

# **Human Centered Methods in Information Systems: Current Research and Practice**

Steve Clarke  
University of Luton Business School, UK

Brian Lehaney  
University of Luton Business School, UK



**IDEA GROUP PUBLISHING**  
Hershey USA • London UK

Senior Editor: Mehdi Khosrowpour  
Managing Editor: Jan Travers  
Copy Editor: Maria Boyer  
Typesetter: Tamara Gillis  
Cover Design: Connie Peltz  
Printed at: BookCrafters

Published in the United States of America by  
Idea Group Publishing  
1331 E. Chocolate Avenue  
Hershey PA 17033-1117  
Tel: 717-533-8845  
Fax: 717-533-8661  
E-mail: [jtravers@idea-group.com](mailto:jtravers@idea-group.com)  
Website: <http://www.idea-group.com>

and in the United Kingdom by  
Idea Group Publishing  
3 Henrietta Street  
Covent Garden  
London WC2E 8LU  
Tel: 171-240 0856  
Fax: 171-379 0609  
<http://www.eurospan.co.uk>

Copyright © 2000 by Idea Group Publishing. All rights reserved. No part of this book may be reproduced in any form or by any means, electronic or mechanical, including photocopying, without written permission from the publisher.

Library of Congress Cataloging-in-Publication Data

Human centered methods in information systems : current research and practice / [edited by] Steve Clarke, Brian Lehaney  
p. cm.

Includes bibliographical references.

ISBN 1-878289-64-0 (pbk.)

1. Information technology. 2. Information technology--Social aspects. 3. Human engineering. I. Clarke, Steve, 1950- II. Lehaney, Brian, 1953-

T58.5 H58 2000  
025.04--dc21

99-088689

British Cataloguing in Publication Data

A Cataloguing in Publication record for this book is available from the British Library.

## Chapter V

# System and Training Design for End-User Error

Jonathan K. Lazar  
Towson University

Anthony F. Norcio  
University of Maryland Baltimore County

## INTRODUCTION

Errors are a major problem for users. In the distant past, the users of computer technology often were limited to computer professionals with extensive technical training. With the growth of personal computers and the Internet, millions of people without technical backgrounds use computer technology on a daily basis, both at work and for leisure activities.

Because errors can be such a problem for the end user, it is important to examine the causes of error, as well as different approaches for assisting the end user. This chapter presents definitions of error, as well as a taxonomy of user error. There are two general approaches for assisting the end user in responding to errors: system design and training design. Both of these are discussed in-depth in this chapter. The purpose of this chapter is to describe the current situation of end user error and suggest ways to improve the end user experience.

## WHAT IS AN ERROR?

Defining an error can be challenging, since several different definitions of errors have been proposed in the literature. These definitions generally fall into two categories (Arnold and Roe, 1987). One set of error definitions is *user-centered*; the other set of definitions is *system-centered* (Arnold and Roe, 1987).

### User-Centered Definitions of Error

Users want to complete their tasks successfully. User-centered definitions consider errors from the point of view of the user. User-centered definitions of error view an error as when a user's desired action is not carried out (Norman, 1983).

Users are concerned with reaching their goals, and from the users' point of view, errors keep them from reaching those goals. Some of the user-centered definitions of error that have been presented in the literature are:

- when a user's intention or goal is not attained (Arnold and Roe, 1987, p. 204)
- the non-attainment of a goal (Frese and Altmann, 1989).

User-centered definitions do not blame users for errors. User-centered definitions of error only state that errors keep users from reaching their goals. Zapf et al. point out that for an error to be defined as such, a specific program or system must be designed to perform the task that the users want (Zapf et al., 1992). If the user has a specific goal, but the program or system is not designed to perform the tasks to reach such a goal, then this is called a functionality problem (Goodwin, 1987). For instance, if a user attempts to use a statistics program to browse the web, this would be considered a functionality problem, not an error, because the application (the statistics program) was not designed to meet the user's task goal (browsing the web). The applications and systems should be designed to perform the tasks to reach the users' goals.

### **System-Centered Definitions of Error**

System-centered definitions of error view errors from the system's point of view; user goals are not addressed. System-centered definitions of error are more technically-oriented. From the system's point of view, if something cannot process successfully, it is due to an error on the user's part. System-centered definitions of error blame the users (Lewis and Norman, 1986). Some of the system-centered definitions of error in the literature include:

- An action that violates a rule (Frese and Altmann, 1989).
- Something that the system cannot respond to (Lewis and Norman, 1986, p. 411).
- Actions that are inappropriate (Booth, 1991).

