THE EFFECTS OF ERROR NOTIFICATION TIMING ON ERROR CORRETION IN ROUTINE DATA ENTRY

by

Kenneth Marr

Barry Floyd

THE EFFECTS OF ERROR NOTIFICATION TIMING ON ERROR CORRECTION IN ROUTINE DATA ENTRY

by

Kenneth Marr
Leonard N. Stern School of Business
Information Systems Department
New York University
90 Trinity Place
New York, NY 10006

and

Barry Floyd
Leonard N. Stern School of Business
Information Systems Department
New York University
40 West 4th Street
New York, NY 10003

January 1990

Center for Research on Information Systems
Information Systems Department
Leonard N. Stern School of Business
New York University

Working Paper Series

STERN IS-90-1

Abstract

Experienced data entry operators participated in an experiment to evaluate the effects of the error notification timing on error correction performance in the data entry task. Three timing strategies were studied: immediate error notification, notification at the end of a field, and notification at the end of a physical line. Based on a model of data entry and a model of the correction process hypotheses were developed to predict operator performance. It was conjectured that operators treat individual fields as separate units tasks and that interrupting within a task would be more disruptive than interrupting between tasks. The results of the study indicated that the error rates during correction were smaller for the end of line treatment although the time to complete the correction was longer for this treatment than the other two. Performance was essentially the same for those operators interrupted who were interrupted immediately and those who were interrupted at the end of a field.

1 Introduction

When a person performs a computer-mediated task, rarely is the performance of that task error free. Errors are often made even by the most experienced user. These errors are usually detected and corrected in a routine fashion by the user. In many cases, however, the error passes unnoticed by the user and so the system (when capable) detects the error and sends an error notification message. The process used to notify the user of the error and to assist in the error's correction is an important component in the user's performance with and acceptance of the system. One aspect of this process is the timing of error message notification. Little research has been performed on error message timing (though see Segal, 1975), though some work has been performed on the effects of task interruption (Field, 1987; Kreifeldt and McCarthy, 1981) and its impact on user performance. In this paper we present a study where we address the issue of error message timing and its effect on error correction in the routine keying task of data entry.

An error message may be sent by the computer system at different points in the task, ranging from immediately following an erroneous keystroke to much later, following the completion of some set of keystrokes. For example, in command language entry, error notification typically is given after the user has entered the complete command. In transcription typing tasks, users often run a spelling checker after typing the entire document – though many checkers offer end of word notification. In commercial data entry, error notification usually occurs at the end of a field.

In deciding whether to notify a user of an error immediately versus waiting to some later time, the designer contrasts probable user performance and system performance under each scenario. In some instances the choice is clear. For example, if the consequences of allowing erroneous user actions to remain uncorrected are very negative, then avoiding them may dictate an immediate notification strategy (e.g., air flight control, nuclear power plant operations). However, with less negative contingencies, the timing of the error message is more problematic. Factors which the designer must contend with in predicting user performance under different interruption scenarios include (1) identifying what the user is doing when he is interrupted, (2) the type of error made while performing that task (or subtask), (3) the impact of interruption on that and subsequent activity, and (4) identifying what the user must do to correct the error under the different timing scenarios.

Describing what the user is doing when he is interrupted can be partially handled by identifying whether it is a problem solving task or a routine task for that particular user. In our study we concentrate our attention on the experienced data entry operator performing a routine task. While the study of novices may provide different results, we are interested here in exploring routine behavior.

We chose data entry as our task because keying tasks of this nature have been extensively researched, thus providing a foundation for describing user activities. We focused on data entry (rather than transcription typing, for example), because of the particular learning effects which we believe occurs in operators from repeatedly keying data from the same document structure. Data entry operators key from the same document structure often for thousands of documents. Thus, by choosing skilled operators in performing a repetitive task of this nature we can make better predictions regarding what the operator is doing at certain times in the data entry activity and explore

the impact of various notification timing strategies on operator performance.

We hypothesize that unit task boundaries are the best locations for providing error notification messages for tasks where the consequences are not severe enough to warrant immediate notification. In particular, we suspect that keying a field is the unit task in data entry and wish to empirically verify that what is currently done in industry (notification at field boundaries) is preferable to immediate interruption or to delaying the notification message to some later time (say at the end of a physical line).

The type of error is important. In some instances, errors of incorrect goal formulation may dictate a more immediate error notification strategy while errors in execution (i.e., referred as slips, Norman, 1981) may allow for delaying notification. We discuss this issue more fully in the next section. The types of errors data entry operators confront most often include transpositions in characters, missing characters, extra characters, and wrong characters. In our study, we assess the impact of various error message timing strategies following these errors of execution during data entry. We contrast user performance on various time measures and on error rates. In doing so, we take into account the various demands different timing message strategies place on the user (e.g., reorientation to the error location during delayed notification).

The remainder of this paper is divided into four additional sections. In Section 2, we discuss work on task interruption, including a discussion of a study performed by Segal (1975). In Section 3, we draw on previous research to describe the task of data entry and describe the demands which different error notification strategies place on the user performing an error correction process. We follow with a presentation of our study in Section 4 and conclude with a discussion of this work in Section 5.

2 Research on Task Interruption

Research on task interruption dates back to the 1920's with work done by Zeigarnik (1927). In this work, she explored the effect of interruptions on recall, showing that interrupted tasks were recalled more often than uninterrupted ones. The type of tasks Zeigarnik explored included cognitive, problem solving ones as well as manual tasks. More recent studies using

computer based tasks have focused on how well people can resume an activity given that an interruption has taken place (Kreifeldt and McCarthy, 1981; Field, 1987) and on the effects of interruption on overall task performance (Segal, 1975) Each of these studies was concerned with comparing different interface designs with Field suggesting that interruption performance is a useful dependent variable in comparative evaluations.

