Appendix C — Transactions for the Pascal Statement
while J < 10 do J := J + 2;

1. (FIND, Number, Memory Location)
   Obtain the value that is stored in memory space J.
2. (FIND, Number, Program)
   Obtain the number 10.
3. (DECIDE, Numbers, Memory Location and Program)
   Decide if the value obtained in Step 1 is less than the value obtained in Step 2.
4. (MOVE, Line Pointer, Program)
   If the result of Step 3 is false, then go on to the next statement.
5. (FIND, Number, Memory Location)
   If the result of Step 3 is true, obtain the value that is stored in memory space J.
6. (FIND, Number, Program)
   Obtain the number 2.
7. (COMBINE, Numbers, Memory Location and Program)
   Add the values obtained in Steps 5 and 6.
8. (DESTROY, Number, Memory Location)
   Erase the current value in memory space J.
9. (CREATE, Number, Memory Location)
   Store the result of Step 7 in memory space J.
10. (MOVE, Line Pointer, Program)
    Go back to Step 1.

Appendix D — Transactions for the Pascal Statement
A[I] := A[I] + 1;

1. (FIND, Number, Memory Location)
   Obtain the value that is stored in memory space I.
2. (FIND, Number, Memory Location)
   Locate the value of array A indexed by the value obtained in Step 1.
3. (FIND, Number, Program)
   Obtain the number 1.
4. (COMBINE, Numbers, Memory Location and Program)
   Add the values obtained in Steps 2 and 3 together.
5. (FIND, Number, Memory Location)
   Obtain the value that is stored in memory space I.
6. (DESTROY, Number, Memory Location)
   Erase the value in array A indexed by the value obtained in Step 5.
7. (CREATE, Number, Memory Location)
   Store the result of Step 4 in array A indexed by the value obtained in Step 5.

Appendix E — Transactions for the Pascal Statement
type R = record I: integer; Y: real end;

1. (DEFINE, Property, Memory Location)
   Define a class of objects with the property of having an integer location labeled I and a real location labeled Y.
2. (LABEL, Name, Memory Location)
   Label this class of objects as R.

Appendix F — Transactions for the Pascal Statement
type C = (F, G, P);

1. (DEFINE, Property, Memory Location)
   Define a class of objects named F, G, P, having the property that F is the predecessor of G, and G is the predecessor of P.
2. (LABEL, Name, Memory Location)
   Label this class of objects as C.

ICONER: A TOOL FOR EVALUATING ICONS

HENDRIKA ALICE EISEN

Icons are becoming a popular component in the design of user interfaces . Finding icons which are most meaningful and clear to end-users poses a challenge to designers. To help with this challenge, "Iconer" was created as a tool that interface designers can use to quickly test icons with end-users.

Iconer is a HyperCard program that runs on a Macintosh. It is used in two modes. First it is a tool with which designers can easily create icons and set up a user test to evaluate them. Second, it is a testing program which is used to present the user test and automatically collect and summarize data. In this sense it is a type of electronic questionnaire in which two types of matching exercises are presented: first, a multiple choice of icons where people read a task definition and select the the icon that best matches the task (see Figure 1), and second, a multiple choice of tasks where people look at an icon and select the task that matches that icon (Figure 2). Information is gathered about the recognizability of icons, appropriateness of icons and user preferences for icons. All of these sources of information, in combination, can indicate where confusions between icons occur and where task-icon matches are inappropriate.

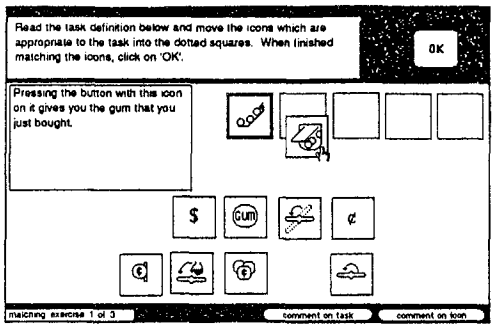

Figure 1. Multiple Choice of Icons. In this screen a user test is in progress -- the user is dragging icons appropriate to the task to the dotted squares.

