Available online at www.sciencedirect.com
ScienceDirect
Computers in Human Behavior 23 (2007) 333–352
Computers in Human Behavior
www.elsevier.com/locate/comphumbeh

Learning performance and computer software: an exploration of knowledge transfer

Robin H. Kay

University of Ontario Institute of Technology, Faculty of Education, 2000 Simcoe St. North, Oshawa, Ontario, Canada L1R 7L7

Available online 18 November 2004

Abstract

Computer studies educators have a challenging task in keeping pace with the rapidly changing content of computer software. One way to meet this challenge is to examine the nature of knowledge transfer. Instead of focusing on unique software packages, teachers could concentrate on knowledge that is likely to transfer from one software application to another. The purpose of the current study was to describe what kind of knowledge is used in learning new software, assess the relative effectiveness of this knowledge in aiding the learning process, and examine how the results could advance educational learning theory and practice. Thirty-six adults (18 male, 18 female), representing three computer ability levels (beginner, intermediate, and advanced), volunteered to think out loud while they learned the rudimentary steps (moving the cursor, using a menu, entering data) required to use a spreadsheet software package (Lotus 1-2-3). Previous understanding of terminology, software concepts and actions, and other software packages had the largest impact, both positive and negative, on learning. A basic understanding of the keyboard and common movement keys was also important, although higher level knowledge (e.g., terms, concepts, actions) is probably necessary for significant gains in learning performance. Computer ability had little impact on the type of transfer knowledge used, except with respect to the use of software concepts and, to a lesser extent, terminology. The interaction between problem type and effectiveness of a specific transfer area suggests that identifying specific common tasks among software packages is important in

E-mail address: robin.kay@uoit.ca.

0747-5632/$ - see front matter
doi:10.1016/j.chb.2004.10.029

detecting useful transfer knowledge. It is equally important that computer users understand labeling idiosyncrasies of these common tasks.

Keywords: Learning; Transfer knowledge; Computer; Software

1. Introduction

The phenomenal pace at which software has developed over the past two decades has made it difficult for educators to determine what they should teach. The variety of software designs available have produced inconsistent command languages, confusing operation sequences, chaotic display formats, incomplete instructions, and complex error recovery procedures that teachers are required to deal with on a daily basis (Shneiderman, 1989). Instructors may feel that time spent teaching a particular software package on a particular computer may be wasted given that new software, upgrades, or technology are just around the corner. It is estimated that major software changes occur every 6–18 months (Bellis, 2004; Franzke & Rieman, 1993).

One approach to dealing with complexity of software and the rate of change is to examine the nature of knowledge transfer. Transfer is defined by Bransford, Brown, and Cocking (1999, p. 15) as "the ability to extend what has been learned in one context to new contexts". Identifying and teaching units of knowledge that readily transfer to new computer applications could help students adapt to a constantly changing software environment. Past research, though, suggests that it is difficult to define appropriate and effective cognitive units of transfer (Sternberg, 1989). By effective cognitive unit, I mean the abstract level at which information learned is likely to carry over from one domain to another.

To date, few studies have looked at transfer and computer software (Stine & Wildemuth, 1990). Furthermore, Yamnill and McLean (2001) suggest little is known about what constitutes proper transfer design in training.

1.1. General knowledge

Intuitively, it might seem that general problem solving strategies would be the most efficient transfer knowledge on which to focus. In theory, these general strategies should allow students to adapt to and eventually learn new software content, unhampered by specific, idiosyncratic software commands of previous applications. This general skills strategy or context-independent approach is deeply rooted in other areas of education (Resnick, 1989), but researchers have produced overwhelming evidence to suggest that general problem solving skills do not readily transfer to different contexts (e.g., Anderson, Reder, & Simon, 1996; Carroll, 1990; Resnick, 1989; Singley & Anderson, 1989; Voss, 1989). In fact, some have argued that teaching decontextualized knowledge or general skills can be detrimental to learning (Resnick, 1989).

The ineffectiveness of general problem solving strategies is supported by research on Logo programming (e.g., Lehrer, Lee, & Jeong, 1999; Lehrer & Littlefield, 1993),

but has not been tested with respect to computer software. Some authors speculate that general skills such as mathematics, programming, and typing are necessary precursors for learning to use computer software (Turkle, 1984; Underwood & Underwood, 1990), but this specualtion has yet to be examined formally.

1.2. Specific knowledge

According to Anderson et al. (1996), much of what is learned is situation specific and not readily transferred to new and different scenarios. In other words, the similarity between a new learning task and the original learning task is directly related to the amount of transfer that occurs (Bransford et al., 1999; Reder & Klatzky, 1994; Stine & Wildemuth, 1990; Thorndike, 1913). This conclusion is supported by research in Logo programming (e.g., Lehrer et al., 1999; Lehrer & Littlefield, 1993), but has yet to be tested in the computer software domain.