In the Kreifeldt and McCarthy (1981) study, the authors compared the effects of interruption on subjects using a Reverse Polish Notation calculator with those subjects using an Algebraic Notation calculator. The subjects, given problems of differing complexity, were interrupted 12 seconds into the task. During the interruption, the subjects were asked to write down multiplication tables for a 1 minute period. The dependent measure evaluated in the study was subject performance subsequent to interruption. They found that the group using the Algebraic Notation calculator completed the task in a shorter amount of time. They also found that there was a general increase in the time to complete the task for both groups, which they attributed to the interruption. In their study, the exact location of the interruption in the task activity was not controlled for. Thus, the subjects could have been keying a planned solution, still planning a solution, etc.

In a study on the effects of interruption in an information retrieval task using a hierarchical menu interface, Field (1987) claims that his interruptions affected user activities following task interruption. In his experiment, subjects were given an interface which allowed the user to review either all of the previous screens seen or to review a subset of screens (i.e., the last one viewed and the first one viewed). His interruptions occurred at a specific point in the information search task and consisted of various activities such as completing a numeric sequence or looking up the title of a book. Unfortunately, his report does not contrast activities which were interrupted to those which were not, and so, it is possible that the differences he found could be traced to the screen review treatment rather than to the effects of interruption as stated.

Segal (1975), in a study on the timing of error correction messages, examined user performance on two keying tasks with error notification occurring immediately after an incorrect keystroke or after the subject had completed the entire task. One task required subjects to enter 20 different permutations of the string 'abcde' with the requirements of 'c' preceding 'a' and 'b' pre-

ceding 'd'. The second task required the subjects to enter 25 of the 50 states in the United States. Segal's results were mixed. He found that the subjects performing the permutation task completed the task more quickly and more accurately when notified of errors immediately than those subjects who were notified at the end of a permutation. For the state naming task, he found that subjects took a shorter amount of time when immediately notified of an error but made a larger number of errors than those subjects who were notified at the end of a name.

The permutation task can be characterized as problem solving where the user must determine which character to enter next. Unless the user follows a strategy which generates the entries, he must use a trial and error approach using the characters displayed on the screen and any error message as feedback. In such a task, immediate feedback provides additional utility to the user by preventing him from following an incorrect solution path. When given an immediate message, the user need only backspace one key and rethink his strategy (which is correct up to that point). On the other hand, a message which arrives at the end of an entry has allowed the subject to continue with characters which are incorrect.

The probable psychological units which the subject manipulates in this permutation task are individual characters. Because this is not a familiar task, the subjects have not developed any higher level cognitive structures (e.g., chunks) which are found in experienced users in many problem domains. We suspect that an analysis of the time between user keystrokes would show significant pauses between characters. The time spent during this pause reflects the user planning the next character to be entered (see Robertson and Black, 1986). Interrupting during these pauses and providing immediate feedback on the character just entered would be useful given our characterization of the task. A message at the end of the task forces the users to begin all over again.

In the state naming task, the psychological units manipulated by the subject are complete words such as 'Maine' or 'Texas'[1] with some chunks being two words such as 'New Mexico'. Thus when keying begins we would expect it to be smooth and fairly quick, as described by many of the transcription typing models (e.g., Rumelhart and Norman, 1982). In such models, a per-

[1]With some flexibility for states like Mississippi, Pennsylvania and California

son places a high level representation of the material (in this case a word) to be typed into a motor output store and then decodes this representation for execution at a lower level. Though not reported by Segal, we expect that the time between keystrokes for this naming task was small. We also expect that interruption in his task meant the interruption of these motor programs. Since the correct completion of keying a name does not rely on the accuracy of the first part of the word, the additional utility of notifying the user of the error immediately (i.e., he need only move back one keystroke to make the correction) must be balanced against the ability of the user to stop, correct the error and continue. Segal found this process to be more error prone.

Thus, the research presented suggests that interruption affects user performance. However, unclear in each of the studies was a detailed analysis relating subject activities to the timing of the error message. We expect that this analysis is important for evaluating different message timings. We do this for the task of data entry.

3 The Task of Data Entry

In modeling the processes of a keying task, one perspective is to view the process serially where the activities which are necessary to create the keystroke operate in sequence. Thus, the activities leading up to the pressing of the key, the pressing of the key, and the activities following the pressing of the key are discrete and non-overlapping. Such a model describes the permutation task in Segal's study where much problem solving behavior is necessary for determining each keystroke. Interruption in these tasks between keystrokes is disrupting an activity such as problem solving or planning which Segal found to result in better subject performance when compared with delayed error notification. In essence, incorrect plans were being interrupted.

In routine keying tasks such as transcription typing, researchers have found that a discrete, character by character model is inaccurate. A parallel processing model where characters flow continuously through a set of typing stages is more descriptive. For example, video tapes of experienced typists (Norman and Rumelhart, 1983) show that the physical movement of the hands and fingers is a highly parallel activity. The tapes revealed that for a word like '*vacuum*', the right-hand's index finger was positioned over the

'*u*' even before the '*a*' had been typed. Studies examining the time to start typing and the time between keystrokes during sustained typing also show that much mental activity is done in parallel (Larochell 83, Logan 82). These activities result in about a one second lag between the time the eyes perceive a character and the fingers type it (Logan 83, p. 199). The parallel mental processes thus resemble a pipeline through which characters flow (Salthouse 84). Interruptions during tasks such as these are ones which are interrupting multiple stages of the keying process. They interrupt the perception of the new material, the loading of the output store, and the execution of the motor programs. Whether one is better than another is task dependent (for example, how contingent is execution of the remaining chunk of keystrokes on the success of the first chunk of keystrokes). However, it is important to know what is being interrupted to make the assessment.

Our task of data entry is a routine keying task (for our subjects), but differs from transcription typing in that data entry documents are designed around *fields* which remain constant in their location on the document. Thus, unlike transcription typing where words are rarely in the same place twice, data entry operators key from a consistent pattern of field locations, often keying from the same document form thousands of times. While the contents of the fields differ from document to document, each field has distinguishing characteristics. A field may consist of only numeric data or only alpha data or both. The field may be perceived as difficult to key due to the nature of the set of characters which usually occurs in the field. And, each field has a particular length associated with it which assists the operator in planning the execution of the task. Thus, constant field position (and thus a constant order in the keying pattern) coupled with different keying demands placed on the operator by different fields suggests that fields play a primary role in the way the operator thinks about and does the task of data entry. We conjecture that the operator treats each field as a separate unit task in the performance of data entry.

