

A PERFORMANCE MODEL OF SYSTEM DELAY AND USER STRATEGY SELECTION

Steven L. Teal

Graduate School of Industrial Administration
Carnegie Mellon University
Pittsburgh, PA 15213
st0i@andrew.cmu.edu

Alexander I. Rudnický

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
air@cs.cmu.edu

ABSTRACT

This study lays the ground work for a predictive, zero-parameter engineering model that characterizes the relationship between system delay and user performance. This study specifically investigates how system delays affects a user's selection of task strategy. Strategy selection is hypothesized to be based on a cost function combining two factors: (1) the effort required to synchronize input with system availability and (2) the accuracy level afforded. Results indicate that users, seeking to minimize effort and maximize accuracy, choose among three strategies – automatic performance, pacing, and monitoring. These findings provide a systematic account of the influence of system delay on user performance, based on adaptive strategy choice drive by cost.

KEYWORDS: System response time, strategy selection, interface design, human factors

INTRODUCTION

The effect of system delay on user performance has been of interest to practitioners and researchers since the emergence of timesharing systems in the 1960s [e.g., 11, 15, 19, 22, 24]. While many feel that this problem became a non-issue with the introduction of faster and cheaper computing technologies, many others find that this interaction still remains problematic [18, 23]. In fact, Shneiderman [23] still lists this issue as one of the seven “golden opportunities” for conducting pioneering and productive research in human-computer interaction.

The failure of researchers to adequately address this issue has resulted in a lack of clear, concise guidelines that system engineers can use to support crucial design decisions. Engi-

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.

neers are therefore often forced to either guesstimate “acceptable” delays on a case-by-case basis or test “user satisfaction” with prototype systems under various system response time and/or hardware/software configurations.¹ However, these methods can prove to be ineffective, time consuming and expensive, and provide no long-term solutions to the problem.

A more promising method for addressing this issue would be to work toward the development of a predictive, zero-parameter engineering model that would characterize the relationship between system delay and user performance. The present study lays the ground work for this process by systematically investigating the relationship between system delay and a user's choice of task strategy. We address two questions. First, does system delay in fact have a significant effect on user performance? Second, if delay does affect user performance, can we define a model that adequately describes this relationship?

PRIOR RESEARCH

Most previous research has assumed that system delay has an effect on user performance and that this effect can be evidenced through increased user productivity at decreased system response times. As an example, Martin & Corl [14] investigated the behavior of subjects performing both data entry and problem solving tasks. Results indicated significant effects across conditions, with productivity increasing with decreased system delays; however, further analysis revealed strong productivity increases for the data entry tasks but not for the problem solving tasks. These results support the hypothesis that an inverse relationship exists between system delay and user productivity, and are consistent with the findings of several other studies [e.g., 1, 13, 25].

¹ Though a technical difference exists between the terms “system response time” and “system delay”, for the purposes of this paper these terms should be considered interchangeable.

In contrast, Dannenbring [6] presented results of a data entry task which indicated a marginally significant effect for user productivity with task completion times increasing with decreasing delays. These results run counter to the hypothesis that an inverse relationship exists between delay and user productivity but are consistent with the findings of other studies that either found a similar effect [2] or failed to find any effect at all [e.g., 4, 5, 7].

A survey of the system response time literature reveals additional instances of contradictory findings. The source of these contradictions can be attributed to differences in experimental designs that include:

1. The type of task being studied. Tasks ranged from simple data entry tasks [6] to complex problem solving tasks [14] to open-ended programming tasks [13]. The use of different tasks can account for some of the inconsistencies in results since task effects are generally acknowledged to be a potential intervening variable in studies of system delay and user performance [e.g., 14, 15, 22].
2. The amount of control exerted over the task. Several of the studies were observational in nature and the researchers were not able to systematically control what task (or tasks) the subjects were performing [e.g., 1, 13, 25]. This makes it difficult to make comparisons across studies even for those studies which supposedly used similar tasks.
3. The amount of control exerted over the delay conditions. In a number of studies, researchers were not able to tightly control the delay conditions presented to the subjects [e.g., 1, 13, 25]. This situation normally occurred when data was collected from an active work site with the subjects connected to a host system. While subject transactions were given a high priority, the delays presented were still dependent on the processing load of the host computer and may therefore have varied widely, in mean and variance, from those intended by the researchers.
4. The match between the task and delay conditions. Most studies did not use delay conditions specifically chosen to complement the experimental task [e.g., 3, 6, 14]. For instance, data entry tasks which may call for subsecond response times were tested with multisecond delays [6] while complex problem solving tasks which may require a great deal of external user preparation between interactions were

implemented with subsecond delays [13]. The success or failure of finding an effect may therefore be blurred by the mismatch of task structure and delay condition.

5. The choice of dependent variables. Many studies did not use dependent variables specifically designed to isolate any hypothesized effects [e.g., 6, 14, 25]. More specifically, the use of aggregate variables such as 'total task completion time' and 'total number of errors' may have disguised effects which would have identified through the use of variables which measured performance at a finer grain of analysis.

The present study was designed to avoid these difficulties. First, task effects were minimized by the use of a simple data entry task. Further, a performance model, detailing the interaction of system delay and user performance, was developed to facilitate an understanding of these effects. Second, the study was performed in a laboratory setting, thus providing a means to control the task being performed by the subjects. Third, the study was conducted on a sole-use computer to provide maximum control over the delay conditions experienced by the subjects, and constant delays were instituted to exclude the issues of predictability and variability.² Fourth, the delays were matched to the task through performance model predictions and pilot study refinements. Finally, the dependent variables chosen specifically supported testing of the effects hypothesized by the performance model.

DEFINITIONS

Figure 1 presents a detailed analysis of an interaction cycle in terms of system and user events.