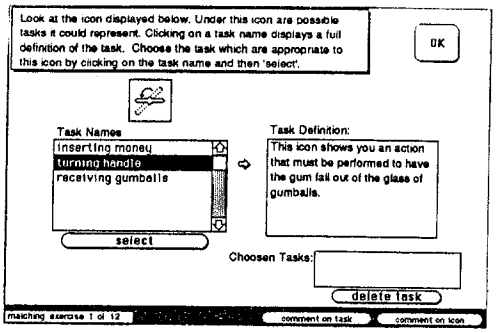

Figure 2. Multiple Choice of Tasks. The user is focusing on the displayed icon and matches the tasks he believes the icon will perform.

Issues Related to Icon Design and Testing

When designing icons, care must be taken to ensure that the meaning of icons is not ambiguous, that the representation used is appropriate, that icons appear in the appropriate context and that the quality of the icon drawing is adequate.

When people use a product, they build a mental model about how it works. Icons play a role in how this model develops. When interpreting icons, different cognitive processes may be used. For example, with a depictive or pictographic icon that is a drawing of an actual object or action, the icon's meaning should be intuitive or obvious based on the user's preexisting knowledge. In the case of an abstract icon that is a symbol representing an object or action, the icon's meaning should be learned quickly and easily recalled. To thoroughly test an icon, the roles of recognition and learning should both be measured.

Iconer focuses on how well users can comprehend the meaning of icons, measuring the discriminability and confusability of icons. Using Iconer once with subjects provides information about the intuitive nature of icons. Repeated testing sessions with the same subjects provides details about the performance of icons that may be related to learning retention.

Iconer should not be thought of as a complete test for icons; it facilitates the first two steps in a three step process. First icons are generated, often with input from users. Second, it is necessary to determine how well a user can comprehend the meaning of icons. Third, icons should be tested in context, on the actual product, in the environment in which the product is used.

Designing and Testing of Iconer's User Interface

Iconer was based on research with pictograms and symbols(Barnard & Marcel, 1984; Easterby & Zwaga, 1984; Green & Pew; 1978; Truijens, 1985). It was created with input from user interface designers from industrial and academic settings. These people were involved in two phases of the development of Iconer. In the first phase, designers were asked how they would use a tool of this type and what kind of features they thought the tool should include. The tool was then designed to meet these requirements. In the second phase, designers participated in a structured user test of the tool.

During the first phase, the major requirement specified by designers was that the ability to discriminate the meaning of an icon should be measured from two perspectives. First, given a description of a task to perform, the subject selects, from a set of potiental icons, the icon or icons that best represent that task. Second, when shown an icon, the subject selects the task, from a set of tasks, that is most closely related to the icon. Other requirements specified for Iconer included the capability to prioritize icons by measuring user preferences and automatic tracking of subject performance. The tool should also include resources to easily create and modify icons

In the second phase, designers participated in two structured user tests of Iconer. In the first test, Iconer was used to perform the matching exercises the actual end-users would perform. In the second test, the tool was used to create the testing materials for a user test of icons.

No major problems emerged using Iconer to perform the matching exercises. Problems did emerge when designers used the tool to create testing materials. The ratings in Table 1 indicate the problem areas. These responses were recorded after the first exposure to Iconer. These problems disappeared after repeated use of Iconer.

Table 1. Rating Iconer's Ease of Use. These numbers represent the frequency of ratings of tasks from the five people run through the user test.

Task	Frequency of Ratings							
	not easy						very easy	
	1	2	3	4	5	6	7	Median
Entering a Task Description	0	0	0	1	1	0	3	7
Copying Icon - Current Stack	2	0	1	0	1	0	1	3*
Copying Icon - Another File	0	0	3	0	1	1	0	3*
Drawing a new Icon	0	0	0	1	1	3	0	6
Deleting an Icon	0	0	0	0	1	0	4	7
Make Test Card & Copying Icons	0	0	0	0	2	1	2	6
Make Test Card not Copying Icons	0	0	0	1	2	2	0	5

* Items failed to meet usability goals.

