing the call capacity $\lambda_{c,\max}$ and must inevitably result in grossly underestimating or overestimating the real-time capacity of the switch. In addi-

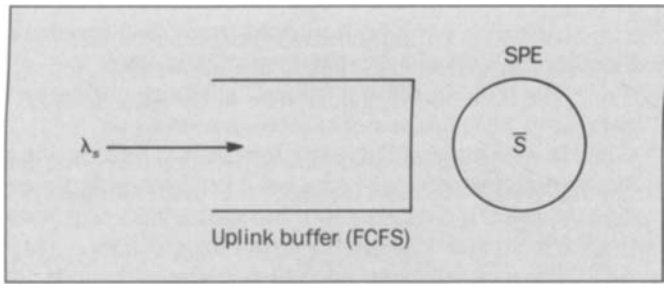

Figure 2. Simplistic single server model. λ_s = average stimulus arrival rate; $\bar{S}$ = average stimulus processing time. FCFS stands for first-come, first-served.

tion, $\bar{t}$ may not reflect a realistic call mix and may be far too small if, for example, it reflects a large proportion of call attempts (calls that are not necessarily completed). In such cases, the advertised BHCR may be highly exaggerated.

Analytic Modeling

Call-processing stimulus-response times in the System 75 architecture can be viewed as having three components: network delay, uplink buffer delay, and SPE processing time as shown in Figure 1. Network delay consists of (1) network uplink delay—the time between stimulus generation and its subsequent arrival to the uplink buffer and (2) network downlink delay—the time between stimulus-response message arrival to the downlink buffer and its subsequent processing completion.

Network delay is random, essentially independent of call arrival rate, and somewhat dependent on the angel type. For the purposes of this study, we assume a multifunction, electronic-telephone-type angel (or MFET angel). Both the uplink and downlink buffers reside in a dual-ported RAM (random-access memory), which is the medium of communications between the SPE and archangel.

Uplink buffer delay is the time between stimulus arrival to the uplink buffer and the subsequent beginning of its processing by the SPE. Uplink buffer delay is random and quite sensitive to the call arrival rate, accounting as it does for SPE contention.

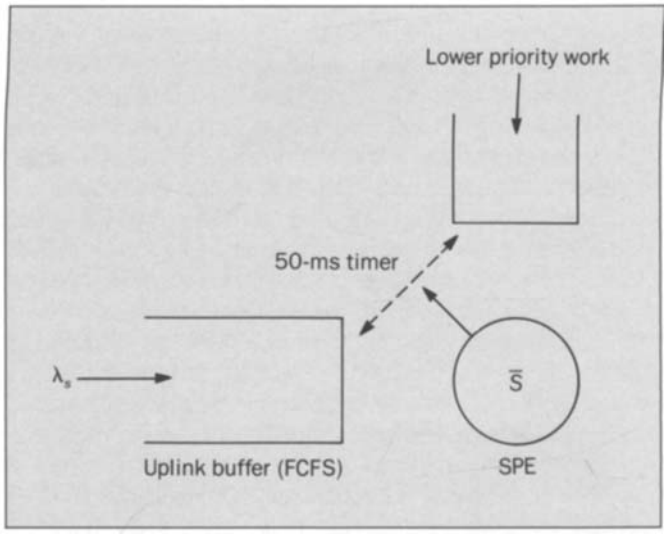

Figure 3. Refined single server model. λ_s = average stimulus arrival rate; $\bar{S}$ = average stimulus processing time.

SPE processing time is the time required by the SPE (including operating system overhead) to process a stimulus. Because many stimuli elicit multiple responses, processing time is response-dependent and, for a particular response (e.g., speech path closed), is considered to be complete when the particular response message arrives at the downlink buffer. SPE processing time for a given stimulus (and, in the case of multiple responses, for a given response) is assumed to be a constant. The dependency of stimulus processing time on factors such as the number of calls in progress is negligible.