System Model

The system model is composed of three components: system response time, display time and lockout time. *System response time* is the time that elapses between when the user initiates an activity and when the computer initiates its response. This measure is composed of three subcomponents: computation, overhead and delay. *Computation* is the time required by the system to perform all operations necessary to support the user's requested activity. *Overhead* is the time required by the system to complete any activity logging after

2. Constant delays were used to ensure that this initial investigation would not be complicated by potentially intervening variables. Once a basic understanding of the effect of system delay on user performance has been developed several extensions can be added including (1) delay variability and/or predictability, (2) task effects, and (3) user modalities.

all computations have been completed. Finally, *delay* is the sum of all time lags which extend the system response time beyond that required for all computational and overhead requirements.

Along the same lines, *display time* is the time required by the system to display the results of the user's request on the proper output device, and *lockout time* is the time that elapses between when the system displays the results for the current activity and when the system becomes available for entry of the next activity.

User Model

The user model is composed of two events: user planning and acquisition time, and execution time. *User planning and acquisition time* is the time that elapses between when the user initiates the current system activity and when the user begins entering the next task into the computer. This measure is composed of four sub-components: system response time, display time, lockout time, and user delay. The first three of these sub-components are detailed above. *User delay* is defined as the time that elapses between when the sys-

tem is ready for the next system input and when the user actually begins to enter the next activity into the system. The last component of the user model, *execution time*, is the time required by the user to enter the task into the system.

Research Variables

It is our belief that observable changes in user behavior under different system delays can be attributed to shifts between different task strategies. Focusing on these strategy shifts is therefore central to understanding the relationship between system delay and user performance, since users will select task strategies which optimize their performance within the constraints imposed by the system. We can adduce the form of these strategies and track users as they shift between them by measuring the following variables:

1. Initial keystroke latency: Initial keystroke latency is the time that elapses between when the computer signals that it is ready for the next input and when the user actually strikes the first key to begin the activity (i.e., the user delay).

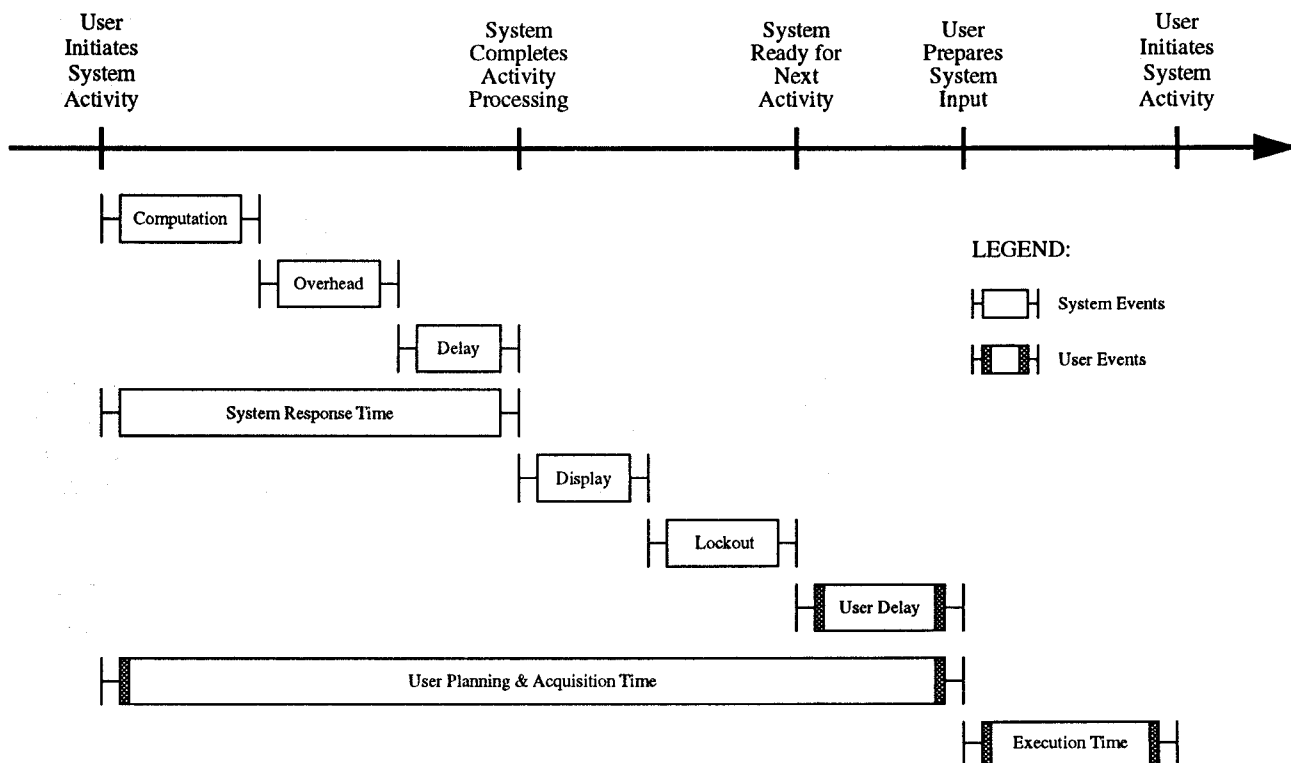


Figure 1. Model of user and system events occurring during a normal human-computer interaction.

2. Subsequent keystroke latency: Subsequent keystroke latencies are the inter-keystroke times for all user inputs other than the initial keystrokes (i.e., the subcomponents of the user's execution time).
3. Anticipation errors: Anticipation errors occur when the user attempts to enter a keystroke prior to the system becoming available for the next user input.³
4. Keying errors: Keying errors are all performance errors other than anticipation errors (e.g., transcription errors).