The major problem, copying icons, was caused by a menu that was nested and used inappropriate terminology. This menu was changed to remove nesting, and names of the menu items were changed to terms that were more familiar to the users.

Summary:

Incorporating icons in the design of a product can improve the product's use if those icons are easily understood by users. This paper describes a tool that was created to help user interface designers test possible icons with end-users to determine the quality of the icons to particular tasks.

References

Barnard, P. and Marcel, T. (1984) Representing and understanding in the user of symbols and pictograms. In Easterby, R. and Zwaga, H. ***Information Design.*** New York: John Wiley and Sons Ltd.

Dreyfuss, H. (1972) ***Symbol Sourcebook.*** McGraw-Hill Book Company: New York.

Easterby, R. and Zwaga, H. (1984) ***Information Design.*** New York: John Wiley and Sons Ltd.

Green, P. and Pew, R. W. (1978) Evaluating pictographic symbols: An automotive application. ***Human Factors,*** **20**(1), 103-114.

Kolers, P. A. (1969) Some formal characteristics of pictograms. ***American Scientist,*** **57**, 348-363.

Truijens, C. L. (1985) Symbols for supplementary telephone services: Experiments within CCITT. ***Eleventh International Symposium on Human Factors in Telecommunications,*** Sevigne, France.

Hendrika Alice Eisen

Alice Eisen in a graduate student in the Psychology Department at Carleton University. Her interests include object-oriented programming, and human factors. Alice's address is: Department of Psychology, Carleton University, Ottawa, Ontario, Canada, K1S 5B6.

COMPUTERIZED PERFORMANCE MONITORING AND PERFORMANCE APPRAISAL

DEBORAH B. FENNER
F. JAVIER LERCH
CAROL T. KULIK

Industrial and governmental employers are increasingly using computerized performance monitoring (CPM) as a tool for evaluating employees' performance. It is estimated that six to seven million employees are currently being monitored by computers and that the number of firms considering such systems is growing (OTA, 1987).

Expectations about the value of CPM systems have been heightened because they permit the collection of vast amounts of objective data about employee performance that cannot be obtained by using traditional employee monitoring methods. CPM systems can provide a supervisor with access to an employee's work performance at any time (real time access) without the employee's knowledge, provide information about an employee's performance at minute levels of detail such as the number of keystrokes typed per minute, provide continuous information about an employee's performance over time, and enable a supervisor to quickly compare an employee's current performance with that of other employees.

Few studies have examined the effects of CPM systems on the workplace and only a subset of these have attempted to directly examine the effects of the systems on the actual users, the supervisors. However, based on the information provided by these studies it is possible to identify some potentially important issues in supervisors' interactions with CPM systems.

Both supervisors and electronically monitored employees believe that CPM systems help to make supervisory evaluations more accurate and fair by providing more objective data about performance (Eisenman,1986; Grant & Higgins, 1989; Irving et al., 1986; Westin, 1986). The CPM literature provides no hard evidence to support this perception. On the other hand, research on the performance appraisal process (DeNisi et al., 1984; Williams, DeNisi, Blencoe, & Cafferty, 1985) suggests that access to CPM data may increase the influence of recorded performance information on employee evaluations by facilitating the supervisor's search for relevant data. In addition, CPM systems may foster the influence of objective performance by functioning as electronic record keepers. Because faulty memories of employee work behavior can influence performance ratings, several performance appraisal theorists have suggested that supervisors keep diaries of employee performance over time which could be consulted when an appraisal is required (Bernardin & Walter, 1977; DeNisi et al., 1984; Feldman, 1981). CPM can be thought of as an electronic diary-keeper that records and reports work performance data. This record keeping function could therefore act in conjunction with CPM's facilitation of the supervisor's information search to support evaluations that are based primarily on objective information.

Performance appraisal researchers have also demonstrated that some types of bias which lead to inaccurate appraisals are difficult to eliminate. For example, several studies show that knowledge of past performance interferes with an accurate assessment of current performance (Huber et al., 1987; Murphy et al.,

---

[1]Full report to be published as "The Impact of Performance Expectations and Computerized Performance Monitoring System Data on Supervisory Performance Evaluation" in the *Journal of Applied Social Psychology*