In the following three sections, we explain the analytic models that describe the three components of stimulus response time.

Network Delay Model (MFET Angel). The uplink component of network delay is dominated by the 25-millisecond (ms) scan rate of the angel, and the approximately 3-ms (no load) polling cycle of the archangel. The downlink delay for network update messages is dominated by the archangel polling rate and the time spent waiting for the angel message handler to run. For lamp and ringer update messages, downlink delay is dominated by the angel 25-ms

scan rate. The only component of network delay that is potentially traffic-dependent is the archangel polling cycle. However, even under busy hour loads, the archangel is lightly loaded and, hence, the total network delay, although random, is essentially traffic-independent. In Appendix A, the distribution of this network delay component of stimulus response time is computed and discussed in more detail.

Uplink Buffer Delay Model. Call stimuli are enqueued in the uplink buffer by the archangel. The stimuli in the uplink buffer are retrieved by the switch control driver (Sc_dr) in a first-come-first-served (FCFS) fashion and sent to the service dispatcher process where call processing begins. For the purposes of modeling, call stimuli may be considered to be processed in a single-thread fashion (i.e., one stimulus will be processed to completion by the SPE before the processing of another begins). The single-thread processing of stimuli suggests that uplink buffer delay can be modeled by a single server queue with FCFS discipline, as shown in Figure 2.

When all call processing is completed, the SPE works on a variety of lower priority work, such as maintenance and administrative tasks. This non-call-processing work in turn will be pre-empted by a 50-ms asynchronous clock that gives control to the Sc_dr. If the Sc_dr finds that a call-processing stimulus is present in the uplink buffer, call processing work will resume at such 50-ms interrupts. Thus, the basic single-server delay model in Figure 2 is a bit too simplistic. A more realistic model, shown in Figure 3, has the effect of the 50-ms timer folded in. In terms of queueing theory, a call-processing busy period cannot begin with the arrival of a call-processing stimulus, but, rather, must wait for the occurrence of the next 50-ms (asynchronous) clock tick. The 50-ms timer just described is actually derived from a 25-ms clock, which interrupts both call processing and background work and allows system-level functions to be performed. These system-level functions consume a small amount of real time, regardless of the real load, and are referred to as no-load system overhead.

The stochastic assumptions made in analyzing the refined model for uplink buffer delay (shown in Figure 3) appear to be quite reasonable. The assumptions made

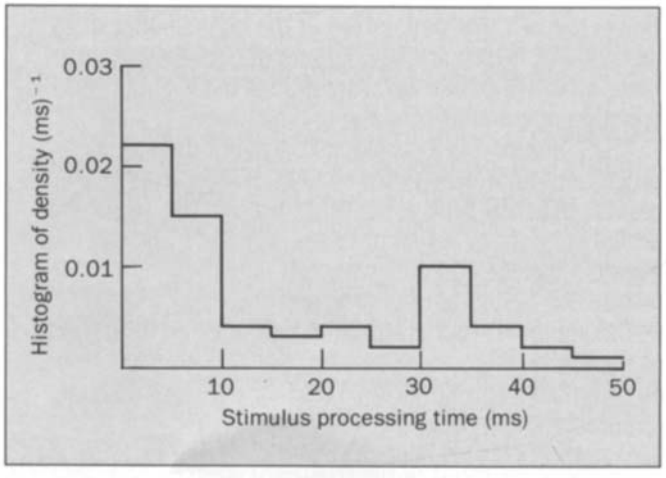

Figure 4. Histogram of stimulus-processing-time density.

about the service time (e.g., distribution of SPE stimulus processing time) are discussed in detail in the next section and are extrapolations of laboratory measurements. The call-stimulus arrival process (to the uplink buffer) is assumed to be Poisson and, hence, is characterized by a single arrival-rate parameter λ_s (stimuli per second). Because two successive stimuli will often be associated with different calls (especially at busy hour call rates, where modeling accuracy is more important), this Poisson assumption is satisfactory. This modeling assumption has been studied in some detail in Reference 2. Analysis of the delay distribution for the model shown in Figure 3 is described below in the section on analysis techniques.