PERFORMANCE MODEL

Initial observations indicated that system delays induce three different task strategies, each characteristic of a particular delay region. The choice of strategy was a rational one based on the relative cost of available strategies, where cost reflects the strategy's demand for cognitive resources and the accuracy rate it affords [e.g., 9, 10, 21].

Region One – Automatic Performance

In the simplest case, system delay is equal to zero. Since under this scenario the user will never have to wait on the computer, task time (as measured by initial keystroke latencies) will be equal to the total time required to retrieve and begin entering the next data item (i.e., the user's planning and acquisition time) (Figure 2a). Similarly, since the system will be ready for entry of the next data item immediately following entry of the current item, it will be impossible for the user to initiate the next activity prior to system availability. The anticipation error rate will therefore be zero (Figure 3a).

As system delays increase, progressively more of the user's planning and acquisition time will be absorbed into the system delay interval. Thus, the user's task time will (falsely) appear to improve in a linear fashion, since initial keystroke latencies are, by definition, derived by subtracting any system delay from the user's planning and acquisition time. As an example consider a user with a task planning and acquisition time of 0.75 secs. Using this parameter, initial keystroke latencies of 0.75, 0.50, and 0.25 secs would be derived with respective delays of 0.00, 0.25, and 0.50 secs. In each case, the user's actual performance would remain constant but the initial keystroke latency measurements would decrease in relation to the system delay provided.

³ A non-buffered input system was chosen to force subjects to contend with the implemented delay conditions (see footnote 2).

This linear decrease in initial keystroke latency will continue until a boundary point is reached where the system delay is equal to the user's planning and acquisition time and an initial keystroke latency of zero is achieved (Figure 2b). At this point the user, theoretically, never waits for the system and the system never waits for the user. However, this boundary point will rarely, if ever, be reached due to the commission and cost of anticipation errors.

Due to the variability inherent in user performance, actual task times will be distributed about some mean value with anticipation errors occurring at a rate corresponding to the shaded region in Figure 4. If a user does not change task strategies and maintains a consistent task time distribution then anticipation error rates in this region will increase with increases in system response times (Figure 3b). The cost of completing the task will therefore increase with increased system response times and, if no corrective action is taken by the user, can become (virtually) unbounded as a 100% error rate is approached. We therefore hypothesize that at some system response level the cost of completing the task will exceed some "acceptable level of effort for the task" and the user will attempt to select a strategy that will successfully accomplish the task but with a lower associated total cost. The tangible result of this process will be that users will never reach a zero-second initial keystroke latency but will shift strategies prior to this point. This shift will be reflected in a change in the user's initial keystroke latency and anticipation error rate (Figures 2c and 3c).

In summary, we can hypothesize the following about user performance in this region. First, with a system delay of zero seconds we expect to see an initial keystroke latency equal to the user's planning and acquisition time and an anticipation error rate of zero. Second, as system delays increase we expect to see a linear decline in initial keystroke latencies and an associated rise in the number of anticipation errors committed. Finally, prior to the achievement of a zero-second initial keystroke latency, we expect to see a shift in initial keystroke latency and anticipation error rate measurements thereby signaling a shift in task strategies.

Since system delays within this region do not require the user to develop any particular task completion strategy in reaction to external demands, we will refer to the characteristic strategy of this delay region as *automatic performance*.

Region Two – Pacing

There is evidence [e.g., 16, 26] that humans, given a predictable stimulus interval, are able to accurately reproduce that interval (as long as the interval remains relatively short).

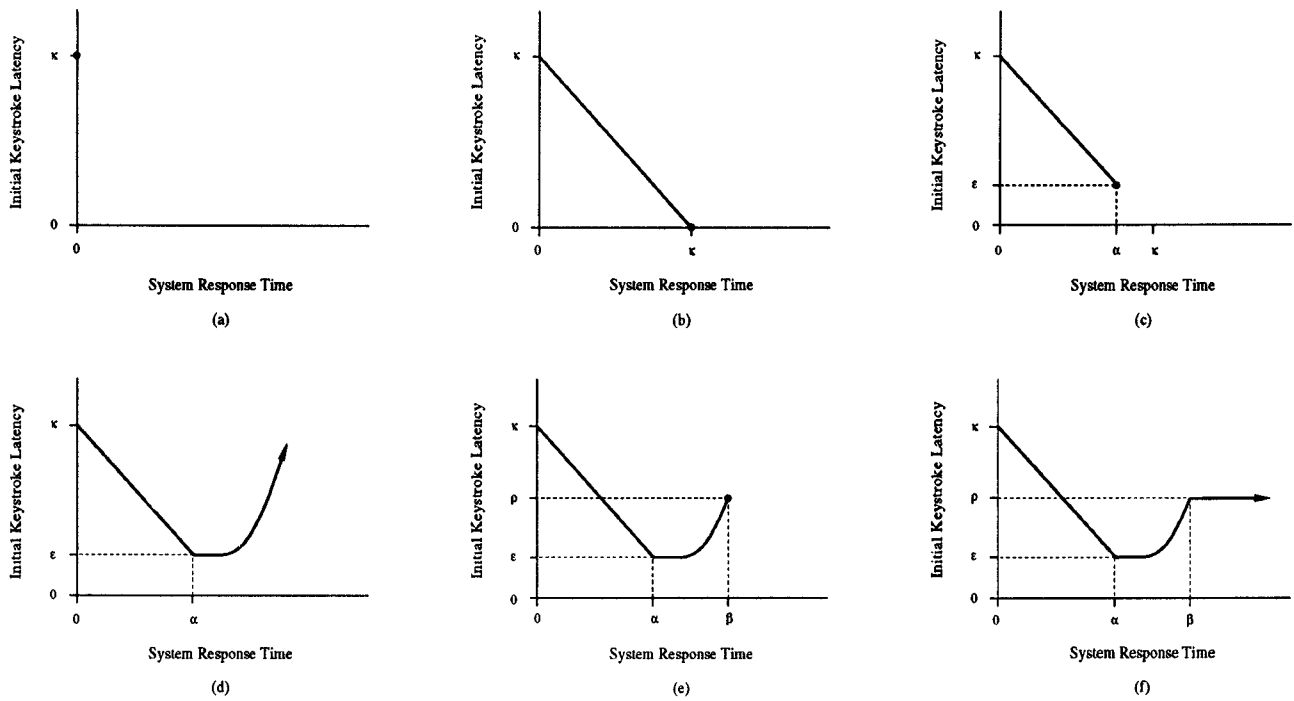


Figure 2. Hypothesized effect of increased system delay on the user's initial keystroke latency with strategy shift points occurring at α (automatic performance to pacing) and β (pacing to monitoring).