### **User Perception Of Error**

Although system-centered definitions of error blame the user for errors, it is not useful to blame a user (Zapf et al., 1992). In human-computer interaction research, the focus is on assisting and designing for the end-users of technology (Dix et al., 1998; Preece et al., 1994; Shneiderman, 1998). We postulate that an error occurs whenever a user **perceives** that an error occurs. If the user perceives that an error occurred, it does not make a difference whether system designers classify it as an error or not. The end user is not concerned with theoretical differences in classifications. Instead, the end user is frustrated because they are not able to reach their task goal. Inexperienced end users frequently tend to blame themselves for making an error (Carroll and Mack, 1984; Lewis and Norman, 1986). Errors intimidate less experienced end users. More experienced end users tend to be confident in their abilities (Carroll, 1990).

## **CLASSIFICATION TAXONOMY OF ERRORS**

The classification system for user errors that is prominent in the world of system design is provided by Donald Norman. At the highest level, Norman separates errors into two types: mistakes and slips (Norman, 1983). Norman defines a mistake as when users choose the wrong commands to reach their goals (Arnold and Roe, 1987; Norman, 1983). A mistake has also been called a conceptual error (Booth, 1991). A slip, on the other hand, is when a user's intended command is correct, but the user makes an error (such as a spelling error) in entering their commands (Norman, 1983). Within slips, there are many different classifications. For instance, Norman defines mode errors, description errors, capture errors, activation errors, and data description errors.

### **Mode Errors**

Frequently, applications and systems have different modes. A keypress while the system is in one mode will provide a different action than the same keypress when the system is in a different mode. A mode error is when users believe that a system is in one mode, when instead, it is in another mode (Norman, 1988). Users then perform their actions with the mistaken belief that the system is in a certain mode. Because the system is in a different mode, their actions may have results other than what the users intended, and the user does not reach their goals.

### **Description Errors**

Many times, different actions or procedures are carried out using a similar set of commands. A description error is when users perform a procedure in a correct manner, but perform it on the wrong file, item or object (Norman, 1983; Norman, 1988). An example of a description error could be to send a business colleague the wrong file as an e-mail attachment. The user performed the procedure in the correct manner, saving a file in the correct directory, uploading it to their e-mail account, and correctly attaching it to their e-mail message. However, these procedures were performed on the wrong file. Description errors frequently occur when objects, files, or items, look similar or are physically close to each other.

### **Capture Errors**

A capture error is when there is overlap between one set of commands and another, and the user performed the wrong sequence of commands (Norman, 1983). When attempting to perform one set of commands, another set of commands, which is similar, takes over (Norman, 1988). There are many commands that are similar. In some of these cases, the commands partially overlap. For instance, a keystroke sequence of control-alt-delete reboots most personal computers. If the key sequence control-alt by itself performed a procedure, it is expected that many times, users would mean to type only control-alt, but instead would type control-alt-delete.

### **Activation Errors**

An activation error occurs when users fail to complete all of the required procedures to reach their intention (Norman, 1983). This may be due to other events that have occurred while users are performing the appropriate actions. These other

events take away the users' attention, causing them to forget the exact procedures to execute and the order in which the procedures should be executed (Norman, 1983). Norman later renamed this specific type of error as a 'loss-of-activation' error (Norman, 1988).

### **Data-Driven Errors**

A data-driven error is based on the arrival of data to our senses (Norman, 1988). Users may be in the process of entering data, when someone tells them that the current ballgame score is 3 to 1, Orioles winning. The users then may enter the numbers 31 instead of the actual data that should be entered. This differs from an activation error, because it does not completely end the user's procedure. The user can continue with the procedure, however, they will have entered incorrect data at an earlier point.

## **FACTOR OF THE NETWORKED ENVIRONMENT**

Norman's taxonomy of error was created before the advent of the networked environment. Before the introduction of local area networks and the widespread adoption of the Internet, the paradigm for most users was the stand-alone personal computer. In the stand-alone environment, the user is in control of all of the computer equipment. If there is either a hardware crash, or the user makes an error, the user has full control over the situation. The only technical components involved are the user's hardware and software. Although the end user might not know the best technique for recovery, the fact remains that the end user *could* control the outcome and implement a recovery solution. In the networked environment, there are a number of factors outside the control of the end user. Even if the end user knew what was wrong, they might not be in a situation to do anything about it.

From the user point of view, the networked environment is quite different from the non-networked environment of stand-alone PCs. The networked environment does not provide the user with the same level of control. A PC might be connected to a local area network, which is in turn connected through an Internet service provider to the Internet. If the user is attempting to access a website, for example, Princeton University, that website also has a set of connections to the Internet. All of these hardware and software components (the local area network, the Internet service provider, the Internet, the Princeton University website) are outside of the user's control. If any one of these components fail, the user has no control over the situation. This is what differentiates the networked environment from the stand-alone PC environment; *the user has a lower level of control*. Because many components are outside of the end user's control, a new type of error can occur. This type of error is called a *situational error* (Lazar and Norcio, 1999b).