Processing Time. The SPE processing time for each stimulus response pair (e.g., ringtrip—speech path closed) is essentially a constant that can be measured in the laboratory. Thus, conceptually, this component of the response time is a given. The *distribution* of stimulus processing times (rather than any given stimulus-response processing time) is an essential input to the uplink buffer delay model. Although conceptually it also is a given, significant uncertainty always exists about the nature of this distribution. This uncertainty is due to the fact that many call types and features (and therefore stimuli) are exercised differently in different environments. Further compounding this situa-

tion is the fact that percentiles of the uplink buffer delay distribution, which are key outputs of our analysis, are quite sensitive to certain assumptions made about this distribution of stimulus processing time.

Figure 4 is a scaled histogram of the stimulus-processing-time density for one set of laboratory measurements. (All SPE processing times used in this paper are scaled versions of actual processing times.) The call mix in Figure 4 includes the following call types:

- Internal—Station-to-station and coverage
- Incoming—Direct inward dialing and listed directory number
- Outgoing—Direct outward dialing, with and without automatic call back.

Assumptions made about the call mix, based on prior PBX traffic studies, are:

- Incoming and outgoing calls each comprise 36 percent of the mix.
- Internal calls comprise the remaining 28 percent.
- Station-to-station and coverage each account for 50 percent of internal calls.
- Similarly, direct inward dialing and listed directory number calls each account for 50 percent of the incoming calls.
- Automatic call back accounts for only 3 percent of the direct outward dialing calls.

In a realistic call mix, with dozens of call types together with the effects of varying customer translation conditions, the "hills and valleys" evident in Figure 4 may become less exaggerated. Uncertainty as to this kind of detail in the distribution of processing times is not of great concern, however, because it has a relatively minor impact on the percentiles of the uplink buffer delay distribution.

Of far greater concern are the unknown answers to: *how many* stimuli will have processing times exceeding some number (e.g., 50 ms in Figure 4) and *how large* will such processing times be in a realistic call mix?

In Figure 5, we introduce a three-parameter family of stimulus-processing-time densities in order to parameterize our uncertainty. Thus, the parameter m_1 denotes the demarcation point for the central region within which α *percent* of all call-processing stimulus-processing times fall, and $1-\alpha$ quantifies the above question of *how many*. The largest stimulus, s_{max}, has a processing time $(r+1)\,m_1$ and hence r quantifies the question *how large*.

By examining the sensitivity of uplink buffer delay to different values of α, m_1, and r, we can calculate the impact that our inherent uncertainty may have on system performance. If $\bar{S}$ denotes the average stimulus processing time for the density in Figure 5, one can easily show that

$$\bar{S} = \frac{1}{2}\left[m_1 + (1-\alpha)\, S_{max} \right] \tag{3}$$

Analysis Techniques

In this section, we sketch the analytical techniques used in obtaining percentiles of the stimuli-response-time distributions. The major challenge is in obtaining the distribution of uplink buffer delay, which is the only traffic-dependent component of response time. Thus, in the next section, we sketch the technique used in obtaining the uplink buffer delay. We then show how the over-all stimulus-response-time distribution is obtained.

Analyzing Uplink Buffer Delay. As mentioned above, our model of the uplink buffer delay is an M/G/1 queueing system with an asynchronous timer (and FCFS discipline), as illustrated in Figure 3. In terms of queueing theory, the effect of the timer is to delay the beginning of a call processing busy period by the forward recurrence time of the timer (e.g., by the time until the next 50-ms interrupt).