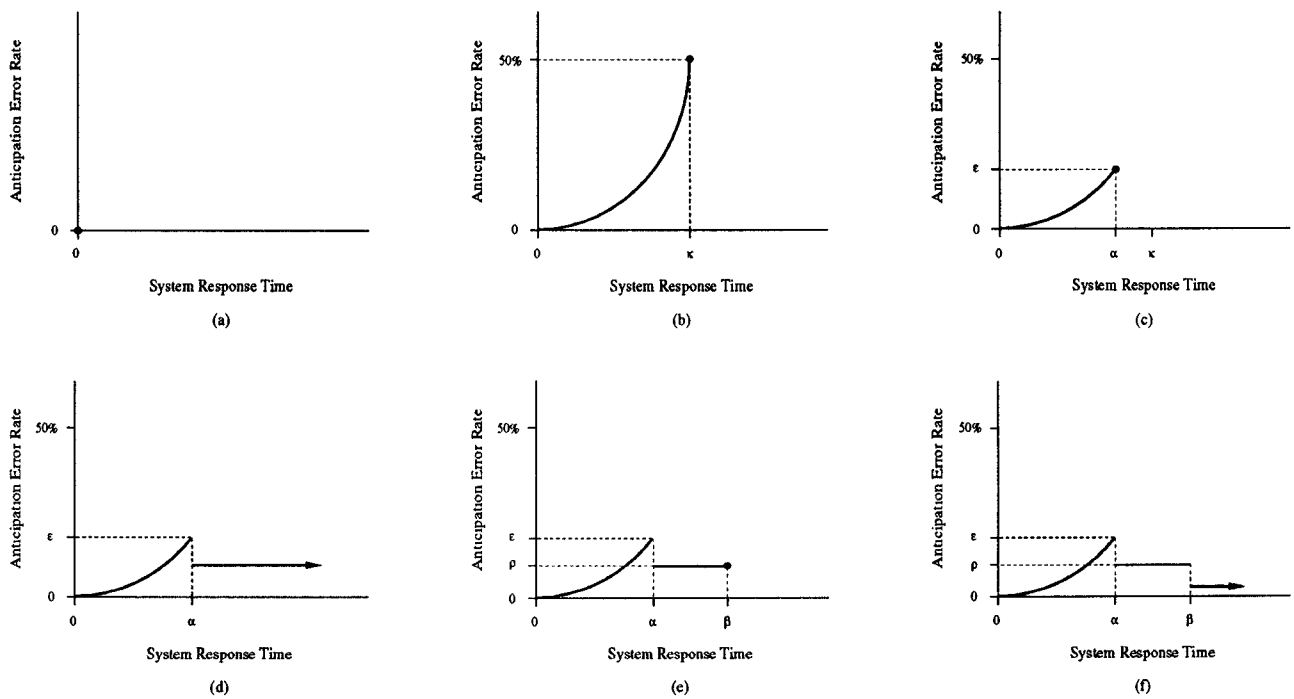


Figure 3. Hypothesized effect of increased system delay on the user's anticipation error rate with strategy shift points occurring at α (automatic performance to pacing) and β (pacing to monitoring).

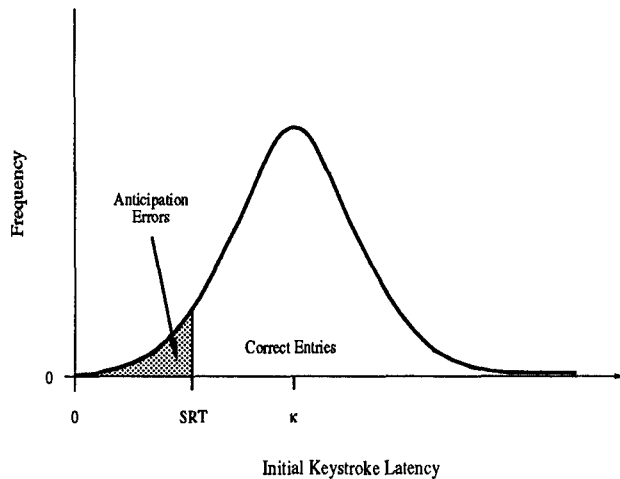


Figure 4. Example initial keystroke latency distribution illustrating the areas of anticipation errors and correct entries derived from the interaction of the system response time (SRT) and the user's mean initial keystroke latency (κ).

Such a skill can be brought to bear in the current situation and incorporated into a strategy that takes into account expected system delays. This would allow the user to reduce the anticipation error rate while maintaining a timely rate of response. Therefore, users can engage a strategy that delays initiation of their response long enough to reduce the number of anticipation errors committed and thereby lower the total cost of completing the task.

This argument is not complete however, for if users merely synchronize their performance with system availability then a 50% error rate would still be expected.⁴ To effectively reduce the error rate, users must not only synchronize their performance with the estimated system delay but must also add an additional "buffer period" to this estimate. This buffer serves to positively shift the user's task time distribution and thereby further lower the associated anticipation error rate. This buffer period will be fine tuned over a number of trials until the anticipation error rate is brought within the individual subject's "bounds of acceptability" (Figure 5).