## **SITUATIONAL ERROR**

If the user attempts to access the website of Princeton University, most likely, they will type in the correct URL of <http://www.princeton.edu>. If the user does not

type in the URL correctly, they have probably committed one of the errors from Norman's taxonomy, described earlier in this chapter. Assuming that the user has typed in the URL correctly, they expect to see the Princeton University website come up on the screen. However, there are a number of possible events that might keep the user from seeing the Princeton University website. Domain name service might be setup incorrectly. The local network might not be functioning properly. The connection to the Internet service provider might be down. Princeton University might be having some technical problems. Even though the user has typed in the URL correctly, the user may not be able to access the Princeton University website. All of these problems are problems that are outside of the control of the user; *the user cannot do anything to remedy the situation* (Lazar and Norcio, 1999b). What could be even more confusing to the user is that the user had typed in the correct URL previously, and was able to successfully complete their task goal, and access the Princeton University website (Lazar and Norcio, 1999b). If the user has typed in the URL correctly, why are they not able to reach the website that they want? This type of situation calls for a new classification of error. This error is called a *situational error*.

In a situational error, a user's task goal requires the use of a network resource which is not available or is not functioning properly (Lazar and Norcio, 1999b). Even if the user enters all of their commands in the appropriate manner, they are not able to reach their task goal. *A situational error is when a user's task goal requires the use of a network resource, which is not available or is not functioning properly* (Lazar and Norcio, 1999b). A situational error might occur for a number of reasons:

- There might be a problem with the connection from a user's computer to a local area network.
- There might be a problem with the connection between a local network and an Internet Service Provider.
- Domain name service might not be set up correctly, so users on a local network may not be able to access sites on the world wide web.
- A network resource may not be properly configured. A remote site may have failed (Johnson, 1998).

Regardless of why a situational error occurs, the fact remains that the user cannot reach their task goal, even though the user may have typed in all commands correctly.

It is important to compare the situational error to the traditional classifications of errors. A situational error cannot be considered a slip. Since the user does not carry out the commands incorrectly, a situational error is not a slip. Furthermore, since the user has chosen the correct commands to reach their goals, a situational error is not a mistake. In mistakes and slips, it is assumed that the user caused an error. However, there is no classification for a situation where a user selected the correct commands, and entered the commands in the correct manner, but was still not able to attain their goal. Other error classification schema have been presented in the literature, but they, too, do not address these situational errors (Zapf et al., 1992).

How is it possible to assist end users in responding to errors (both traditional

and situational)? There are generally two approaches to assisting end users with errors. These two approaches are *system design* and *training design* (Lazar and Norcio, 1999b).

## SYSTEM DESIGN FOR END-USER ERROR

Computer systems should be designed in a way that they assist the users in responding to errors. Although it might be virtually impossible to eliminate errors altogether, systems can be designed to minimize many user errors (Lewis and Norman, 1986). In his book, *The Psychology of Everyday Things*, Donald Norman discusses three goals for system design related to error:

- Minimize the causes of error
- Make it possible to reverse actions easily and make it hard to perform actions that cannot be reversed
- Make it easy to discover errors when they occur, and make it easy to correct the errors (Norman, 1988).

## MINIMIZING CAUSES OF ERROR

Many times, systems are not designed to minimize error. There are many different approaches to minimizing error. The type of interface mode can affect the probability for end-user error. For instance, if users have only 5 or 6 possible choices to make, a menu might be an appropriate interface choice. By using a menu, the possibilities for user error are limited. If a command language is used, there is an increased chance for the user to make an error (Turban, 1995). With a command language, there are thousands of possible commands for the user to choose from. A command language can be confusing to all but the most experienced user (Turban, 1995).

Regardless of what type of interface mode is chosen, it is important to perform evaluation testing with end users, possibly in the form of prototyping (Booth, 1991; Norman, 1988). By performing user testing, it is possible to learn more about the types of errors that occur (Senders and Moray, 1991). It might be possible to 'engineer-out' some of the errors. By designing a system with consideration of the errors that the user make, it might be possible to limit the number of errors that are made. However, it is not always possible to 'engineer-out' the errors from the system. In cases where it is impossible to eliminate an error-producing behavior, it is important to design the system in a way that errors are easy reversible, without causing large-scale failure (Senders and Moray, 1991).