1.3. Terminology

A number of researchers believe that use of terminology can strongly influence the learning of new computer applications (Carroll & Mack, 1984; Norman, 1986; Owen, 1986). Norman (1986) notes that many terminology problems can ensue when acquiring new software skills. ''Problems arise when different systems are involved, oftentimes with similar functions that have different names and conventions, and with similar names that have different meanings'' (Norman, 1986, p. 59). While observed in some detail in open-ended studies (Carroll & Mack, 1984; Norman, 1986; Owen, 1986), the use of terminology in learning computer software has not been examined systematically.

1.4. Learning context

There are at least two schools of thought with respect to establishing an ideal learning context for transfer of knowledge. One perspective is that the amount of transfer is partially dependent on where attention is directed during learning (Anderson et al., 1996). Keller (1990), in a comprehensive review of the literature on Logo programming and transfer, suggests that transfer is enhanced by making learning explicit and helping students to reflect more on their work. Typically, this kind of guided instruction occurs in a more formal learning environment (e.g., Keller, 1990; Lehrer et al., 1999; Lehrer & Littlefield, 1993).

Other researchers note that transfer can occur naturally in a more informal learning environment (Mayer, Quilici, & Moreno, 1999). Rieman (1996) observed significant learning of computer related concepts where subjects primarily used exploratory learning strategies to solve specific computer related work problems. Some authors (Anderson et al., 1996; Halpin, 1999; Rieman, 1996) have argued that transfer of computer skills are optimized if learning takes place in more complex, real-world environments.

The model followed by the vast majority of researchers has been to compare the effectiveness of specific strategies in a formal, guided learning environment (e.g.,

Keller, 1990). The role of transfer knowledge in learning a computer software package in an informal learning environment has not been explored to date. Rieman (1996) notes that an informal learning environment may be far more reflective of the real-world computer software user who has to cope with a wide range of specific tasks.

1.5. Computer ability and transfer

Anderson et al. (1996), in a general review of the literature, noted that representation and degree of practice were major determinants of transfer from one task to another. This conclusion is supported by Shayo and Olfman (1998) who observed that previous ability had an effect on perceived transfer by users who were learning database software. In addition, Simpson and Pelligrino (1993) found that inexperienced and experienced users responded differently with respect to transfer knowledge while learning an operating system. Finally, Shih and Alessi (1993) reported that a positive relationship between quality of mental models and transfer of knowledge existed when subjects were learning a BASIC programming language. It is possible, then, that advanced computer users will use transfer knowledge differently than their less able counterparts when learning a new software package.

1.6. Current study

In this study, four domains of transfer knowledge will be investigated: *general knowledge* (e.g., mathematics, personal experience, general rules of thumb), *hardware knowledge* (e.g., knowledge of the key locations, computer memory), *software knowledge* (e.g., knowledge of how to use other software packages), and terminology (e.g., understanding computer terms). Subjects will be asked to learn specific spreadsheet software tasks in a relatively informal, exploratory learning environment. In addition, three computer ability levels (beginner, intermediate, and advanced) will be examined.

The purpose of this study is to:

(a) describe what kind of transfer knowledge is related to learning performance in a new software;
(b) assess the effectiveness of this knowledge in aiding learning performance;
(c) examine how the results can advance theory; and
(d) provide recommendations for educators of computer studies.

2. Method

2.1. Sample

The sample consisted of 36 adults (18 male, 18 female): 12 beginners, 12 intermediates, and 12 advanced users, ranging in age from 23 to 49 ($M = 33.0$ years), living in the greater metropolitan Toronto area. Subjects were selected on the basis of

convenience. Equal numbers of males and females participated in each ability group. Sixteen of the subjects had obtained their Bachelor's degree, eighteen their Master's degree, one a Doctoral degree, and one, a community college diploma. Sixty-four percent (n = 26) of the sample were professionals; the remaining 36% were students ($n = 13$). Seventy-two percent ($n = 26$) of the subjects said they were regular users of computers. All the subjects voluntarily participated in the study.

2.2. Computer ability

Three computer ability levels were compared in this study: beginners, intermediates, and advanced users. The criteria used to determined these levels included years of experience, previous collaboration, previous learning, software experience, number of application software packages used, number of programming languages/operating systems known, and application software and programming languages known. A multivariate analysis showed that beginners had significantly lower scores that intermediate and advanced users ($p < 0.005$), and intermediates users had significantly lower scores than advanced users on all eight measures ($p < 0.005$).