Task times within this region will therefore be equal to the user's estimate of the system response time plus the addi-

⁴ A normal $N(0,1)$ distribution is assumed here simply for the sake of argument. While actual performance distributions will in all likelihood result in some positive degree of skewing, the argument of a 50% error rate closely approximates the user performance distributions most likely to occur.

tional time included for the buffer period. Correspondingly, anticipation error rates will drop from the level of the previous region and will remain stable but nonzero. Perfect performance is not expected, due both to the variability inherent in user performance and to constant user performance adjustments of the type described by Pachella [17].

Performance in this region will begin to break down as the user's ability to accurately estimate the system delay decreases with increased delay times [16]. With each increase in system delay, the user's task time distribution must therefore shift in increasing proportions to ensure an acceptable error rate (Figure 2d and 3d). This increase in mean task time will increase the total cost of completing the task until, as with the shift from automatic performance, the cost of completing the task exceeds some "acceptable level of effort for the task" and the user again attempts to select an alternative, lower-cost strategy (Figures 2e and 3e).

In summary, we can hypothesize the following about user performance in this region. First, we expect initial keystroke latencies and variances to remain relatively steady through the first part of this region but to rise as system delays increase. Second, we expect anticipation error rates to remain relatively stable.

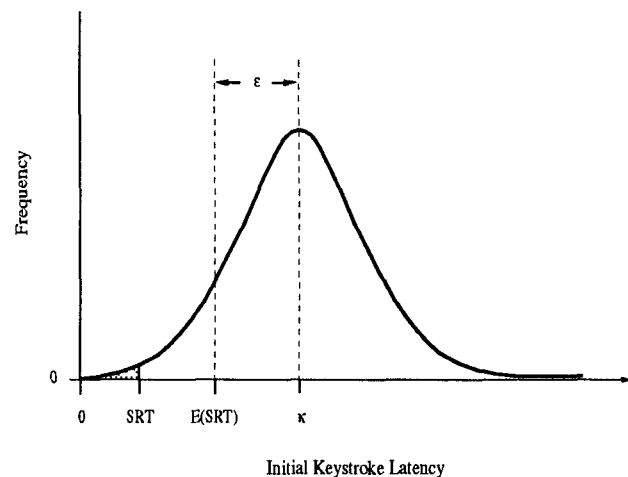


Figure 5. Example initial keystroke latency distribution illustrating the relationship between the user's estimate of the system response time ($E[SRT]$) and the buffer delay (ϵ) added to this estimate to keep the error rate within an acceptable range. The anticipation errors rate consists of the shaded area of the distribution lying to the left of the system response time (SRT).

Since users attempt to synchronize their performance with the system delay, we will refer to the characteristic strategy of this delay region as *pacing*.

Region Three – Monitoring

At sufficiently long delays, users will no longer be able to synchronize their performance with the system delay and keep the accuracy rate within the “bounds of acceptability”. They will then be forced to rely on external stimuli to signal system availability. Initial keystroke latencies will therefore be equal to the user’s simple reaction time (Figure 2f) and the error rate will drop to near-zero (Figure 3f).

Since users are forced to rely on an external stimuli to indicate system availability, we will refer to the characteristic strategy of this delay region as *monitoring*.

EXPERIMENT 1 – INITIAL INVESTIGATION

This experiment tested the proposal that system delay affects user performance and that it leads to the selection and use of different task strategies within distinct delay regions.

Method

Subjects. Fourteen subjects were recruited from the general university population. No computer experience was necessary. All subjects were naive regarding the experimental manipulation and were fully debriefed following the experiment. Subjects were paid for their participation.

Apparatus and Materials. The experimental stimuli were presented to the subjects on a Macintosh IIfx personal computer. The computer was equipped with a color monitor, and used a standard Macintosh keyboard with the numeric keypad on the right-hand side. To provide maximum control over system delays, the computer was disconnected from all outside networks and all nonessential memory resident programs (e.g., screen savers, RAM disks) were removed.

The materials used for this experiment consisted of seven sets of 90 three-digit pseudorandom numbers. Number sets were presented to the subjects on sheets containing 45 numbers each. The numbers on each sheet were arranged into groups of fives forming a 3x3 matrix with rows separated by single line dividers and columns separated by double line dividers. Each data sheet was encased in a plastic, non-reflective protective cover and placed on an upright typing stand.

The software program which controlled the experiment was written in the cT programming language [20]. A delay module included in the program delayed the computer’s response for a predetermined time after the subject entered a complete three-digit number. The timing algorithm had a worst-case

cycle time of 35 msec. Experimental conditions were thus implemented using the target delay minus the worst-case algorithm cycle time.⁵

The software program was designed so that all subject keystrokes and their associated latencies were recorded; however, not all keystrokes were echoed back to the subject on the computer screen. In particular, keystrokes which occurred during the target delay period (i.e., anticipation errors) were not displayed. These keystrokes were recorded to disk but were essentially lost from the subject’s perspective. The subject therefore experienced a non-buffered input system. Once the target delay was complete, all subsequent keystrokes were displayed.

Experimental Design. A single-factor, within-subject design was used for this experiment. The conditions were based on a pilot study which provided the following approximations for the strategy shift points: 0.625 sec for shifting to a pacing strategy, and 3.00 sec for shifting to a monitoring strategy. The seven delay conditions used were: 0.00, 0.312, 0.625, 1.25, 2.50, 5.00, and 10.00 secs. An N x 2N Latin square design was used to counterbalance potential ordering effects. Subjects were randomly assigned to the conditions.