### Reversible Actions

Since users are very likely to make errors, actions should be easily reversible (Arnold and Roe, 1987; Norman, 1983). This means that there should be an easy way that users can return their system to the last previous good state, before the erroneous action was taken (Arnold and Roe, 1987). Many interfaces provide an 'undo' feature, which can easily reverse the last action performed by the user. Some

actions, such as deleting a file, might not be reversible once performed (Norman, 1983). For these irreversible actions, the system should make it harder for users to perform such actions (Norman, 1983). One way of doing this is to require confirmation of the action, or to require users to type in a special command (Norman, 1983). A good example of this is when a user posts a message to a USENET newsgroup using the PINE mail program. If a user indicates that they want to post a message to a newsgroup, PINE first asks them 'Send Message?' If the user responds positively, PINE responds 'Posted message may go to thousands of readers. Really post?'. Once the user has responded yes to 'Really Post?' the message will automatically be sent to thousands of news servers, and it will be very hard to undo this action.

### **Error Messages**

Before a user can respond to an error, a user must be aware that an error occurred. To maximize the discovery of the error, system status information needs to be available to the user (Patrick, 1987). The sooner a user is aware of an error, the sooner they can correct it (Lazonder and Meij, 1995). If users are not immediately aware of an error, the error can be compounded over time, into a larger error which is harder to recover from (Carroll and Carrithers, 1984; Carroll and Mack, 1984). This is especially true for end-users, who rely on error messages to determine that something could be wrong (DuBoulay and Matthew, 1984). How should an error message attract the attention of the user? Lazonder and Meij emphasize the location of the error message. If an error message is displayed in a dialog box superimposed on the center of the active screen, users will be likely to notice it (Lazonder and Meij, 1995). If an error message is displayed in the lower right hand corner of the screen, and no other signals (graphics, sounds, etc.) draw the user's attention to the error message, the user might not even notice the error message (Lazonder and Meij, 1995). The user cannot respond to an error when they do not even know that one has occurred!

As part of assisting users in recovering from errors, users must be provided with easy-to-understand error messages (Brown, 1983; Shneiderman, 1998). The goal of error messages should be to assist users in recovering from errors (Davis, 1983). If an error message is not clear, users can be confused about what the error is, and therefore, might not be sure how to respond to the error. Shneiderman found that error messages that are easier for the user to understand and interpret can result in users being better able to respond appropriately to errors (Shneiderman, 1982). He suggests that error messages should:

- be specific
- be positive
- tell users what to do to respond to the error (Shneiderman, 1998).

Arnold and Roe go one step further, by encouraging system designers to not just tell the users what to do, but to give users information about the different alternatives that they have to respond to the error (Arnold and Roe, 1987). This can facilitate recovery from the error.

Although guidelines have been presented in the research literature for design-

ing error messages, most of the guidelines are not followed. An example of a confusing error message, as described by DuBoulay and Matthew, is the error message 'fatal error in pass zero' (DuBoulay and Matthew, 1984). In their anecdotal experience with students, novice users have had trouble understanding the meaning of that message (DuBoulay and Matthew, 1984). Current error messages, such as the 'general protection fault' error message, are equally confusing. What does a 'general protection fault' mean to end users? Such a message does not communicate to end users what error actually occurred. Nor does such a message give the end user any information on what the next action should be. From the end user point of view, a 'general protection fault' message has the same meaning as a message saying 'an error occurred.' Such a message does not provide any useful information to the end user. Error messages for networked applications (such as web browsing) are no better. Users are given error messages that say things such as 'server cannot connect' and '404: file not found.' These error messages do not communicate to the user what occurred, nor does it instruct the user on what alternatives they have to respond to the error.

To assist novice users in responding to errors in the networked environment (situational errors), more information needs to be provided to the user. What is Domain Name Service? What is a 404 error? The novice user may not know what these messages mean. Therefore, it may be necessary to provide the novice user with further information, such as a message saying 'There is an error on the network, but it is not due to your actions.' Another approach, such as 'The network is experiencing problems; please try again later' might be appropriate. This is similar to the phone company message 'All circuits are busy. Please try again later.' These error messages let the novice user know that, although they may not be immediately able to reach their task goal, that:

- This error is not their fault
- They should attempt their goal again later that day
- On their next task attempt, they can use the same strategy.

It is especially important not to give the user the 'feeling' that they have done something wrong, if indeed they have not done anything wrong. If the user feels that they have done something wrong, they may attempt a different set of actions to reach their task goal, when their original set of actions was appropriate. This can only increase the problem.

## **TRAINING DESIGN**

Systems should be designed to meet the needs of the end user. However, a well-designed system, without adequate training, is not sufficient. Training the end user is an important factor in the successful use of an information system (Martin et al., 1994; Whitten and Bentley, 1997). However, all training is not alike. Different training methods may present the same material using different approaches. Specifically, different training methods approach errors in a different manner.