Research in other domains has shown that people learn consistent patterns in the task domain and use them to organize and plan the execution of the task (Reber 1967; Chase and Simon 1973; Reitman 1976; Egan and Schwartz 1979; Reitman and Reuter 1980). In particular, in research on the data entry task, Neal (1977) reports evidence showing that operators use fields as keying boundaries. Neal used experienced data entry operators and

found that their interkeystroke times between fields were longer than within fields and were longer still between records (lines of data).

Thus, in assessing performance in error correction in data entry, we look at the effect of different error correction strategies based on this understanding of what it is that is being interrupted. If an error message occurs within a field, then we expect that the interruption occurs during keying, and that many multiple stages in the keying process are interrupted. Our conjecture is that interruption within a field causes deterioration in operator performance because the operator is working towards closure in completing execution of that unit task. If the operator is interrupted between fields, then the interruption affects the planning and preparation (e.g., viewing the field, loading the cognitive buffer before execution of the motor programs) for the keying of the following field. Thus, we contrast *immediate interruption* with two different between-field strategies: one where the system sends the error message at the *end of a field* containing the error, and one where the system sends the error message at the *end of a line* containing the field in which the operator made an error.

Finally, it is clear that different error timing strategies place different demands on the user during the correction process. If an error message occurs immediately, the operator is focused on the errant material. If the error message is delayed, the operator must reorient himself to the location of the error, which takes operator time. We address these issues in the next section where we describe the error correction process.

3.1 The Error Correction Process

There are five distinct stages in the error correction process: *creating* the error, *detecting* the error, *removing* the erroneous keystrokes, *entering* the correct keystrokes and *resuming* the keying process (see Card, Moran, and Newell 1983, pg. 177). While most of these steps are self-explanatory, we mention a few characteristics of these steps which are typical of data entry work beginning with the error.

While there are many different types of errors, for the work addressed in this paper we study errors in execution. These types of errors include transpositions in characters, missing characters, extra characters and wrong

characters. Once having created an error, operators are fairly adept at detecting it (For example, Shaffer and Hardwick (1969, pg. 212) found that experienced typists catch about half their keying errors.). In removing errors, if the error was system detected, typical data entry systems erase the field containing the error for the user. If the operator detects the error, he usually has two methods which he can use to delete characters - the backspace key and the field delete key. The backspace key removes characters as it moves the cursor back one space. The field delete key removes all characters from the field and repositions the cursor at the beginning of the field. Resuming the keying process follows the last correct keystroke during reentry and can differ according to the notification strategy.

There are two prototypical strategies in error notification timing which we identified in describing the Segal task. One is to interrupt the operator immediately while the second is to interrupt the operator at some later time. The advantage in interrupting the operator immediately is that he is focused on the location where the error has occurred and so does not need to search to match his input source to the error displayed on the screen. Furthermore, the effort to remove keystrokes may be small. However, disadvantages include not allowing the operator to detect his own errors and the potentially negative effect of distracting the user via interruption and not allowing him to reach closure on keying that particular field.

In delaying error notification, the designer has chosen to avoid whatever distraction effects may occur until some future time. The penalty for doing so, however, is the cost the operator faces in returning to the point where the error occurred and matching it to the source document. This takes additional time and allows for errors in matching the correct location on the source document to the location on the screen where the error occurred. Thus in determining optimal performance, the effects of disruption must be contrasted with the effects that differing notification strategies may cause as an artifact of the design.

In the next section we describe a study which we conducted to test our conjecture about the effects of differing error notification timing strategies.

4 Experimental Method

4.1 Subjects

Twelve[2] subjects participated in this study. Each was a professional data entry operator recruited from a local agency specializing in temporary assignments. The average age of a subject was 34.5 years with a range of 18-49. The average years of experience for the group was 8.45 years with a range of 1 year to 18 years. Ten of the subjects were female. We paid the agency $12.50 per hour per subject with the agency paying subjects their normal commercial wages.

4.2 Apparatus and Materials

The experimental environment consisted of Zenith model 158 PC's with hard disks in a fairly spacious lab setting, a set of documents containing data, and a computer program designed to allow data entry, to provide an error notification message when an error occurred, and to collect information regarding operator performance.

The PC's consisted of an 8088-2 (8 MHz) processor, a standard PC keyboard and a color (CGA) monitor. We selected the standard keyboard because it closely resembles keyboards with which our subject population is familiar because the numeric pad is immediately to the right of the alpha keys. The processor was fast enough to allow data entry and to collect data on individual keystrokes without any noticeable delay to the subjects. The PC contained a clock with a precision of 0.055 seconds which we used for timestamping keystrokes.

The data entry program presented an image of the data entry document on the screen. It accepted and recorded every keystroke in a file along with the time the keystroke was made. The program also detected every error that the operator made and recorded these as well. The program was capable of detecting all errors because the data used to generate the data entry documents was stored online for immediate comparison. Finally, the program

[2]Only eleven subjects completed the experiment - one subject became ill and left after 2 hours of work. This subject's data is not included in the analysis.

sent an error message to the operator according to one of three predefined strategies (e.g., the end of a physical line).

The error notification message consisted of three parts. First, because operators are generally looking at the document, not the screen, the system sounded an audio tone. Second, the system displayed a copy of the erroneous field on the right side of the screen (the data entry display was confined to the left side with the right side reserved for error display). Third, it removed all the characters from the field in which the error was made, and repositioned the cursor at the beginning of that field. The system would not allow operators to continue with normal data entry until they had first corrected the error (i.e., correctly retyping the field's contents). After the operator completed correction of the error, the system automatically set the cursor at the beginning of the field in which data entry was to resume.

We used only one form of source document as the stimulus for the experiment. The document form consisted of 70 fields (13 of these fields were alpha fields, 57 were numeric fields) with a maximum of 337 characters per document and an average field length of 4.81 characters and a minimum field length of 2 characters. Typically a document contained only 306 characters (i.e., all of the fields contained data, however, some alpha data items only partially filled the field). These documents are typical of the data entry documents used in industry, consisting of both alphanumeric and numeric only fields. Two hundred of these documents, each containing different data values, were provided to the subjects in 'batches' of 25.