Let W_τ denote the delay of a stimulus in the uplink buffer (τ denotes the period, 50 ms, of the asynchronous clock), and X the SPE processing time of a stimulus (with density as in Figure 5). Let $\alpha_\tau(s)$ and $\beta(s)$ denote the Laplace Stieltjes transforms of the distributions of W_τ and X, respectively. Also, let Z_τ denote the time until the next clock tick, measured from the arrival instant of a stimulus that finds the system empty (of all call processing work), and let $\delta(s)$ denote the Laplace Stieltjes transform of the distribution of Z_τ. Obviously, the density of Z_τ has support $[0,\tau]$. Then, a simple application of delay cycle analysis[3] shows that:

$$\alpha_\tau(s) = P_o\delta(s) + \frac{(1-P_o)(1-\rho)(1-\beta(s)\delta(s))}{E(X+Z_\tau)(s-\lambda_s+\lambda_s\beta(s))} \tag{4}$$

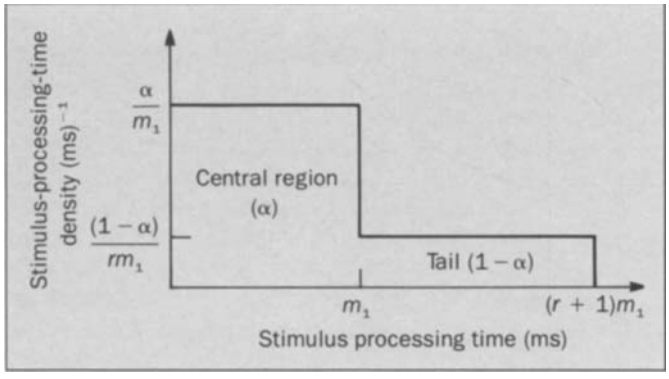

Figure 5. Three-parameter family of densities.

where $\rho = \lambda_s E(X)$ is the SPE call processing utilization, λ_s is the stimulus arrival rate, E denotes expectation, and P_o is the (time average) probability that the system is empty (no stimuli are in queue or in service).

It is easy to show that

$$P_o = \frac{1-\rho}{1+\lambda_s E(Z_\tau)}$$

and hence equation (4) can be rewritten as

$$\alpha_\tau(s) = v(s)\, w(s) \tag{5}$$

where

$$v(s) = \frac{\lambda_s - (\lambda_s - s)\,\delta(s)}{s(1+\lambda_s E(Z_\tau))} \tag{6}$$

and

$$w(s) = \frac{s(1-\rho)}{s - \lambda_s + \lambda_s \beta(s)} \tag{7}$$

$w(s)$ is, of course, the standard M/G/1 waiting-time transform, and this factorization shows that uplink buffer delay is the sum of ordinary M/G/1 delay and a timer-induced component whose transform is given by equation (6). We assume that Z_τ is uniformly distributed in $[0,\tau]$, since the probability of two or more stimuli in length τ is small at typical call rates.

We note, as an aside, that equation (4) can also be obtained [and the factorization in equation (5) anticipated] by viewing the model as an M/G/1 queue with setup time as shown by Doshi in Reference 4, equation (4.4).

A stable and efficient numerical inversion technique developed recently by Jagerman[5,6] can be used to obtain the distribution of uplink buffer delay $F_B(t) = P(W_\tau \leq t)$ from equation (4).

Stimulus-Response-Time Distribution. If $N(s)$ denotes the Laplace Stieltjes transform of the distribution of network delay (see Appendix A), $\alpha_\tau(s)$ is the Laplace Stieltjes transform of the distribution of uplink buffer delay obtained in the previous section, and t is the required SPE processing time for the call-processing stimulus under consideration, then the Laplace Stieltjes transform of the response-time distribution is given by:

$$F(s) = N(s) \cdot \alpha_\tau(s) \cdot e^{-ts} \tag{8}$$

The assumptions here are that uplink buffer delay and network delay are independent, random variables, and that SPE processing time t for a given stimulus is deterministic. These assumptions appear to be entirely realistic.