### **Traditional (Procedural) Training**

Traditional training methods, also called *procedural training*, typically involve giving users a list of specific steps to follow in order to learn a task (Carroll, 1984; Santhanam and Sein, 1994; Wendel and Frese, 1987). Users are expected to follow the steps exactly (Carroll, 1984). In traditional training methodologies, users are not given any information about the structure of the system (Santhanam and Sein, 1994).

Traditional or procedural methodologies for training novice users in using computer applications tend to focus on avoiding errors (Carroll, 1990; Frese and Altmann, 1989). The assumption of these training methodologies is that users never make errors when performing tasks (Carroll, 1990). This does not realistically model the end-user, since it is virtually impossible to avoid errors when learning new tasks. (Arnold and Roe, 1987; Carroll, 1990; Greif and Keller, 1990; Lazonder and Meij, 1995). In fact, novice users may make errors as soon as they begin their task (Carroll and Mack, 1984). It is more realistic to assume that since all of the features of a computer system can not be taught in training sessions, errors will occur in a workplace situation (Frese and Altmann, 1989).

Regardless of why the errors occur, the fact is that errors do occur frequently with novice users. Typically, novice users make insignificant errors, but in traditional procedural training, they are not instructed on how to recover from these errors (Carroll, 1984; Carroll, 1990; Carroll and Mack, 1984; Lazonder and Meij, 1995). Carroll and Carrithers found that a few insignificant errors may combine to form more significant errors, which frustrate users, who are not able to recover from the error sequence (Carroll and Carrithers, 1984). Due to the errors, users may not be able to reach their task goals, and simply may give up. Because novice users frequently make errors, it is important to train them in appropriate responses to errors. The literature has described three different approaches to training novice users in responding to errors: error management, exploration, and conceptual models (Lazar and Norcio, 1999a).

### **Error Management**

When learning a new task on a computer, it is inevitable that at some point, users make mistakes. When novice users make errors, they become frustrated, and many times, they give up (Frese and Altmann, 1989; Frese et al., 1991). Error management highlights the positive aspects of errors (Frese and Altmann, 1989). For instance, novice users are provided with statements such as 'Make Errors! You can learn from your errors!' and 'I have made an error. Great!' (Frese et al., 1991; Greif and Keller, 1990). The cornerstone of error management is to train users that errors are not bad. Users are taught that errors are good, because errors are opportunities for learning (Frese and Altmann, 1989).

Another aspect of error management is making users aware of potential problem areas in a system, where errors are more likely to occur (Frese and Altmann, 1989; Frese et al., 1991). Users can receive information on what types of errors are likely to occur. Users may then pay extra attention to these areas, which might increase the likelihood that users can avoid making errors in those areas. On the other hand, if users do make errors in these areas, they are paying special attention, so they are more likely to notice the occurrence of an error. By warning

users of these potential error areas, users are both less likely to make an error, and more likely to be able to respond if they do make an error. This is similar to road signs that warn drivers of road conditions ahead, such as: "bridge freezes before road surface"; "road slippery when wet"; or "sharp curves ahead".

There are other positive aspects of making errors. Errors might keep incorrect sequences from becoming automated (Frese and Altmann, 1989). By making errors, users should be able to correct their actions before a procedure becomes habitual. Users might also learn new procedures as a result of making an error (Frese et al., 1991). Also, in the real-world work environment, errors occur on a frequent basis (Frese et al., 1991). In the work environment, there is not always someone (a trainer) there to assist the user. Users should be prepared for how to handle errors when errors occur on the job.

These error management strategies assist the users in viewing errors as a learning experience and becoming less frustrated by errors (Frese and Altmann, 1989). Greif and Keller agree with this assessment, saying that '...it is important to redefine errors as learning situations for which emotional and cognitive coping strategies have to be developed' (Greif and Keller, 1990, p. 242).

### **Exploration**

Exploration has been defined as encountering objects and situations with a certain degree of uncertainty (Greif and Keller, 1990). Exploratory training has also been described as an inductive approach to learning tasks (Davis and Bostrom, 1993). In exploration, users are encouraged to explore their task environment (Dormann and Frese, 1994). Instead of giving users a step-by-step list of how to perform a task, a more general overview of the environment is provided (Dormann and Frese, 1994). Users are instructed in techniques for navigating through their task environment.

In traditional training, to instruct the user in moving from a home directory to the 'www' subdirectory, the user would be told to type `<cd www>`. In exploratory training, the user would receive a description of the `<cd>` command, and how to implement it. The user would not, however, be told exactly what to type. The users are encouraged to attempt the correct command based on their knowledge of the command.