4.3 Design

The twelve subjects were randomly assigned to one of three experimental treatments (i.e., four subjects per treatment). Subjects in the first treatment group were notified of an error immediately, subjects in the second treatment group were notified of an error at the end of the field in which the error occurred, and subjects in the third treatment were notified of an error at the end of the physical line in which the error occurred. We will refer to these three groups of subjects as IMM, FIELD and LINE respectively.

We chose a 'between subject' design to remove any disruptive effects which such diverse error message strategies may pose to an operator who

works on them in sequence. The subjects also worked on the system for 7 hours allowing them to become acclimated to the error notification strategy and to the document structure.

Within this between subject design, for the end-of-field error notification treatment (FIELD) and for the end-of-line error notification treatment (LINE) we differentiate between error events where the *operator* detects the error and those where the *system* detects and notifies the operator of the error. For the immediate treatment, we are unable to determine whether the operator detected the error before the error message occurred because the error message was instantaneous[3].

In addition to contrasting differences among our three treatments, we contrast *operator* detected and *system* detect error events, to assess how disruptive the computer generated error message is to the individual operator. For example, if the operator is able to correct self-detected errors more quickly and accurately than errors which are detected by the system, then the system message may be a disruptive influence to operator performance.

A third factor in our experiment consisted of designing into the data entry system one 'error' per document. These forced 'errors' consisted of the system notifying the operator that an error occurred even though the operator had correctly typed the data. The system triggered a single such 'error' for each document, but at a different place. That is, when keying a particular document, all operators had the error at the same place, but the location of the error was different for each document. This provided us with a minimum set of error events for even the most accurate operator. It also allowed us to compare operator performance on the exact same set of error locations for a subset of the system-detected error condition. These designed errors were not placed in the first two documents.

To test our hypothesis regarding the effect of the timing of error messages on operator performance we gathered data on the time it took each operator to do various subtasks associated with error detection and correction and each operator's error rate in these subtasks. These measures are described in Table 3.1[4].

[3]It is possible for an operator to detect that he is going to make an error before the key is actually pressed (Rabbitt, 1978).

[4]Fields consisting of 2 characters were excluded from analysis for the calculation of system detected errors rates.

Measure	*Description*
Error Measures	
Error rate during data entry.	Number of errors divided by the number of error-free keystrokes.
Error rate during error correction:	
System-detected errors	Number of incorrect keystrokesdivided by the total number of keystrokes made to reenter field.
Self-detected errors	Number of incorrect keystrokes divided by the number of keystrokes made to reposition cursor at point where user began backspacing.
Timing Measures	
Time to detect	Time from last erroneous keystroke to the first removal keystroke (e.g., a backspace)
Time to remove	Time between first error removal keystroke and last removal keystroke
Time to respond	Time between system message and first keystroke to correct.
Keying rate during correction	Time between first and last keystroke of error correction divided by the number of characters during error correction.
Time to resume	Time between last keystroke of correction and first keystroke of next field for system detected errors and the time between last keystroke correcting an error and the following keystroke for operator detected errors.

Table 4.1: Performance measures.

4.4 Procedure

The experimental setting closely resembled a real data entry work environment. Subjects were run in three separate sessions in groups of 3 to 5 individuals over the course of two weeks. A session lasted about eight hours including two fifteen minute breaks and a half-hour lunch period (the subjects were allowed to choose the time of the break and lunch). Subjects were also allowed to leave the terminal periodically to stretch and for any other necessary activities.

At the beginning of the session, we told the subjects that they would be working on an experimental data entry system. We explained how to

use the terminal, reviewing the field release, character delete and field delete keys. We also explained that occasionally there may be some data transmission errors (our forced 'errors') and that they should treat the situation as a straightforward error condition and simply rekey the 'erroneous' field. For each treatment we reviewed the error correcting procedure. For the IMM treatment the procedure was to reenter the data starting at the beginning of the field in which they were working and not just the last (erroneous) character. For the FIELD treatment the procedure also consisted of reentering the field that they had just finished. For the LINE treatment the procedure included examining the screen to determine just which field was to be reentered, finding that field on the document, and then reentering the complete field.

The documents were grouped together in batches of 25 and all the subjects were given the first batch during the instruction period. When a subject completed one batch he came to the room supervisor who gave him the next batch of documents. As the operators keyed the data into the system, the data entry program collected every keystroke along with the time of that keystroke. The program then compared the keystroke with the correct one stored internally. If the keystroke was in error and the operator did not correct it, the system notified him according to the treatment he was in. After notification, the system required that the operator correct the error before allowing him to proceed with further data entry.

5 Results

5.1 Performance during routine data entry

Table 5.1 shows operator productivity by treatment group listing documents completed and average keying speed within fields, between fields and between lines. The operators in the LINE treatment were slower, completing an average of 57 documents and having an interkeystroke time of 483.2 ms within fields (this value reflects keystroke times for error free fields). The operators in the FIELD treatment had an average interkeystroke time of 353.7 ms within fields - 36% faster than the LINE subjects. The IMM subjects fell about midway between these two groups with an average interkeystroke time

of 439.1 ms.

The overall average time between keystrokes within a field (415ms) corresponds to a keying speed of 29 words per minute. This is slow for experienced transcription typists but is in line with other published data on the keying speeds of data entry professionals (Bailey, 1982).

Treatment Group	*Average Documents Per Operator*	*Average Time Between Keystrokes Within Fields*	*Average Time Between Fields*	*Average Time Between Lines*
Immediate	65.0	439.1	1,254	2,131
End of Field	72.5	353.7	1,206	2,210
End of Line	57.0	483.2	1,602	2,687
Overall	65.5	415.2	1,317	2,286

Table 5.1: Operator performance during data entry (time is in milliseconds).