2.3. Procedure

2.3.1. Overview

Table 1 provides an overview of the procedure used in this study. Each subject was given an ethical review form, computerized survey and interview before attempting the main task of learning the spreadsheet software package. Note that the survey and interview data were used to determine computer ability level. Once instructed on how to proceed, the subject was asked to think-aloud while learning the spreadsheet software for a period of 55 min. All activities were videotaped with the camera focused on the screen. Following the main task, a post-task interview was conducted. A debriefing session concluded the study (see Table 1).

2.3.2. Learning tasks

The learning tasks used in this study were developed from a popular learning manual on spreadsheet software. Spreadsheet software is used to create, manipulate

Table 1
Overview of procedure

Step	Time (min)
Ethical review form	5
Pre-task survey (computer administered)	20
Pre-task interview (tape-recorded)	5
Instructions	5
Learning spreadsheet software (videotaped)	55
Post-task interview (tape-recorded)	5
Debriefing interview	5
Total	100

and present rows and columns of data. None of the subjects had ever used the specific spreadsheet software package used in this study (Lotus 1-2-3). Tasks were ordered in ascending level of difficulty as assessed by the manual. Subjects completed a maximum of 5 tasks including (1) moving around the spreadsheet (screen), (2) using the command menu, (3) entering data, (4) deleting, copying, and moving data, and (5) editing.

In the 55-min time period allotted to learn the software, a majority of the subjects completed all learning tasks with respect to moving around the screen (100%) and using the command menu (78%). About two thirds of the subjects attempted to enter data (69%), although only one third finished (33%) all the activities in this area. Less than 15% of all subjects completed the final tasks: deleting, copying, moving, and editing data.

2.4. Data collection and analysis

2.4.1. Think-aloud protocols (TAPs)

The main focus of this study was to examine the use of transfer knowledge with respect to learning computer software. The use of think-aloud protocols (TAPs), where subjects verbalize what comes to their mind as they are doing a task, is one promising technique for examining transfer. Essentially, the think-aloud procedure offers a window into the internal talk of a subject while he/she is learning. Ericsson and Simon (1980), in a detailed critique of TAPs, conclude that "verbal reports, elicited with care and interpreted with full understanding of the circumstances under which they were obtained, are a valuable and thoroughly reliable source of information about cognitive processes" (pp. 247).

The analyses used in this study are based on think-aloud data. More specifically, *transfer behaviors*, defined as "previous knowledge used by a subject that influenced learning or the completion of an assigned task", were rated according to the degree to which they influenced the learning.

2.4.2. Presentation of TAPs

The following steps were carried out in the think-aloud procedure to ensure high quality data:

Step 1
(Instructions). Subjects were asked to say everything they were thinking while working on the software. Subjects were told not to plan what they were going to say.
Step 2
(Examples). Examples of thinking aloud were given, but no practice sessions were done.
Step 3
(Prompt). Subjects were told it was important that they keep talking and that if they were silent for more than 5 s, they would be reminded to "Keep talking".

Step 4
(Reading). Subjects were permitted to read silently, but they had to indicate what they were reading and summarize when they had finished.
Step 5
(Giving help). If a subject was really stuck, he/she could ask for help. A minor, medium, or major form of help would be given, depending on how "stuck" a subject was;
Step 6
(Recording of TAPs). Both thinking aloud and the computer screen were recorded using an 8 mm video camera.

2.4.3. Analysis of TAPs

A comprehensive analysis of the written transcripts for all subjects revealed 1342 *transfer behaviors*. Each behavior was assigned a transfer category (terminology, software, hardware, general knowledge) and specific transfer area within that category. The five principal categories and their respective sub-areas are presented in Table 2.

The effect of each transfer area on learning was evaluated using three variables: *frequency of learning behavior*, *mean influence of learning behaviour* (a score from −3 to +3 see Table 3 for rating criteria), and *percentage of subjects* who displayed the learning behaviour. Conceptually, these three variables assessed intensity (mean

Table 2
Transfer knowledge classification criteria

Transfer sub-categories	Criteria
General knowledge	
General	Non-computer knowledge is used (e.g., mathematics, typing)
Metacognitive	Strategic knowledge is used e.g., "I always try and Save to get an idea of how the software works"
Previous personal experience	Refers to personal experience (e.g., "I always had a lot of trouble with columns, so I am very apprehensive about trying to change them")
Hardware	
Hardware – keyboard	Knowledge of the keyboard is used (e.g., where certain keys are)
Hardware – miscellaneous	Refers to knowledge based on hardware not used or visible in the study (e.g., the mouse, memory size)
Software	
Software – other packages	A specific software package is mentioned/used as a reference point, but not at the level of a specific concept, action, or keystroke
Software – concepts	A concept is a static piece of knowledge or a fact (e.g., understanding what a column or a row is)
Software – actions	Refers to a specific procedure (e.g., how to move, copy, edit)
Software – keystrokes	Refers to specific keystrokes or combinations of keys used in other software packages
Terminology	A previously learned word or phrase is referred to