Carroll believes that exploration is a more appropriate methodology for training novice users, because they are not overloaded with too much information (Carroll, 1984). Furthermore, exploration more closely models how novice users naturally tend to approach new tasks (Wendel and Frese, 1987). Greif and Keller state that when interacting with a computer, users usually do not plan their actions in advance. Instead, users follow something akin to trial-and-error (Greif and Keller, 1990). Therefore, Wendel and Frese suggest modeling the training on the users' behavior, and encouraging users to explore (Wendel and Frese, 1987). Payne and Howes also note that exploration might be considered 'more fun,' and less like work, than procedural methods, for novice users (Howes and Payne, 1990; Payne and Howes, 1992).

With the knowledge of the structure of their task environment, users are better able to respond to errors. There are also other advantages to exploration. Through

exploration, users possibly can find a better way to perform the task (Frese and Altmann, 1989; Senders and Moray, 1991). Exploration can benefit users because it has '...the additional effect of eliciting positive emotional feelings and self-evaluations of competence and efficacy' (Greif and Keller, 1990, p.236). Users may feel more confident when they explore, allowing them to respond to errors with more confidence.

### Conceptual Models

Another method that has been presented to assist novice users in responding to errors is conceptual models (Santhanam and Sein, 1994). A conceptual model is an 'accurate, consistent and complete representation of the target system' (Staggers and Norcio, 1993, p. 588). These models are useful when teaching human users about computer systems (Norman, 1987; Staggers and Norcio, 1993). A conceptual model of a computer system could consist of a basic description of the components of a computer system, along with how those components work together (Santhanam and Sein, 1994). The human users, when given a conceptual model, compare it to what is happening in their world (Staggers and Norcio, 1993).

There are two types of conceptual models: *analog* and *abstract* models (Santhanam and Sein, 1994). An analog conceptual model compares the target system (the system that the user is learning about) to another type of system (Santhanam and Sein, 1994). For instance, a computer network could be compared to a system with two cans and a string. An abstract conceptual model describes a system using charts, diagrams, and mathematical expressions (Santhanam and Sein, 1994). Sein et. al. found that abstract conceptual models were more effective than analog conceptual models, in training users (Sein, Bostrom and Olfman, 1987). In a study of subjects learning to use an electronic mail package, subjects who received conceptual models had a higher level of performance than subjects who received procedural training, although the difference was not statistically significant (Santhanam and Sein, 1994). In the commentary on their experiment, Santhanam and Sein note that conceptual models might be helpful in explaining errors to users (Santhanam and Sein, 1994).

### Error Training

A combination of techniques, called 'Error Training,' has been presented in the literature (Dormann and Frese, 1994; Frese et al., 1991; Nordstrom, Wendland and Williams, 1998). Although Frese et. al., Dormann and Frese, and Nordstrom, Wendland, and Williams present error training as one training method, it is actually a combination of two different training approaches — *error management* and *exploration*. There are currently three published studies on the effects of error training. These studies are the Frese, et. al. (1991) study, the Dormann and Frese (1994) study, and the Nordstrom, Wendland, and Williams (1998) study. The 1991 and 1998 studies focused on the task environment of word processing, and the 1994 study focused on the task environment of statistical software. In all three of these studies, subjects who received error training had higher levels of performance than those who received traditional training. In addition, satisfaction levels were measured in the 1991 and 1998 studies, and in both studies, subjects who received

error training were more satisfied with their experiences than subjects who received traditional training.

Error training is the combination of two other training methods - error management and exploration. Although the experimental effects of error management alone are unknown, there are studies published on exploration, and researchers found that users preferred using the exploratory approach (Carroll and Mack, 1984; Carroll and Mazur, 1986). It is important to note that there has also been related work done on user manuals using the exploratory approach (Carroll, 1984; Carroll, 1990; Wendel and Frese, 1987).

### **Training Methods in the Networked Environment**

Since errors occur more frequently in the networked environment, it is important to learn more about how successful the different training methods are in network-based applications (such as web browsing, groupware, and e-mail). We recently completed an experiment on training methods for web browsing. This experiment involved over 250 subjects. Although we are still in the process of completing the data analysis and writing up the results, we can present a preliminary finding here. In the experiment, we found that exploration was the most appropriate training method for training end users in web browsing. Subjects who received exploratory training had the highest level of performance on a set of tasks.

The research on training methods for networked software applications needs to be increased. For example, it would be useful to learn which training methods are most appropriate for other task applications in the networked environment, such as e-mail and groupware. It is possible that in the future, all applications will be 'networked applications.' With each new release of word processing, spreadsheet, and database applications, these applications are becoming more intertwined with the Internet. In the future, it may be that software applications that are stand-alone are a rarity. Indeed, in the future, all software applications might be networked applications.