The time between fields for our subjects was significantly longer than the time between characters within fields. On average the time between fields was 1315 ms, or 901.8 ms longer than the average time between characters within a field. Part of this time may be accounted for by time required to reposition hands when changing from one field type to another (e.g., alpha to numeric). The smallest of these times, moving from numeric to numeric when no hand change is required, takes 1,104 ms which is 668 ms longer than the average time between keystrokes within fields. Table 5.2 shows the average time between the two different field types according to the direction of change (e.g., alpha to numeric). We interpret the additional time between fields which is unaccounted for by repositioning hands as indicative that the subjects are treating the fields as unit tasks.

Hand Change	*Time*
Alpha to Alpha	1,728
Alpha to Numeric	2,125
Numeric to Alpha	2,249
Numeric to Numeric	1,104

Table 5.2: Operator times for between field hand changes (time is in milliseconds).

Table 5.3 presents error rates for the three treatment groups for each of three tasks: routine data entry, correcting self-detected errors, and correcting system-detected errors. In this section we discuss the first task: error rates during routine data entry. For this task, the overall average error rate was 2.41% (i.e., 2.41 erroneous keystrokes occurred when entering

100 keystrokes). In comparing the treatment groups, the LINE subjects were more accurate (error rate: 1.72%) than either the IMM subjects (error rate: 2.44%) or the FIELD subjects (error rate: 2.78%). These differences were statistically significant (partial F for treatment effects is 38.93 with df=720,2; P-value = 0.0001). Furthermore, at the 1% level, Scheffe's test for simultaneous confidence levels shows all three treatment groups to be statistically different from one another. These error rates are based on all errors made by the operators during routine data entry[5], including errors the operator caught (1.54%) and errors the system caught (.87%). Industry reports system-detected error rates of about 1% (Morin 86). Thus, the study's operators were somewhat more accurate than the general industry.

5.2 Performance during error correction

In correcting self-detected errors, subjects were more accurate (average error rate was .55%) than when they were routinely keying data[6]. However, there was no statistically significant difference between the LINE and FIELD groups during the correction of self-detected errors (F=.36). Thus, although the LINE and FIELD groups differed in the overall number of errors made during routine data entry, when they went about correcting errors, the error rate during this process was about the same.

Treatment Group	*Error rate during data entry*	*Error rate during correction: for self-detected errors*	*Error rate during correction: for system-detected errors*
Immediate	2.44	N.A.	4.11
End of Field	2.78	0.60	4.56
End of Line	1.72	0.41	1.85
Overall	2.41	0.55	4.04

Table 5.3: Error analysis of operator performance (error rate: errors/100 keystrokes).

There are significant differences though between the error rates during routine data entry and those during the correction of self-detected errors for both groups. For the LINE group, the difference in error rates between normal data entry (1.72%) and when correcting self-detected errors (0.41%) was

[5]However, this value does not include errors which occur during error correction

[6]The IMM treatment group is not included in this analysis because they did not have the opportunity to detect their own errors.

1.31%; this difference is statistically significant (t=6.255). For the FIELD group, the difference between regular (2.78%) and self-detected (0.60%) was 1.18%; this difference was also statistically significant (t=10.983). Thus, keying during correction of self-detected errors was more error free than during routine data entry plus not being interrupted by an external source. The majority of this effect can be attributed to the small number of keystrokes (usually one or two) entered during correction of self-detected errors. Keying only one character, for example, precludes some errors such as transpositions.

Performance during the correction of system notified errors, however, is significantly worse. In correcting system-detected errors, the IMM subjects were almost twice as error prone as during normal data entry (error rates of 4.11% vs. 2.44%; t=3.832). The FIELD group subjects were also about twice as error prone during the correction of system-detected errors as during routine data entry (error rates of 4.56% vs. 2.78%; t=3.812). In addition, FIELD subjects were over 7 times more error prone during the correction of system-detected errors than during correction of self-detected error (error rates of 4.56% vs. 0.6%; t=16.514). The LINE group, however, remained at about the same error rate during the correction of system-detected errors as during normal data entry (1.85% vs. 1.72%; t=0.349), though they were also significantly more error prone during the correction of system-detected errors then during the correction of self-detected ones (error rates of 1.85% vs. 0.41%; t=5.431). Thus, the magnitude of the difference between the error rates during normal data entry and the error rates during the correction of system notified errors for the IMM and the FIELD group suggests that the timing of the error message is more disruptive for these subjects than for the LINE group and that the disruption for the IMM and the FIELD groups were about the same on this metric. Note that in all three treatments, the correction task required rekeying the complete field that contained the error.

5.3 Timing during error correction

In section 4.3, we identified five timing measures associated with error correction (see Table 4.1). These measures are time to detect the error, time to remove the erroneous keystrokes, time to respond to the error message, keying rate during correction of the error, and time to resume normal data entry. In this section we discuss subject performance on each of these mea-

sures. Within this discussion, again we will contrast performance on system detected errors to performance on subject detected errors. As described earlier, subject detected errors differ in that the subject (1) detects the error and so we have a measure for the detection time and (2) the subject must enter keystrokes to remove the error. In the system detected condition, the erroneous keystrokes are removed by the system (i.e., the complete field is erased when the message is provided). Thus, the time to detect and time to remove measures are replaced by a time to respond measure. Since the correction of system detected and self-detected errors requires slightly different activities, we present the detailed analyses separately.

5.3.1 System-detected errors and timing performance

Table 5.4 presents the timing data for our three measures: time to respond, keying speed during correction, and time to resume. For contrast, the table also shows keying speeds during routine entry.

Treatment Group	*Keying Speed During Routine Entry*	*Correcting System-Detected Errors*		
		Time to Respond	*Keying Speed During Correction*	*Time to Resume*
Immediate	439	2,029	533	1,578
End of Field	353	2,518	469	1,668
End of Line	483	3,763	517	3,073
Overall	415	2,342	522	1,728

Table 5.4: Timing Analysis of Correcting System Detected Errors (time is in milliseconds).