Using the Jagerman transform inversion technique,[5,6] it is very easy to invert $F(s)$ and numerically obtain the distribution of stimulus response time. Note that the input parameters are

- λ_s—The stimulus arrival rate (stimuli per second)
- α, m_1, r—The parameters that determine the stimulus-processing-time distribution
- t—The SPE processing time of the stimulus in question.

The three parameters α, m_1, and r, can be chosen to match the average stimulus processing time, which is obtained from the workload characterization. Clearly [see equation (3)], this constraint leaves two degrees of freedom; this freedom can be used to understand the sensitivity of response-time percentiles to uncertainty, as the examples in the section below illustrate. The stimulus arrival rate, λ_s, is an independent variable and is related to

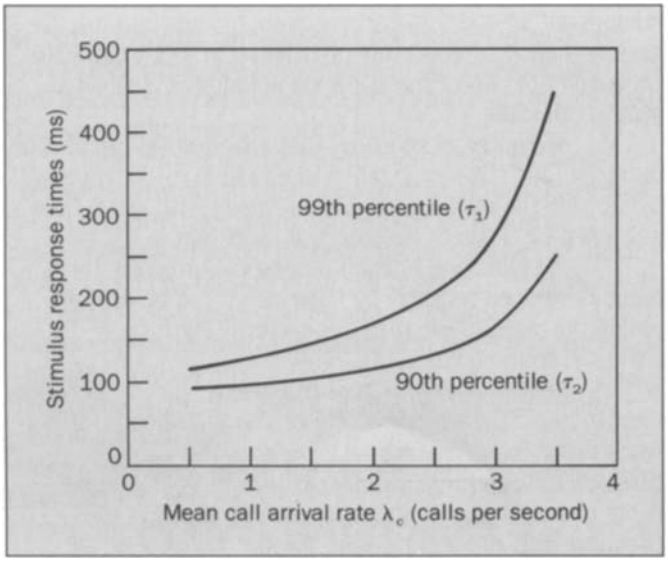

Figure 6. 90th and 99th percentile stimulus response times. m_1 = 36 ms; α = 0.92; s_{max} = 50 ms.

the average calling rate λ_c by the equation

$$\lambda_s = \bar{n}\lambda_c$$

where $\bar{n} = \sum_i p_i n_i$ is the average number of stimuli per call and is obtained from the workload characterization.

Illustrative Example. Suppose that the two most time-critical responses are r_1 and r_2, which are generated in response to stimuli s_1 and s_2. Let τ_i denote the response time for r_i to be generated in response to stimuli s_i and assume that the performance criteria for τ_i are known:

$$P(\tau_i > \alpha_i) < q_i \qquad i = 1,2$$

For concreteness, assume α_1 = 300 ms, q_1 = 0.01 and α_2 = 150 ms, q_2 = 0.10. Thus, the 99th and 90th percentiles of τ_1 and τ_2 may not exceed 300 ms and 150 ms, respectively.

Suppose that a preliminary standard-call-mix model has been defined and measurements show that the average stimulus processing time is 20 ms, the maximum