## **LACK OF ATTENTION TO SYSTEM DESIGN AND TRAINING DESIGN**

Frequently, concerns related to system design and training design for end-user error are not addressed. Many times, systems are designed with the necessary functionality, but system designers ignore the needs of the end user (Goodwin, 1987). However, usability should be a central consideration in system design (Preece, 1990). In some cases, the interface design is a last-minute consideration; an afterthought. This is a mistake, since to the user, the interface IS the system. If a system is poorly designed for the end user, appropriate and sufficient training might help make up for the system.

Unfortunately, training is frequently lacking. In many situations, training is a 'less-visible' expense (United States General Accounting Office, 1998). When training budgets are cut, the effects are not immediately as obvious. It is obvious that you cannot provide an e-mail system without hardware, software, and network wiring. However, the absence of end user training will not be as immedi-

ately obvious. Even when training is provided, many times, it does not address user error (Lazar and Norcio, 1999a). Training, and specifically, training that meets the needs of the end user, is necessary to ensure the successful use of technology by the end user.

Paying attention to errors may cost more in the system development life cycle, as well as increase the costs of end-user training. However, if attention is not paid to errors in the system and training design, the end users will wind up being less productive. Errors decrease productivity because users spend a large amount of time trying to recover from them (Carroll and Carrithers, 1984). The end users will wind up being less productive and less satisfied. It is also possible that the end users will overtax the resources of the information technology support staff (usually at a 'help desk') that is assigned to assist the end users. End user error must be addressed, as it is an important factor in the successful use of an information system.

## CONCLUSION

End-user error is a persistent problem that requires attention. It is important to understand what types of errors can occur. With this understanding, systems can be designed to assist the end user with error. But designing the system is only the beginning. Training can help end users understand why errors occur, and help them learn to respond to errors effectively. With the increasing use of the Internet, new types of errors which are beyond the control of the user can occur. Users need to be made aware of these situational errors, and error messages need to be better designed to assist the user with situational errors.

## REFERENCES

- Arnold, B., and Roe, R. (1987). User errors in human-computer interaction. In M. Frese, E. Ulich, and W. Dzida (Eds.), *Human computer interaction in the workplace* (203-220). Amsterdam: Elsevier Science Publishers.
- Booth, P. (1991). Errors and theory in human-computer interaction. *Acta Psychologica*, 78(1/3), 69-96.
- Brown, P. (1983). Error messages: The neglected area of the man / machine interface. *Communications of the ACM*, 26(4), 246-249.
- Carroll, J. (1984). *Minimalist design for active users*. Proceedings of the Human-Computer Interaction- INTERACT '84, 39-44.
- Carroll, J. (1990). *The nurnberg funnel: Designing minimalist instruction for practical computer skill*. Cambridge, Massachusetts: MIT Press.
- Carroll, J., and Carrithers, C. (1984). Training wheels in a user interface. *Communications of the ACM*, 27(8), 800-806.
- Carroll, J., and Mack, R. (1984). Learning to use a word processor: By doing, by thinking, and by knowing. In J. Thomas, and M. Schneider (Eds.), *Human Factors in Computer Systems* (13-51). Norwood, N.J.: Ablex Publishing.
- Carroll, J., and Mazur, S. (1986). Lisa Learning. *IEEE Computer*, 19(11), 35-49.
- Davis, R. (1983). User error or computer error? Observations on a statistics package.