Procedure. The experiment was run in a quiet, controlled environment. At the beginning of the session, the task was verbally described to the subject using a sample data sheet. In addition, the following instructions were provided just before initiation of the first task. First, numbers could only be entered into the system using the numeric keypad. Second, any keying errors that occurred could be corrected using the delete key on the main keyboard; however, once a complete number had been entered into the system it could not be corrected. Third, if positional sequencing between the data sheets and the computer screen was lost, the subject was to reorient themselves using the current computer data entry cell as a base and continue with the task ignoring any previously committed errors. Finally, the subject was instructed to perform the task as quickly and accurately as possible. Subjects were also informed during this time that they might experience some delays in the computer’s performance due to other processing tasks that the computer was performing. Subjects were not informed of the number of conditions that they would experience.

⁵ Given the sophistication of the equipment used for this experiment, the computer’s display time was considered to be virtually instantaneous and was therefore not included as a factor when setting the target delays. Similarly, since a lockout period was not instituted, this response time component did not factor into setting the conditions.

At the beginning of each experimental condition, subjects were given a new set of random numbers which they placed on the typing stand situated next to the computer. Each condition was initiated by the subject striking any key on the keyboard. This action caused the computer screen to be cleared and a replica of the data sheets (minus the random numbers) to be drawn on the screen. At this time, the screen contained only the lines dividing the rows and columns, a box indicating the current data entry position, and an underscore character inside the data entry box indicating where the next digit that the subject keyed in would appear. Subjects then began to enter the current number set into the computer.

Subjects entered numbers into the computer using the numeric keypad on the right hand side of the keyboard. With each key press the appropriate digit was appended to any digits previously keyed into the system and the resulting number centered in the data entry box. Once all of the individual digits had been keyed, the subject entered the resulting number into the computer by pressing either the 'enter' or the 'return' key. If the number in the data entry area consisted of three digits (whether or not it correctly corresponded to the current number of interest), it was accepted for entry and the underscore character and data entry box were erased. If the number contained too few or too many digits then the computer signaled an error with an audible beep and the subject had to correct the error before the transaction would be accepted. Once a valid number had been entered into the system, the target delay for that condition was instituted. After the target delay had lapsed the underscore character and data entry box were redrawn in the next data entry location and the computer was ready to receive the next data entry.

Subjects entered numbers into the computer beginning with the first number in the upper left-hand corner of the first data sheet. They then worked down the sheet until all of the numbers in the first column had been entered. They then moved to the top of the next column to the right and performed the same procedure. This process continued until all of the numbers from the first data sheet had been entered into the system. At this point, the subjects removed the sheet from the typing stand, placed it aside on the desk, and began working on the second sheet.

The computer screen automatically prepared for entry of the second sheet of numbers after a valid number had been entered into the lower right-hand cell of the computer data entry screen. When all of the numbers from the second data sheet had been entered, the subject pressed the 'q' key to end the experimental session. The computer screen then faded black and a message was displayed informing the subject

that the next condition could be started at any time by pressing the 'return' key on the keyboard. Subjects were allowed an optional two-minute rest period after any of the experimental conditions and all subjects were required to take a five minute rest after completion of the fourth condition. This process continued until the subjects completed the task under all seven experimental conditions.

Results

We examined latency variance across the 90 trials in each condition and found it to be elevated over the first 15 trials, but stable thereafter. Presumably the subjects were using these initial trials to adjust their performance for each condition, accordingly we excluded the first 15 trials from subsequent analysis. Figure 6 displays initial keystroke latencies and anticipation errors across the seven conditions used in this experiment. The form of the latency curve agrees quite closely with the curve shown in Figure 2f and the anticipation error curve follows the one shown in Figure 3f. The curves for individual subjects agree with the composite curves, although their location along the time axis varies considerably.

An analysis of variance was performed for the four variables described earlier. This analysis revealed significant effects for initial keystroke latency ($F[6,72] = 63.05, p < .001$), subsequent keystroke latency ($F[6,72] = 3.53, p < .01$), and anticipation errors ($F[5,60] = 6.78, p < .001$).⁶ Ordering effects were only significant for keying errors ($F[6,72] = 4.77, p < .001$). Initial keystroke latency and anticipation error results are presented in Figure 6.

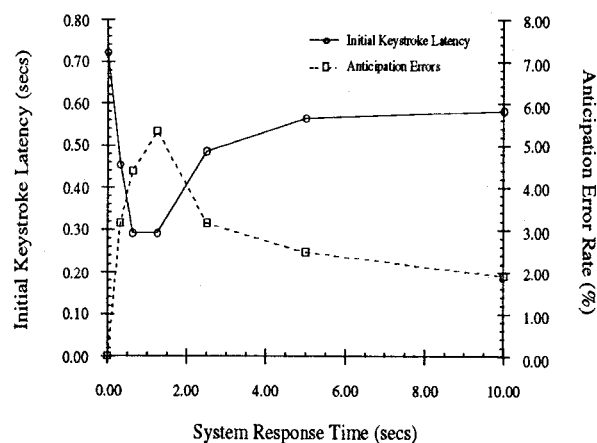


Figure 6. Double y-axis plot detailing the results of Experiment 1 with initial keystroke latency and anticipation error rate plotted against system response time.

Analysis using Tukey's HSD revealed significant differences between three system delay regions – 0.00 to 0.625 secs, 1.25 to 2.50 secs, and 5.00 to 10.00 secs – for initial keystroke latencies and anticipation errors. The differences were significant at the $p < .001$ level. There were no significant within-region differences. Neither subsequent keystroke latencies nor keying errors demonstrated significant differences using this more conservative measure.

Discussion

The results of this experiment clearly support an account of system delay effects using the cost-based strategy selection model presented earlier. The results are also consistent with the seemingly contradictory findings reported by others. It is clear from our data, for example, that Dannenbring's [6] failure to find an effect was due to an improper sampling of delays and that delay effects in Martin & Corl's [14] study were masked by an improper choice of dependent variables. In this latter case, if we transform our data into the form used in that study ('total task completion time' and 'total number of errors') and recompute the analysis, we find that our data produce results comparable to those reported earlier.