Operators took 2,342 ms to respond to the system error notification message. This is about twice as long as their average pause time between fields during data entry (see Table 5.1). There were differences between treatments due, at least in part, to differences in locating the error on the document. Although the erroneous field was highlighted on the screen and the cursor was positioned at the beginning of that field, the operators needed to locate the corresponding field on the document. The IMM and the FIELD treatment differed from the LINE in this subtask in that an operator's visual focus was already fixated on, or very near, the field which needed correction—it was either the field from which they were currently entering data (IMM) or the one from which they had just finished entering data (FIELD). The LINE group though had to shift focus to locate the erroneous field. Thus, it is not

unexpected that the LINE group took 80% longer to respond to the error message than either the IMM and FIELD groups. We might also expect the IMM group to respond faster, which they did; the FIELD group took 26% longer than the IMM group. The differences between the groups are statistically significant ($F_{(10,3727)}$=100.63) and Scheffe's test shows that all three groups are statistically discernible from one another at the $\alpha = 1\%$ level.

The keying speed during reentry averaged 522 ms which is slower than the keying speed observed during regular data entry (380 ms/keystroke). Furthermore, this pattern held for individual operators; every operator reentered data more slowly than he initially entered the data.

There were some group differences in how much each group slowed their keying rate (i.e., an increase in the interkeystroke time) during correction. Operators in the IMM and FIELD groups increased their interkeystroke times by about 100 ms, while those in the LINE group increased their interkeystroke time by only 30 ms. A conventional explanation for such slowing down is to improve accuracy. This is also in line with Rabbitt's work which shows that in many keying tasks operators, speed up until they create an error and then slow down, only to build up speed again until the next error. However, we found that although the operators in the IMM and FIELD groups slowed their reentry keying more than the LINE group, their error rates during reentry were much higher than the operators in the LINE group, who had not slowed down as much.

In resuming data entry, operators paused an average of 1,730 ms between the last keystroke of reentering the data and the first keystroke for new data entry. The LINE group paused for about twice as long as the other two groups. This difference in pause times was statistically significant (F=101) and Scheffe's test shows that, although the IMM and FIELD groups are not statistically distinguishable from one another, both are statistically distinguishable from the LINE group. The LINE operators probably used much of this extra time (1,460 ms)[7] to find the correct starting position on the document. For the IMM and FIELD groups, the data entry field was immediately following the field they had just corrected. Unlike the LINE group, FIELD and IMM operators did not need to search for the field as which to resume data entry. Thus, their time to resume corresponds very closely to their av-

[7]The combined mean of FIELD and IMM is 1,610; 3,070-1,610=1,460.

erage inter-field time (for IMM, 1,254ms vs. 1,578ms and for FIELD, 1,602 vs. 1,668ms). The time to resume data entry for the LINE is also consistent with this group's time between *lines* during regular data entry (3,073 ms vs. 2,286 ms).

5.3.2 Self-detected errors and timing performance

Operators detect their keying errors fairly quickly after making them. On average the FIELD group (see Table 5.5) detected a keying error and started backspacing 1,538 ms after having made the error. The LINE group took somewhat longer, taking 2,530 ms. This difference is statistically significant ($F_{(1,1736)}$=31.07). In evaluating subject performance on this measure we also reviewed the number of keystrokes made by the operators following the creation of an error. Overall, operators detect 60% of their errors immediately after making the error, not making any additional keystrokes after the erroneous one. In another 20% of the cases they detected their error after completing one additional keystroke and in the other 20%, operators entered an additional 2 or more keystrokes. Thus, operators averaged only .85 additional keystrokes between the error and the beginning of error removal. A comparison between the FIELD and LINE groups shows no statistically significant differences ($F_{(1,1736)}$=0.70), each group averaging the same number of keystrokes following self-detection of errors.

The time to remove these erroneous keystrokes varied from 950 ms for the FIELD group to 1,655 ms for the LINE group, a difference which is significantly different ($F_{(1,1768)}$=55.37). Thus, the LINE subjects were slower in backspacing during correcting self-detected errors.

	Correcting Self-Detected Errors			
Treatment Group	*Time to Detect*	*Time to Remove*	*Re-Entry Keying Speed*	*Time to Resume*
End of Field	1,538	950	321	480
End of Line	2,530	1,655	538	847
Overall	1,791	1,133	376	576

Table 5.5: Timing Analysis of Correcting Self-Detected Errors (time is in milliseconds).

When an operator removes his error immediately and does the reentry accurately, the pattern of keystrokes is: (Enter erroneous character) (Character delete) (Enter correct character) (Enter following character). Because there

is only a single reentry keystroke there is no measure of the time between keystrokes during reentry and hence, there can be no estimate of reentry keying speed. As noted, operators only required a single reentry keystroke in 60% of the cases. Thus, calculation of the reentry keying speed rests with the remaining 40% of the cases. Table 5.5 shows that the reentry keying speed of the LINE (538 ms) group is slower than that of the FIELD group (321 ms) but this difference (307 ms) has a probable value of only 1.8% ($F_{(1,666)}$=5.67). We believe the difference in reentry keying speed is not a treatment effect but is because the operators in the LINE group simply keyed more slowly than those in the FIELD group.

However, there are interesting differences in reentry keying speed between the self- and system-detected corrections. The FIELD group reentered the data faster when correcting self-detected errors than when correcting system-detected ones (321 ms vs. 469 ms). The difference (148 ms) is statistically significant (t=7.836, d.f.=1733). The LINE group though reentered data at about the same speed in both types of correction (538 ms vs. 517 ms; t=0.588, d.f.=496).

Overall operators resumed data entry in 576 ms. There were differences between the two groups with the FIELD group faster (480 ms) than the LINE group (847 ms). The difference (367 ms) was statistically significant ($F_{(1,1781)}$=19.92), though, again, we do not attribute this effect to the differences in treatment. We find that *both* groups resumed data entry 50% more slowly than their reentry keying speed. The FIELD group took 169 ms or 50% longer for the resumption keystroke than their time between keystrokes when reentering data entry than in reentering (480/321=150%); the LINE group took 309 ms or 57% longer (847/538=157%).

5.3.3 Analysis of Forced Errors

In our experimental design, we had included 1 'forced error' per document so that we could evaluate subject performance on the same set of errors. The results show that, although, on average, operators corrected 'forced' errors less accurately than regular ones (i.e., errors made by an operator), this difference (4.41% for 'forced' vs. 3.97% for regular errors) was not statistically significant (t=1.21). In our analysis of these errors, we found similar effects which have been discussed, thus, we have included these forced errors in the

overall analysis and have only presented the overall pooled results.