Table 3
Rating system for influence score

Score	Criteria used	Example
−3	A significant misunderstanding or mistake is evident that is judged to use a significant amount of time	Subject thinks that the on-line help is the main menu and spends 15 min learning to do the wrong task
−2	A significant misunderstanding or mistake which leads subject away from solving the task at hand	Subject believes all commands are on the screen and does not understand that there are sub-menus. This results some time loss and confusion
−1	Minor misconception that has little effect on the direct learning of the task at hand	Subject tries HOME key, which takes him back in the wrong direction, but does not cause a big problem in terms of moving to the specified cell
0	(a) Activity has no apparent effect on progress OR (b) Cannot directly determine effect of activity OR (c) Both good and bad effects	(a) Subject tries a key and it does not work (e.g., gets beeping sound) (b) Subject gets upset, but it is hard to know how it affects future actions (c) Subject moves to cell quickly, but fails to learn a better method. It is good that he completed the task, but bad that he did not learn a more efficient method
1	Minor task or piece of knowledge learned which is not significantly useful in learning the software	Subject learns to use the cursor keys, which are moderately helpful in terms of moving around the screen
2	Learns significant piece of knowledge which is useful in terms of learning how to use the software	Subject learns the function of the ENTER key when pressed in the command menu. This is an important key
3	A significantly useful piece of knowledge is learned which saves the subject considerable time	Subject finds page in the manual which clearly lists all the movement keys. Without this page, most subjects take at least 10 min to learn only a subset of the movement keys

influence of leaning behaviour) and prevalence (how often the behavior was observed and by how many subjects).

In addition, a composite score (*total effect score*) was calculated by multiplying the above three variables together. For example, previous understanding of a keyboard was observed 155 times, had mean influence of 0.66, and was used by 97% of the subjects. The *total effect score*, then, was 99.2 ($155 \times 0.66 \times 0.97$).

Finally, *learning performance* was calculated by adding up the number of subgoal scores that each subject attained during the 55-min time period. For each task, a set of learning subgoals was rated according to difficulty and usefulness. For example, the task of "moving around the screen" had five possible subgoals that could be attained by a subject: using the cursor key (1 point), using the page keys (1 point), using the tab keys (1 point), using the GOTO key (2 points), and using the End-Home keys (2 points). If a subject met each subgoal successfully, a score of 7 would be given. If a subject missed the last subgoal, a score of 5 would be assigned.

2.4.4. Reliability of TAPs

Reliability and validity assessments were derived from the feedback given during the study and a post-task interview. One principle concern was whether the TAPs influenced learning. While, several subjects reported that the think-aloud procedure was "weird", "frustrating" or "difficult to do", the vast majority found the process relatively unobtrusive. Almost 70 of the subjects ($n = 25$) felt that thinking aloud had little or no effect on their learning.

The accurate rating of the influence of transfer behaviour on learning is critical to the reliability and validity of this study. Because of the importance of the influence scores, six outside raters were used to assess a 10%, stratified, random sample of the total 1342 transfer behaviors identified. Inter-rater agreement was calculated using Cohen's κ (Cohen, 1960), a more conservative and robust measure of inter rater agreement (Bakeman, 2000; Dewey, 1983). by the total number of behaviors rated. The κ coefficients for inter rater agreement between the experimenter and six external raters (within one point) were as follows: *Rater 1*: 0.80, *Rater 2*: 0.82, *Rater 3*: 0.95, *Rater 4*: 0.94, *Rater 5*: 0.93, *Rater 6*: 0.93. Coefficients of 0.90 or greater are nearly always acceptable and 0.80 or greater is acceptable in most situations, particularly for the more conservative Cohen's κ (Lombard, Snyder-Duch, & Bracken, 2004).

3. Results

3.1. Description of transfer knowledge sub-categories

Think-aloud data provided a rich description of the kind of transfer activities that subjects use while learning new software concepts. Table 4 provides samples of each of the transfer knowledge sub-categories investigated in this study.