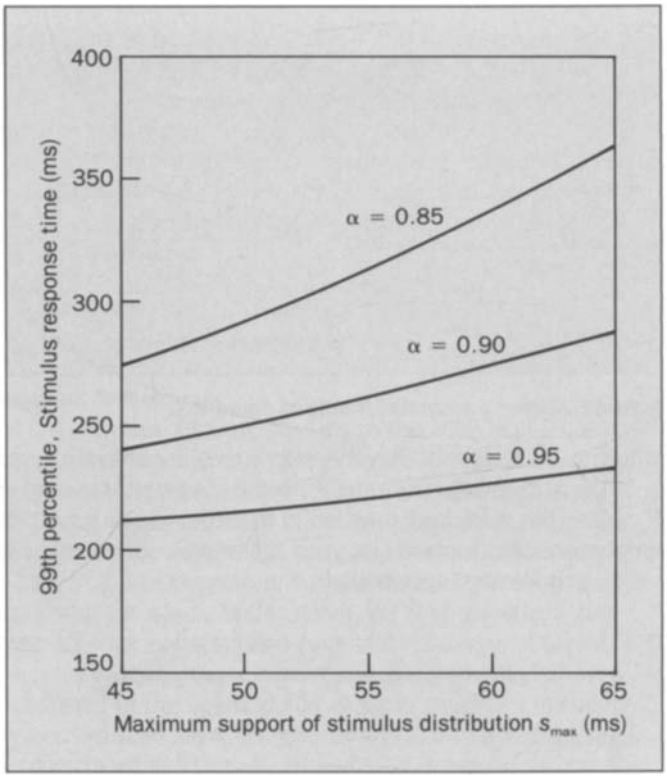

Figure 7. Sensitivity of 99th percentile stimulus response time to workload parameters. m_1 = 36 ms; λ_c = 2.8 calls per second.

stimulus processing time is 50 ms, and most stimuli have processing times less than 36 ms. The distribution of stimulus processing times for this preliminary call model may be approximated using the three-parameter stimulus-processing-time distribution (described in the section on processing time) by choosing α = 0.92, m_1 = 36 ms, and s_{max} = 50 ms. Note that, although the exact stimulus-processing-time distribution is available for this preliminary call model and may also be used (e.g., all stimulus processing times have been measured), little benefit will accrue from using it. The reason for this is twofold. First, the differences in response times obtained when using the exact and the approximate three-parameter distribution are negligible. More importantly, however, is the fact that we are primarily interested in the sensitivity of response times to

changes in the standard-call-mix model; and these are most easily examined by perturbing the three-parameter distribution.

Assume that the two stimuli of interest (s_1 and s_2) have SPE processing times of 9 ms and 16 ms, respectively (in the preliminary call model), and that the average number of stimuli per call is 12. Using the analysis described in the section on the stimulus-response-time distribution and the powerful inversion technique of Jagerman mentioned earlier, we easily compute the 99th and 90th percentiles of τ_1 and τ_2, respectively, as a function of the average call arrival rate λ_c (see Figure 6). In this example, the 90th criterion on τ_2 is limiting, and we find that both criteria are satisfied as long as $\lambda_c \leq 2.8$ calls per second (the 99th percentile of τ_1 is equal to 300 ms when $\lambda_c = 3.1$). Note that the average call in the preliminary call model requires 12×20 ms $= 240$ ms of SPE processing and, hence, at capacity ($\lambda_{c,\max} = 2.8$), the SPE call processing utilization is approximately 67 percent.

Because the preliminary call mix model is often just a rough estimate of one of many call mixes that will be encountered in a real user environment, it is necessary to study the sensitivity of capacity to changes in the call model. If, for example, there is evidence to suggest that "most" of the stimuli in a more mature call mix will have SPE processing times in the same central region (less than 36 ms) but there is uncertainty about what "most" is, and perhaps even more uncertainty about how large $s_{\max}$ may be, it is prudent to vary both α and $s_{\max}$.

In Figure 7, we show the 99th percentile response time (τ_1) as a function of $s_{\max}$, with the average calling rate fixed at 2.8 calls per second. The different curves shown are parameterized by various values of α. It is important to note that the average stimulus processing time increases with increasing values of α and/or $s_{\max}$ and, since the calling rate is fixed in this figure, so too is the SPE utilization—due to call processing. Thus, for example, the SPE utilization due to call processing (in Figure 7) is 63 percent at $\alpha = 0.95$, $s_{\max} = 45$ ms but 76 percent at $\alpha = 0.85$, $s_{\max} = 65$ ms.

Although both the parameters and the response-time criteria used in this section are fictitious (as is the resulting SPE call capacity), the examples are similar to calculations that were used to evaluate the call processing performance of System 75 while it was still in the development stage.