6 Discussion

Routine errors in the data entry task may be easily corrected by the operator, especially if the error is *detected by the operator.* In these instances the operator may backspace the required one or two characters, rekey the data and continue on with the form. However, when the system notifies the operator of the error, the error rate during correction increases significantly. For the two treatments where we can make the contrast (FIELD and LINE), the operator error rate was 2.5 times higher during correction of system detected errors than during operator detected errors for LINE operators and 7.5 times higher for FIELD operators.

Why are operator-detected errors corrected more accurately? We identify two reasons. First, the operator-detected error correction task is more parsimonious. In correcting self-detected errors, the operator can direct his attention only to the erroneous keystrokes. To correct the error, he backspaces to the error location and rekeys only one or two keystrokes. In the system-detected error condition the operator was required to re-key not only the erroneous characters but the entire field. This provides a greater opportunity to make an error, allowing, for example, for errors which occur due that particular sequence of keystrokes (e.g. capture errors).

Second, in correcting system-detected errors, an external interruption has occurred, forcing the operator to halt his current activities and to switch to an unplanned task. Although the operator knows what to do, the error notification message is still an abnormal event. Operators most often do not make mistakes - our operators received error messages on less than 1% of the keystrokes - thus, responding to an error message is not part of the normal routine and is disruptive.

In correcting system detected errors we proposed that the error message interruption would be more disruptive when the message arrived within the field than when the message arrived between fields or at the end of a line. That is, we hypothesized that interrupting the process of keying would be much more disruptive than interrupting during times when the operator would be reviewing the work completed and planning the keying for the

following field, activities which normally occur at task boundaries. By interrupting immediately we believed we would be stopping the operator from reaching closure on the completion of a unit task - the keying of a complete field. We thought that this disruption would manifest itself as an extended delay in beginning to respond as well as in a higher error rate during correction, and a longer time to resume normal keying activities. We found that there were very little differences in performance behavior in contrasting subjects in the IMM and FIELD treatments. However, in contrasting performance by the subjects in the LINE treatment to the IMM and FIELD treatments, we found that the IMM and FIELD treatment error rates during correction were significantly higher.

We believe that the error notification disrupted all of the operators, however, in the end of line treatment, there are two noticeable differences which may account for the lower error rate during correction. First, there is usually a longer pause at the end of line with the operator taking longer to get started on the next field (located on the next line). The difference between the activities at the end of a line and those at the end of a field is time required for a line-to-line transition, that is, recognizing that the line is complete and moving attention from the last field of one line to the first field of the following line. The time for this to occur, took on average, 2.2 seconds. This is approximately 1.0 second more than the time between fields. Thus the disruption effect of the error message is less, occuring during the time to make the line to line transition. Secondly, besides interrupting a different activity, the additional time required to find the erroneous field and to begin to correct it, allows time for the disrupted influence of the message to dissipate.

Another possibility in interpreting our results is that the operator who is interrupted immediately and the operator who is interrupted at the end of the field, retain in short term memory the keystrokes just entered. These character memory traces compete with the characters the operator sees when reviewing the field for reentry and result in conflict, thus increasing the likelihood of error[8]. In contrast, operators who are notified of an error at the

[8]An alternate, though similar, explanation is that the operator having in memory the keystrokes just entered, reviews the field only to see which keystrokes were in error and then rekeys the field using memory of the prior keystrokes plus the corrected ones. His memory, however, may be faulty resulting from the interruption (e.g., he recalls the keystrokes but

end of a line do not have these memory traces nor do they easily recall the contents of the field. Thus they review the field in a fashion similar to when they first see a field, resulting in an error rate similar to their routine error rate. This argument, though, is partially refuted by our the error rates found in the 'forced error' condition. These error rates were the same as those found in the 'normal' error condition. Thus, we would have found these error rates to be lower, which we didn't.

Why did we not see different results between the IMM and the END OF FIELD treatments? The outcome of our results does not support the notion that interrupting the operator between fields within a line is an optimal strategy. We thought that it would be best because it allows the operator to focus on material he has just keyed and because he has just completed a unit task uninterrupted. However, the effects of interruption for this task appear to override these issues. Alternately, our understanding of the keying task could be incorrect, in that subjects group fields into larger organizational units (especially those that are small and have similar characteristics such as being only numeric). While our distribution of pauses does not suggest this, the keying process and the creation of unit tasks could be more dynamic rather than simply one stable structure as we hypothesize.

In our design of the error correction procedure, we assured that the procedure was the same for each treatment. This, however, was punitive for the immediate interruption group. If the error can be detected immediately, a different correction procedure allowing the operator to only rekey the incorrect keystrokes would be more efficient. Such a design would mimic the self-detection error correction procedure where the operator rarely entered a field delete key, normally he would simply back up one or two keystrokes, deleting the error and then rekeying those one or two keystrokes. We are investigating this issue in a subsequent study.

In summary, we found that immediate and end of field interruption results in significantly poorer error correction performance as measured by errors rates to error notification which is delayed to a later time, i.e., end of line. When interrupting an operator at the end of line, the price that the operator pays for this lower error rate is an increase in the amount of time to find the erroneous field requiring correction. However, for the task of

---

the ordering information is incorrect), concluding with an increase in his error rate.

data entry, clearly the designer (and users of the data) would typically prefer optimization on error rates.

Bibliography

John R. Anderson (editor), (1981). *Cognitive Skills and Their Acquisition.* Lawrence Erlbaum Associates. Hillsdale, New Jersey.

John R. Anderson, (1982). Acquisition of cognitive skill. *Psychological Review.* 89(4):369-406.

John R. Anderson, (1983). *The Architecture of Cognition.* Harvard University Press, Cambridge, MA.

Robert W. Bailey, (1982). *Human Performance Engineering: A Guide for System Designers.* Prentice Hall, Inc., Englewood, New Jersey.