Table 4
Examples of transfer behaviors

Transfer category	Example
General knowledge	
Miscellaneous	Subject remembers that there are 8000 rows, and he calculates the number of columns and concludes that Home is the better key (using mathematical knowledge)
Metacognitive	"Uhm, I'm just thinking is it really necessary for me to do this, at this point. But, ah, I think ah I guess what maybe I'm feeling is that I'm a real novice. I've never used this before and I may as well start at the beginning"
Personal experience	"No, I was just thinking I have to tell you with things like this, I have terrible phobias. I was one of the kids in school who never got the columns right"
Hardware	
Keyboard	Subject is looking for a key that says help and IBM does not have such a key
Miscellaneous	Subject reasons that maybe a spreadsheet can have a finite number of rows and columns because there are memory limitations
Software	
Other packages	"I'm trying to figure out ... if the top of each column is to say–name, telephone, sex– or if A (the set column label) is going to be a name" (This is a Word-processing metaphor applied to the spreadsheet software package)
Concepts	Subject does not have a hierarchical model of the menu; she believes that the complete set of commands is right on the screen
Actions	"There's probably a way to highlight the entire thing and do it all but... [pause 5 seconds] Well, I was thinking there must be a way to kind of do the entire column"
Keystrokes	"I'm assuming that pressing escape would get you out of the program entirely" "Now let's see, if I just hit enter that should be the other way to move within this stuff"
Terminology	
Terminology	"Move to the command retrieve. That's a file thing so ...Uhm [searches the menu using keyboard] – I'll try file [moves to the File command] – and there's is retrieve. I'm at it" "Okay, move to the command set width. Uhm...I would guess that might be under Range"

3.2. Frequency of transfer behaviors

3.2.1. Main transfer categories

Subjects in this study used four main categories of transfer knowledge to learn the spreadsheet software. Software knowledge ($n = 724$, 55%) was used most often, followed by previous terminology ($n = 332$, 25%), hardware knowledge ($n = 163$, 12%), and general knowledge ($n = 105$, 8%).

3.2.2. Software knowledge

Previous knowledge of specific *keystrokes* and *actions* (e.g., copy, move, delete) was used most frequently by subjects, followed by *software concepts* (e.g., what is a row or column) and general references to *other software packages.* The negative and positive effects of each of the previous software knowledge sub-categories were roughly equal (Table 5).

3.2.3. Terminology

Use of terminology was the second most prolific transfer category. It is worth noting that the influence of previous terminology had a negative influence on learning two out of every three times it was used (Table 5).

3.2.4. Hardware

Knowing the *keyboard* was the most frequently observed hardware knowledge sub-category used by subjects, and more often than not had a positive effect.

Table 5
Frequency of negative and positive transfer knowledge used

Transfer sub-category	Total	Negative transfer		Positive transfer	
	N	*N*	% of total	*N*	% of total
Software					
Keystrokes	255	128	50	127	50
Software actions	229	115	50	114	50
Software concepts	138	79	57	59	43
Other packages[a]	102	62	61	40	39
Total	724	384	53	340	47
Terminology					
Total	332	224	67	108	33
Hardware					
Keyboard	155	64	41	91	59
Hardware miscellaneous	8	6	75	2	25
Total	163	70	43	93	57
General					
General misc.	49	23	47	26	53
Metacognitive	45	33	73	12	27
Personal exp.	11	9	82	2	18
Total	105	65	62	40	38

[a] Subject refers to the name of another software package but not a specific concept or action.

Previous knowledge of other hardware areas (e.g., memory) was used infrequently (Table 5).

3.2.5. General knowledge

With respect to general knowledge, general miscellaneous information (e.g., mathematics, how to use a manual) and metacognitive strategies were used most often. Metacognitive knowledge, though, produced a negative effect 73% of the time (Table 5).

3.3. Effect on learning

3.3.1. Total effect scores

With respect to overall effect on learning, *specific keystrokes* (e.g., use of PgUp, Esc, or Ctrl + arrow key) acquired with past software use helped subjects most. Previous knowledge of the *keyboard* (e.g., where keys were), *software actions*, and *terminology* had a moderate influence on learning. Referring to other software packages in a general way, miscellaneous hardware knowledge (e.g., mouse and memory), or recounting previous personal experiences had a negligible overall effect on learning (Table 6). Note that the mean influence score of software actions ($M = 0.36$) and terminology transfer ($M = 0.15$) were relatively low compared to other sub-categories.

3.3.2. Negative vs. positive transfer

It is difficult to observe the full effect of transfer, because the total effect score includes both negative and positive transfer areas. It is possible that a particular transfer area could have both a strong positive and a strong negative influence. The two polarities would cancel each other out and give a neutral estimated effect score. It is informative, then, to separate each transfer area into positive and negative transfer sub-groups, and add their absolute values.