Conclusion

Measurements, modeling, and analysis were employed in System 75 primarily to give developers timely feedback on the performance implications of various design decisions. Because of space limitations, this paper has focused on the development of an accurate model to predict stimulus response times (and, hence, SPE call capacity). However, the model did find application in the development process, where architectural refinements were evaluated and adopted, and most significantly in support of subsequent decisions to increase the advertised call capacity and line size. As a result, System 75 was able to cover a much larger segment of the PBX market, which has been a key factor in its profitability. Furthermore, the modeling and analysis effort was considerably more far reaching than we could detail in this brief paper. Similar studies were undertaken to aid development efforts in such other areas of System 75 as system administration and traffic overload control.

References

1. *AT&T Technical Journal*, Vol. 64, No. 1, Part 2, January 1985.
2. J. S. Kaufman, M. S. Karlsson, and J. S. Willie, "Workload Characterization in PBX Performance Studies," *Proceedings of the International Conference on Private Switching Systems and Networks*, London, June 1988.
3. R. W. Conway, W. L. Maxwell, and L. W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading, Massachusetts, 1967.
4. B. T. Doshi, "A Note on Stochastic Decomposition in a G/G/1 Queue With Vacation or Set-Up Times," *Journal of Applied Probability*, Vol. 22, 1985.
5. D. L. Jagerman, "An Inversion Technique for the Laplace Transform with Application to Approximation," *The Bell System Technical Journal*, Vol. 57, Part 1, No. 3, March 1978, pp. 669-710.
6. D. L. Jagerman, "An Inversion Technique for the Laplace Transform," *The Bell System Technical Journal*, Vol. 61, Part III, No. 8, October 1982, pp. 1995-2002.

Appendix A. Distribution of Network Delay

Let N_u and N_d denote the uplink and downlink components of network delay, respectively. Assuming MFET angels, N_u is approximated by:

$$N_u = u[25,50] + u[0,3] + u[0,3] + 2 \qquad \text{(A-1)}$$

where $u[a,b]$ denotes a random variable uniformly distributed between a and b. As described in the section on analytic modeling, uplink network delay is dominated by the angels' 25-ms scan rate (e.g., stimuli must be detected on two successive scans, which requires between 25 and 50 ms). The archangel scans all angels for activity before transferring messages from any angel. Thus, two archangel polling cycles (at most) can elapse before activity (e.g., an uplink message) is detected at an angel, which accounts for the two $u[0,3]$ terms in equation (A-1). The 2 ms of deterministic delay is included to account for message handling overhead at both the angel and TDM bus.

Downlink network delay, N_d, is approximated by:

$$N_d = u[0,25] + u[0,5] + u[0,3] + 2 \qquad \text{(A-2)}$$

The dominant term in N_d is also due to the angels' 25-ms scan rate, and accounts for the delay encountered when a message (e.g., ringer, lamp update, etc.) must be sent to an MFET device. The $u[0,5]$ term accounts for the MFET message handler delay in responding to a message and the $u[0,3]$ term accounts for the one (at most) archangel polling cycle required to transfer a message from the downlink buffer to the MFET. The 2 ms of deterministic delay accounts for both angel message handler and TDM bus transport delay.

Thus, our approximate model of network delay $N = N_u + N_d$ can be written as:

$$N = N_{\text{scan}} + N_{\text{poll}} + u[0,5] + 29 \text{ (ms)} \qquad \text{(A-3)}$$

where N_{scan} is the sum of two independent $u[0,25]$ random variables and N_{poll} is the sum of three independent $u[0,3]$ random variables. Equation (A-3) is the characterization of network delay used in all stimulus-response-time calculations. Note that the distribution of N has support on the interval [29, 93], and has a mean of 61 ms.

Biographies (continued)

Institute, an M.S.E.E. from the University of Michigan, and a Ph.D. in computer, information, and control engineering from the University of Michigan.

(Manuscript received August 19, 1988)