Robert W. Bailey, (1983). *Human Error in Computer Systems.* Prentice Hall, Inc., Englewood, New Jersey,

Norman Bodek, (1988). Keystrokes per hour. *The Data Entry Management Assoc. Newsletter.* 120, June.

Stuart K. Card, Thomas P. Moran and Allen Newell, (1983). *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

W. G. Chase and Herbert A. Simon, (1973). Perception in chess. *Cognitive Psychology.* 4:55-81.

William E. Cooper (editor), (1983). *Cognitive Aspects of Skilled Typewriting.* Springer-Verlang, New York.

N. Delvolve and Y. Queinnec, (1983). Operator's activities at CRT terminals: A behavioural approach. *Ergonomics.* 26(4):329-340.

D. E. Egan and B. J. Schwartz, (1979). Chunking in recall of symbolic drawings. *Memory and Cognition.* 7(2):149-158.

G. Field, (1987). Experimentus Interruptus. *SIGCHI Bulletin.* 19(2):42-46.

Barry D. Floyd, (1985). *Mental Organization for a Command Language.* PhD thesis, The University of Michigan.

Donald R. Gentner, (1983). Keystroke timing in transcription typing. In William E. Cooper (editor), *Cognitive Aspects of Skilled Typewriting*, chapter 5. Springer-Verlang, New York.

Judy Haley, Data Entry Training Supervisor. (1986). Interview: May.

B. H. Kantowitz and J. R. Buck, (1983). Feedback and control. In Barry H. Kantowitz and Robert D. Sorkin (editor), *Human Factors: Understanding People-System Relationships*, chapter 12. John Wiley, New York.

H. D. Kimmel, Evert H. van Olst, and J. H. Orlebeke, (1979). *The Orienting Reflex in Humans.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Roberta L. Klatzky, (1980). *Human Memory: Structures and Processes, 2nd ed.* W.H. Freeman, San Francisco.

E. T. Klemmer and G.R. Lockhead, (1962). Productivity and errors in two keying tasks: A field study. *Applied Psychology* 46(6):401-408.

J. Kreifeldt and M. McCarthy, (1981). Interruption as a test of the user-computer interface. In *Proceedings of the 17th Annual Conference on Manual Control*, pp. 655-667. JPL Publication 81-95, Jet Propulsion Laboratory, CIT.

Clayton Lewis and Donald A. Norman, (1986). Designing for error. In Donald A. Norman and Stephen W. Draper (editor), *User Centered System Design: New Perspectives on Human-Computer Interactions.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Gordon D. Logan, (1982). On the ability to inhibit complex movements: A stop-signal study of typewriting. *J. Experimental Psychology: Human Perception and Performance.* 8(6):778-792, December.

Gordon D. Logan, William B. Cowan, and Kenneth A. Davis, (1984). On the ability to inhibit simple and choice reaction time responses: A model and a method. *J. Experimental Psychology: Human Perception and Performance.* 10(2):276-291.

John Long, (1976). Visual feedback and skilled keying: Differential effects of masking the printed copy and the keyboard. *Ergonomics.* 19(1):93-110.

John Long, (1976). Effects of delayed irregular feedback on unskilled and skilled keying performance. *Ergonomics.* E 19(2):183-202.

Kenneth L. Marr, (1988). *Error Correction in Data Entry: A Cognitive Based Model of a Routine, Highly Practiced Computer Mediated Task.* PhD thesis, New York University, Leonard N. Stern School of Business.

G. Minton, (1969). Inspection and correction error in data processing. *Journal of the American Statistical Association.* :1256-1275.

George Morin, Manager of Data Entry, (1986). Interviews. Spring.

Alan S. Neal, (1977). Time intervals between keystrokes, records, and fields in data entry with skilled operators. *Human Factors.* 19(2):163-170.

David M. Neves and John R. Anderson, (1981). Knowledge compilation: Mechanisms for the automatization of cognitive skill. In John Anderson (editor), *Cognitive Skills and Their Acquisition.* Lawrence Erlbaum Associates, Hillsdale, New Jersey.

Donald A. Norman, (1981). Categorization of action slips. *Psychological Review.* 88(1):1-15, January.

Donald A. Norman and David E. Rumelhart, (1983). Studies of typing from the LNR research group. In William E. Cooper (editor), *Cognitive Aspects of Skilled Typewriting*, chapter 3. Springer-Verlang, New York.

Patrick Rabbitt, (1978). Detection of errors by skilled typists. *Ergonomics.* 21(11):945-958.

Arthur S. Reber, (1967). Implicit learning of artificial grammars. *Journal of Verbal Learning and Verbal Behavior.* 6:855-863.

Judith S. Reitman, (1976). Skilled perception in GO: Deducing memory structures from inter-response times. *Cognitive Psychology.* 8():335-356.

Judith S. Reitman and Henry H. Reuter, (1980). Organization revealed by recall order and confirmed by pauses. *Cognitive Psychology.* 12(4):554-581.

Scott P. Robertson and John B. Black, (1983). Planning units in text editing behavior. In *CHI '83 Proceedings*, pages 217-221. New York.

D. E. Rumelhart and D. A. Norman, (1982). Simulating a skilled typist. *Cognitive Science* 6:1-36.

Timothy A. Salthouse, (1984). The skill of typing. *Scientific American.* 250(2), February.

Barr Z. Segal, (1975). *Effects of method of error interruption on student performance at interactive terminals.* Master's thesis, Dept. of Computer Science, University of Illinois at Urbana- Champaign.

L. H. Shaffer, (1987). Timing in the motor programming of typing. *Quarterly Journal of Experimental Psychology.* 30:333-345.

L. H. Shaffer and Jane Hardwick, (1969). Errors and error detection in typing. *Quarterly Journal of Experimental Psychology.* 21:209-213.

Alan T. Welford, (1976). *Skilled Performance: Perceptual and Motor Skills.* Scott, Foresman and Company, Glenview, Ill.,

Leonard J. West and Yizchak Sabban, (1982). Hierarchy of stroking habits at the typewriter. *Applied Psychology.* 67(3):370-376.

B. Zeigarnik, (1927). Das Behalten erledigter und unerledigter Handlungen, *Psychologische Forschung* 9:1-85.