Table 6
Total effect of each transfer area

Category	Sub-category	*N*	%[a]	Mean influence (SD)[b]	Total effect[c]	Percent
Software	Keystrokes	255	100	0.66 (1.0)	168.3	38
Hardware	Keyboard	155	97	0.66 (1.0)	99.2	22
Software	Actions	229	92	0.36 (1.3)	75.8	17
Terminology	Terminology	332	97	0.15 (1.2)	48.3	11
Software	Concepts	138	94	0.17 (1.3)	22.1	5
General	Miscellaneous	49	75	0.53 (1.0)	19.5	5
Software	Other packages	102	92	0.03 (1.3)	2.8	1
Hardware	Miscellaneous	8	19	0.00 (0.8)	0.0	0
General	Metacognitive	45	53	−0.02 (1.0)	−0.5	0
General	Personal exp.	11	19	−0.55 (1.4)	−1.1	0
Total		1324	100		435.1	

[a] Percent of subjects who used this transfer knowledge area.
[b] Scores can range from −3 to +3.
[c] Equals frequency (*N*) times percent (%) times mean influence.

For the purpose of this study, negative transfer was defined as transfer knowledge that led to no effect (influence = 0) or a negative effect (<0), whereas positive transfer was defined as knowledge that produced influence scores greater than or equal to 1. Negative and positive total effect scores were adjusted to compensate for the effect of "0" on the negative transfer mean influence score.

The results of the combined absolute values of each transfer area are presented in Table 7. This amalgamated score gives a reading of the potential of a particular transfer area to have an impact on learning.

Terminology and *software keystrokes* were the most effective transfer areas, followed by software actions, new keystrokes, and keyboard knowledge. Software concepts, knowledge of other software packages, and new knowledge actions were moderately effective. General knowledge played a relatively minor role.

A comparison of estimated effects with the combined absolute value scores (Table 7) shows that *terminology*, *software actions*, *concepts*, and *other software packages* had more impact on learning than is indicated by the total effect score. In other words, the effect of these three sub-categories was more pronounced when both negative and positive effects were considered.

3.3.3. Transfer and task type

It is reasonable to assume that different kinds of transfer knowledge are good for different kinds of tasks. For example, subjects would be expected to use keyboard-related information when trying to move around the screen (Task 1), because this task required the identification a special movement keys. On the other hand, subjects would need to draw on their knowledge of terminology when trying to find nested commands in a menu (Task 2), because this task required a categorical search through hierarchically organized list of names. Finally, knowledge of software actions and concepts would be needed when trying to enter a coordinated data set (Task 3), because this task required an understanding of software activities like copy,

Table 7
Combined absolute values of negative and positive estimated effects

Category	Sub-category	Total effect[a]	Negative effect[b]	Positive effect[c]	Combined[d]
Terminology	Terminology	48.3	−69.5	64.5	134.0
Software	Keystrokes	168.3	−13.0	80.8	93.8
Software	Actions	75.8	−39.8	47.8	87.6
Hardware	Keyboard	99.2	−10.4	48.1	58.5
Software	Concepts	22.1	−29.9	20.1	50.0
Software	Other Packages	2.8	−23.2	14.6	37.8
General	Miscellaneous	19.5	−1.9	7.3	9.2
General	Metacognitive	−0.5	−1.1	2.0	3.1
Combination	Combination	−0.4	−0.9	0.3	1.2
Hardware	Miscellaneous	0.0	−0.2	0.1	0.3
Total		435.1	−189.9	285.6	475.5

[a] Frequency times percent of subjects times mean influence.
[b] Same as total effect but includes influence scores that are negative only.
[c] Same as total effect but includes influence scores that are positive only.
[d] Absolute value of negative and positive effects.

delete, edit, and erase. These expectations were borne out when an analysis of task frequency was done as a function of task type (Fig. 1). Use of keyboard knowledge dropped from 52% during the "move" task (Task1) to 24% during the "enter data" task (Task 3). The use of previous software actions and concepts almost doubled in relative frequency when subjects were entering data (Task 3), as opposed to moving around the screen (Task 1) or finding commands on a menu (Task 2). Finally, as expected, previous knowledge of terminology peaked during the menu task (Task 2).

In summary, knowledge of software keystrokes, actions and concepts, as well as an understanding of terminology and the keyboard appeared to have the most impact, both positive and negative, on learning the spreadsheet software package. General knowledge, on the other hand, did not impinge on the learning process. The effect of transfer knowledge was also partially dependent on task type.

Learning performance – correlational analysis. For each subject, the mean influence on learning score was calculated for all 10 transfer sub categories and correlated with how well a subject performed (score given for number of tasks completed correctly). The probability level for rejection was set at 0.10 because this study is an exploratory effort. The results should be interpreted somewhat cautiously because of the high number of independent tests completed (Kirk, 1982).

Overall, transfer sub-categories with high total effect scores (reported in Table 7) correlated significantly with learning performance (Table 8). These areas included *terminology* and *all software areas* (keystrokes, actions, concepts, and other software packages). Previous knowledge of the keyboard, while high in estimated effect score did not correlate significantly with learning performance. It is possible that keyboard-related information was necessary, but not sufficient to increasing learning performance.