The results obtained for subsequent keystroke latencies speak to the adaptability of the subjects. Times for this variable decreased with increasing delays as subjects used any "down time" to prepare for the next task by either placing their fingers on the appropriate number keys or practicing the number (without actually pressing the keys) while waiting.

The ordering effects obtained for keying errors can apparently be explained by subject fatigue. Replotting the data by experimental position showed that keying errors increased over the first four trials, dropped off after the mandatory rest period, and then again climbed until the end of the experiment.

Finally, although the data in Figure 6 support our model, it was clear that a high degree of inter-subject variability was present. For example, mean initial keystroke latency for the no-delay condition ranged from 0.583 to 0.979 sec. One consequence of this was to obscure the shape of the (aggregate) curves in Figure 6 as subjects switched task strategies at different system delays. Another consequence was a sub-optimal sampling of delays in the pacing region, hampering analysis of certain other variables of interest, such as latency

6. By definition, it is impossible to commit an anticipation error with a zero-second system response time. Therefore, data for the 0.00 sec delay condition were excluded from the analysis since its inclusion would have artificially inflated the resulting mean square values. This data correction is evidenced by a decrease in the available degrees of freedom from $F(6,72)$ to $F(5,60)$.

variance. To overcome these shortcomings, we performed a second experiment.

EXPERIMENT 2 – REFINEMENT

This experiment was designed to more closely investigate the region corresponding to the second strategy, pacing. This region is of particular interest since the strategy that the user applies attempts to actively compensate for any delay and requires the user to keep track of the actual delay involved. The delays in the present experiment were restricted to a 3.00 sec range centered on each subject's pacing region, the location of which was determined through a normalization procedure.

Method

Subjects. Twelve subjects were recruited from the general university population. No computer experience was necessary. All subjects were naive regarding the experimental manipulation and were fully debriefed following the experiment. Subjects were paid for their participation.

Apparatus and Materials. This experiment used the same apparatus and materials as experiment 1.

Experimental Design. Delay conditions were set dynamically, based on a pretest using a 0.00 sec delay. The subject's median initial keystroke latency (κ) from the pretest was used as a base point to set the actual experimental conditions. Delays were set at 0.250 sec intervals from ($\kappa - 0.250$) to ($\kappa + 2.50$) secs. Delays beginning at ($\kappa - 0.250$) sec and increasing at 0.50 sec intervals were considered part of condition set one, while the remaining delays were considered part of condition set two. Finally, each data set consisted of 75 rather than 90 three-digit numbers. A Latin square design was used to counterbalance potential ordering effects. Subjects were randomly assigned to orders.

Procedure. This experiment was conducted in the same manner as Experiment 1 with the exception that the experiment was broken into two sessions with each session consisting of six delay conditions. This was done to reduce potential fatigue effects. Subjects participated in sessions on consecutive days at the same relative time of day.

Results

Exploratory data analysis revealed that subjects did pace the system in a relatively stable region. Individual differences were substantially less than those observed in the first experiment. Subject within-condition performance was examined and was again found to stabilize after the first fifteen data points. The first fifteen data points for each condition were therefore removed prior to statistical analysis.

An analysis of variance was performed for the variables of interest. The analysis revealed significant effects for initial keystroke latency ($F[11,111] = 27.38, p < .001$), anticipation errors ($F[11,111] = 8.86, p < .001$), and initial keystroke variance ($F[11,111] = 33.70, p < .001$). Subsequent keystroke latencies and keying errors were not significant. Ordering and session effects were not significant for any of the variables. In addition, a t-test comparing subject performance across pretests was not significant, $t < 1$. Initial keystroke latency, anticipation error, and initial keystroke variance results are presented in Figure 7.

Tukey's HSD revealed significant differences between five system delay regions (in relation to the subject's median performance) for initial keystroke latency and initial keystroke variance – region 1 [0.25, 0.00], region 2: [0.25, 0.50], region 3: [0.75, 1.75], region 4: [2.00, 2.25], and region 5: [2.50] secs. Similar results were found for anticipation errors with one modification, the third delay region covered a narrow range – 0.75 to 1.50 secs – while the fourth region extended from 1.75 to 2.25 secs. These differences were significant at the $p < .001$ level. There were no significant within-region differences.

Discussion

The results of this experiment identify a pacing region that lies between 0.75 and 1.75 sec. For delays in this region users appear to have little difficulty pacing the system and are able to maintain apparently optimal levels of response latency and error. The rise in response latency and variance at

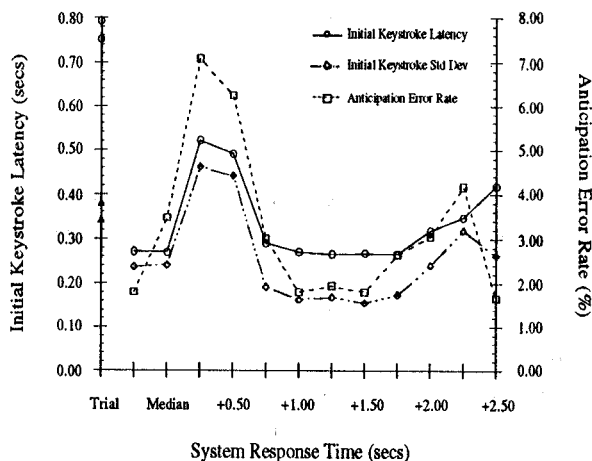


Figure 7. Double y-axis plot detailing the results for Experiment 2 with initial keystroke latency, the standard deviation of the initial keystroke latency, and anticipation error rate plotted against system response time.

longer delays (as well as the sudden and significant drop in error rate at the longest delay) are also in accord with the model presented earlier. As users lose control over their ability to pace the system (*viz.* the increase in variance), errors rise to an unacceptable level and a different strategy (monitoring) becomes preferable.