- International Journal of Man-Machine Studies*, 19(4), 359-376.
- Davis, S., and Bostrom, R. (1993). Training end users: An experimental investigation of the roles of the computer interface and training methods. *MIS Quarterly*, 17(1), 61-85.
- Dix, A., Finlay, J., Abowd, G., and Beale, R. (1998). *Human-Computer Interaction*. (2nd ed.). London: Prentice Hall England.
- Dormann, T., and Frese, M. (1994). Error training: Replication and the function of exploratory behavior. *International Journal of Human-Computer Interaction*, 6(4), 365-372.
- DuBoulay, B., and Matthew, I. (1984). Fatal error in pass zero: How not to confuse novices. *Behaviour and Information Technology*, 3(2), 109-118.
- Frese, M., and Altmann, A. (1989). The treatment of errors in learning and training. In L. Bainbridge, and S. Quintanilla (Eds.), *Developing skills with information technology* (65-86). Chichester, England: John Wiley & Sons.
- Frese, M., Brodbeck, F., Heinbokel, T., Mooser, C., Schleiffenbaum, E., and Thiemann, P. (1991). Errors in training computer skills: On the positive function of errors. *Human-Computer Interaction*, 6(1), 77-93.
- Goodwin, N. (1987). Functionality and usability. *Communications of the ACM*, 30(3), 229-233.
- Greif, S., and Keller, H. (1990). Innovation and the design of work and learning environments: The concept of exploration in human-computer interaction. In M. West, and J. Farr (Eds.), *Innovation and creativity at work: Psychological and organizational strategies* (231-249). Chichester, England: John Wiley & Sons.
- Howes, A., and Payne, S. (1990). *Supporting exploratory learning*. Proceedings of the Human-Computer Interaction - INTERACT '90, 881-885.
- Johnson, C. (1998). Electronic gridlock, information saturation, and the unpredictability of information retrieval over the world wide web. In P. Palanque, and F. Paterno (Eds.), *Formal Methods in Human-Computer Interaction* (261-282). London: Springer.
- Lazar, J., and Norcio, A. (1999a). *A Framework for Training Novice Users in Appropriate Responses to Errors*. Proceedings of the International Association for Computer Information Systems 1999 Conference, in press.
- Lazar, J., and Norcio, A. (1999b). *To Err Or Not To Err, That Is The Question: Novice User Perception of Errors While Surfing The Web*. Proceedings of the Information Resource Management Association 1999 International Conference, 321-325.
- Lazonder, A., and Meij, H. (1995). Error-information in tutorial documentation: Supporting users' errors to facilitate initial skill learning. *International Journal of Human-Computer Studies*, 42(2), 185-206.
- Lewis, C., and Norman, D. (1986). Designing for error. In D. Norman, and S. Draper (Eds.), *User-centered system design* (411-432). Hillsdale, NJ: Lawrence Erlbaum Associates.
- Martin, E., DeHayes, D., Hoffer, J., and Perkins, W. (1994). *Managing information technology: What managers need to know*. New York: Macmillan Publishing Company.
- Nordstrom, C., Wendland, D., and Williams, K. (1998). To err is human: An examination of the effectiveness of error management training. *Journal of Busi-*

- ness and Psychology, 12(3), 269-282.
- Norman, D. (1983). Design rules based on analyses of human error. *Communications of the ACM*, 26(4), 254-258.
- Norman, D. (1987). Some observations on mental models. In R. Baecker, and W. Buxton (Eds.), *Readings in human-computer interaction: A multidisciplinary approach* (241-244). San Mateo, California: Morgan Kaufmann Publishers.
- Norman, D. (1988). *The psychology of everyday things*: Harper Collins Publishers.
- Patrick, J. (1987). Information at the human-machine interface. In J. Rasmussen, K. Duncan, and J. Leplat (Eds.), *New technology and human error* (341-345). Chichester, England: John Wiley & Sons.
- Payne, S., and Howes, A. (1992). A task-action trace for explanatory learners. *Behaviour and Information Technology*, 11(2), 63-70.
- Preece, J. (1990). *A Guide to Usability*. Milton Keynes, England: The Open University.
- Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., and Carey, T. (1994). *Human-Computer Interaction*. Wokingham, England: Addison Wesley Publishing.
- Santhanam, R., and Sein, M. (1994). Improving end-user proficiency: Effects of conceptual training and nature of interaction. *Information Systems Research*, 5(4), 378-399.
- Sein, M., Bostrom, R., and Olfman, L. (1987). *Conceptual models in training novice users*. Proceedings of the Human-Computer Interaction- INTERACT '87, 861-867.
- Senders, J., and Moray, N. (1991). *Human Error: Cause, Prediction, and Reduction*. Hillsdale, N. J.: Lawrence Erlbaum Associates.
- Shneiderman, B. (1982). System message design: Guidelines and experimental results. In A. Badre, and B. Shneiderman (Eds.), *Directions in Human/Computer Interaction* (55-78). Norwood, N.J.: Ablex Publishing.
- Shneiderman, B. (1998). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. (3rd ed.). Reading, Massachusetts: Addison-Wesley.
- Staggers, N., and Norcio, A. (1993). Mental models: Concepts for human-computer interaction research. *International Journal of Man-Machine Studies*, 38(4), 587-605.
- Turban, E. (1995). *Decision Support and Expert Systems: Management Support Systems*. Englewood Cliffs, NJ: Prentice-Hall.
- United States General Accounting Office. (1998). *School technology: Five school districts' experiences in funding technology programs* (GAO/HEHS-98-35). Washington, D.C.: United States General Accounting Office.
- Wendel, R., and Frese, M. (1987). *Developing exploratory strategies in training: The general approach and a specific example for manual use*. Proceedings of the Human-Computer Interaction- INTERACT '87, 943-948.
- Whitten, I., and Bentley, L. (1997). *Systems Analysis and Design Methods*. Boston: Irwin McGraw-Hill.
- Zapf, D., Brodbeck, F., Frese, M., Peters, H., and Prumper, J. (1992). Errors in working with office computers: A first validation of a taxonomy for observed errors in a field setting. *International Journal of Human-Computer Interaction*, 4(4), 311-339.