Ability and learning performance. It is conceivable that the effect of transfer knowledge varies with computer ability, so a regression analysis was done for all ten transfer sub-categories. Computer ability was entered in the regression equation first, followed by subject mean influence score to predict *learning performance*.

The results from the regression analysis in Table 8 are consistent with those from the total effect score and correlation analyses with one exception: after ability was

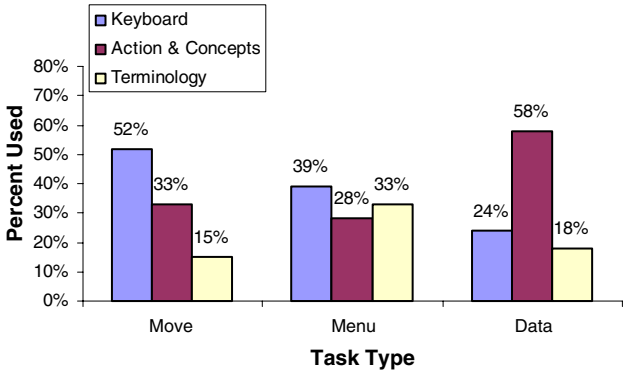

Fig. 1. Percentage of transfer knowledge used as a function of task type.

Table 8
Correlation between mean influence score for transfer knowledge and learning performance

Category	Sub-category	N[a]	Correlation	Controlling for ability[b]
Terminology	Terminology	34	0.59 ($p < 0.001$)*	$p < 0.10$*
Software	Actions	33	0.83 ($p < 0.001$)*	$p < 0.0001$*
	Concepts	34	0.66 ($p < 0.001$)*	n.s.
	Other packages	34	0.58 ($p < 0.001$)*	$p < 0.001$*
	Keystrokes	36	0.37 ($p < 0.05$)*	$p < 0.10$*
Hardware	Miscellaneous	7	0.44 (n.s.)	Sample too small
	Keyboard	35	0.25 (n.s)	n.s.
General	Personal exp.	7	0.77 ($p < 0.05$)	Sample too small
	Metacognitive	18	0.05 (n.s.)	Sample too small
	Miscellaneous	26	−0.02 (n.s.)	n.s.

Note. n.s.: not significant.
[a] Maximum number = 36.
[b] Significance of t score after entering computer ability into a regression equation.
* Probability of a Type I error is 0.65 ($\alpha = 0.10$), 0.40 ($\alpha = 0.05$), 009 ($\alpha = 0.001$), and. 0009 (=0.0001) see Kirk (1982, p. 101).

entered in the regression equation, knowledge of *software concepts* was no longer a significant predictor of learning performance. A one-way ANOVA revealed that advanced, intermediate, and beginners used software concepts with significantly decreasing expertise (Scheffé post hoc analysis; $p < 0.05$). In other words, knowledge of software concepts has varying success that is partially dependent on computer ability. This pattern, although less prominent, was also seen for terminology knowledge. Advanced subjects used terminology significantly better than intermediates and beginners (Scheffé post hoc analysis; $p < 0.05$).

4. Discussion

4.1. Overview

As suggested by many researchers (e.g., Bruner, 1986; Singley & Anderson, 1989; Sternberg, 1989), transfer knowledge appears to be one of the fundamental determinants of learning, and understanding how a new software package works is no exception. While it may seem obvious that *how much we know* is likely to have a large impact on *how much we can learn*, identifying effective units of transfer knowledge is more of a challenge. Think-aloud protocols proved to be an effective method for examining transfer knowledge used while learning software.

4.2. General knowledge

Consistent with past investigations (Larkin, 1989; Sternberg, 1989), general knowledge, when compared with more specific software knowledge, had a minor

impact on learning. Somewhat contrary to previous research (e.g., Keller, 1990; Lehrer et al., 1999; Lehrer & Littlefield, 1993), metacognitive strategies did not have a large influence on learning, although no attempt was made supplement or modify their influence. One can only conclude, that in a relatively informal, exploratory setting, metacognitive strategies do not appear frequently, nor are they influential when they do appear. Specific knowledge, based on previous software use, was much more influential with respect to altering a subject's progress.

4.3. Specific knowledge

The most influential transfer knowledge used by subjects in this study was specific software-related knowledge of keystrokes, actions (e.g., copy, paste, and delete), and concepts. The software transfer category accounted for almost 60% of total effect on learning and all software sub-categories correlated significantly with learning performance.

The effectiveness of each transfer sub-category was partially dependent on the specific task being attempted. Identifying critical areas of transfer knowledge will not be effective, if the type of knowledge does not suit the task. Transfer, at least in the area of computer software, may be more complicated than past research in other domains suggests.