An unexpected finding was the discovery of significant user difficulty at the crossover between automatic performance and pacing. It is not immediately apparent why this occurs. A possible explanation is that engaging a strategy produces a "blind spot" that interferes with performance. The very high variance at this point suggests that users might be having difficulty consistently using a single strategy, perhaps because the difference in cost is too small to gauge accurately. Alternately, performance difficulties might be due to problems in engaging an internal timer for the short intervals required. Further investigation would be necessary to arrive at an acceptable account.

CONCLUSIONS

User performance is systematically affected by system delays. Results indicate that users choose task strategies suited to a given system delay, with the choice apparently based on the relative cost of a strategy at that given delay. A closer examination of delays inducing a pacing strategy revealed intervals where disproportionate difficulties occurred. The source of this latter effect is unclear at this time, but is consistent with a strategy-based account of performance.

Using this study as a basis, work is currently in progress to extend the performance model into an engineering model which can be used to derive pre-development, zero-parameter response time guidelines for specific tasks. Additional studies should be carried out to (1) evaluate system response time effects across different user modalities, (2) investigate the additional delays dimensions of predictability and variability, and (3) probe the process of strategy engagement.

Development of a model that accounts for the influence of system delay on user performance would support system engineers in the design of new systems and provide additional guidelines for the formulation of system management policies. Our goal is to lower system development and maintenance costs through the inclusion of additional design parameters based on system delay. The inclusion of such parameters will help ensure user acceptance of and satisfaction with new systems, as well as improve productivity.

REFERENCES

1. Barber, R. E., & Lucas, H. C., Jr. (1983). System response time, operator productivity, and job satisfaction. *Communications of the ACM*, 26(11), 972-986.
2. Bergman, H., Brinkman, A., & Koelega, H. S. (1981). System response time and problem solving behavior. *Proceedings of the 25th Annual Meeting of the Human Factors Society*, 749-753.
3. Boies, S. J. (1974). User behavior on an interactive computer system. *IBM Systems Journal*, 13(1), 2-18.
4. Butler, T. W. (1983). Computer response time and user performance. *Proceedings of the Annual Meeting of the ACM SIGCHI (SIGCHI'83)*, 58-62.
5. Dannenbring, G. L. (1983). The effect of computer response time on user performance and satisfaction: A preliminary investigation. *Behavior Research Methods & Instrumentation*, 15(2), 213-216.
6. Dannenbring, G. L. (1984). System response time and user performance. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14(3), 473-478.
7. Goodman, T. J., & Spence, R. (1981). The effect of computer system response time variability on interactive graphical problem solving. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11(3), 207-216.
8. Grossberg, M., Wiesen, R. A., & Yntema, D. B. (1976). An experiment on problem solving with delayed computer responses. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(3), 219-222.
9. Johnson, E. J., & Payne, J. W. (1985). Effort and accuracy in choice. *Management Science*, 31(4), 395-414.
10. Kleinmuntz, D. N., & Schkade, D. A. (1989). *The cognitive implications of information displays in computer-supported decision making* (Working Paper 88/89-4-11). Austin, TX: University of Texas at Austin, Graduate School of Business.
11. Kosmatka, L. J. (1984). A user challenges the value of subsecond response time. *Computerworld*, 18(24), ID 1-8.
12. Kuhmann, W. (1989). Experimental investigation of stress-inducing properties of system response times. *Ergonomics*, 32(3), 271-280.
13. Lambert, G. N. (1984). A comparative study of system response time on program developer productivity. *IBM Systems Journal*, 23(1), 36-43.
14. Martin, G. L., & Corl, K. G. (1986). System response time effects on user productivity. *Behaviour and Information Technology*, 5(1), 3-13.
15. Miller, R. B. (1968). Response time in man-computer conversational transactions. *Proceedings of the AFIPS Fall Joint Computer Conference*, 33, 267-277.
16. Näätänen, R., Muranen, V., & Merisalo, A. (1974). Timing of expectancy peak in simple reaction time situation. *Acta Psychologica*, 38(6), 461-470.
17. Pachella, R. G. (1974). The interpretation of reaction time in information-processing research. In B. H. Kantowitz (Ed.), *Human information processing: Tutorials in performance and cognition* (pp. 41-82). Hillsdale, NJ: Lawrence Erlbaum Associates.
18. Rudnicky, A. I., & Quirin, J. L. (1990). *Subjective reaction to system response delay: A pilot study*. Unpublished manuscript, Carnegie Mellon University, School of Computer Science, Pittsburgh.
19. Rushinek, A., & Rushinek, S. F. (1986). What makes users happy? *Communications of the ACM*, 29(7), pp. 594-598.
20. Sherwood, B. A., & Sherwood, J. N. (1988). *The cT Language*. Pittsburgh: Carnegie Mellon University.
21. Siegler, R. S., & Jenkins, E. (1989). *How children discover new strategies*. Hillsdale, NJ: Lawrence Erlbaum Associates.
22. Shneiderman, B. (1987). Response time and display rate. *Designing the user interface: Strategies for effective human-computer interaction* (pp. 272-309). Reading, MA: Addison-Wesley.
23. Shneiderman, B. (1991). Human values and the future of technology: A declaration of responsibility. *ACM SIGCHI Bulletin*, 23(1), 6-10.
24. Smith, D. (1983). A business case for subsecond response time: Faster is better. *Computerworld*, 17(16), ID 1-11.
25. Thadhani, A. J. (1981). Interactive user productivity. *IBM Systems Journal*, 20(4), 407-423.
26. Wickens, C. D. (1984). *Engineering psychology and human performance*. Glenview, IL: Scott, Foresman and Company.