Overall, influence of specific software knowledge on learning performance and specific learning tasks is consistent with the literature (Bransford et al., 1999; Stine & Wildemuth, 1990; Thorndike, 1913).

4.4. Terminology

Previous understanding of terminology appeared critical to learning. As predicted by previous, informal observations of computer software learning (e.g., Carroll & Mack, 1984; Norman, 1986; Owen, 1986), some form of adequate labeling is required to be successful. Vygotsky (1978) was a pioneer in exploring the role of language in thought. He noted that conceptual learning was a collaborative effort requiring supportive dialogue. Bruner (1986) added that changes in knowledge, thought, and learning are ''impeded, and distorted by the way in which we talk about the world and think about it in the coin of that talk'' (p. 121). A primary aspect of learning in a particular environment is the language that individuals use to communicate with each other about commonly perceived actions and events (Bruner, 1983), and the computer milieu is no exception.

4.5. Negative vs. positive transfer

It is interesting to note that specific transfer areas varied regarding the amount of negative and positive transfer observed. Knowledge of terminology, software actions and software concepts appeared to be prone to misinterpretation and error. More straightforward knowledge of the keyboard and simple keystrokes, on the other hand, resulted in positive effects a majority of the time.

A useful technique in identifying practical effects of transfer knowledge on learning was to compare and combine the absolute value of negative and positive effects. This approach identified potential "hot spot" areas of transfer and provided an estimation of overall importance. Four were noted: terminology, actions, basic concepts, and knowledge of keystrokes.

4.6. Learning context

As Rieman (1996) predicted, a number of subjects performed well in a task-specific, informal, exploratory learning environment. About two thirds of the subjects were able to successfully complete a total of 15 learning performance tasks in 55 min. In other words, without any instruction whatsoever, most subjects were able to make considerable progress based on their previous knowledge. While explicit strategies might help to improve the selection and use of correct transfer knowledge, it is clear that implicit knowledge transfer can occur, as suggested by Mayer et al. (1999).

4.7. Computer ability and transfer

It is worth noting that the effectiveness of specific transfer areas was relatively independent of computer ability. Previous knowledge of software and terminology was equally effective for beginner, intermediate and advanced computer users with one exception. Advanced users were able to put software concepts and to a lesser extent terminology to better use when they were learning. It could be speculated that beginner and intermediate level subjects did not have a sufficiently developed understanding of software concepts and terminology to improve their overall learning of the spreadsheet software. This specualtion is consistent with the results reported by Shih and Alessi (1993). However, previous computer ability could not predict learning success alone. It was necessary to examine specific types of transfer knowledge to get a more complete picture.

4.8. Suggestions for educators

Previous research suggests that domain-specific learning strategies have been successful in a variety of areas (Brown & Palinscar, 1989; Chi & Bassok, 1989; Collins, Brown, & Newman, 1989; Lampert, 1986). The results of this study present strategic areas in which educators in computer studies might consider. The following guidelines are offered:

(1) A basic understanding of the keyboard and common movement keystrokes is a good start to learning a new software package, although more advanced information is probably needed to increase learning performance.
(2) Terminology has a decided impact on learning, although misapplication of terms is responsible for considerable negative transfer. A clear understanding of terms is critical for successful learning and transfer.

(3) Focusing on basic software concepts and actions or procedures appears to be a good strategy to follow, again taking care to clear up potential misunderstandings that lead to negative responses.
(4) General knowledge about how the computer works, mathematics, and miscellaneous hardware facts do not appear to be particularly useful for learning new software.
(5) Identifying how and where a specific transfer knowledge area can be used is important. Teaching students about keyboards, for example, will not help them much with learning how to use menus, or how to enter data.
(6) Since transfer knowledge is sensitive to problem type, identifying common tasks among software packages could be an important first step to facilitate transfer. Understanding labeling idiosyncrasies of these common tasks might also improve cognitive transfer.

More research is needed to: (a) identify common task structure among a variety of software packages and (b) clarify how specific terms, concepts, and actions would assist in solving these common tasks.

4.9. Caveats

No research endeavor is without flaws – this study is no exception. These factors should be considered when interpreting the results and conclusions of this study:

(1) Although over 1300 learning activities were analyzed, the sample consisted of only 36 subjects, who were highly educated, and over 30 years of age. The results might be quite different for other populations.
(2) Only one software package was examined – spreadsheets. A variety of software packages need to be examined to increase the confidence in the results of this study.
(3) The subjects in this study had no known previous desire to learn spreadsheet software the results might be different if they had a vested interest in learning the software package presented (Holmlid, 1997).
(4) Finally, and perhaps most important, the study focused on learning performance. The results and conclusions do not necessarily apply to long term gains. This is an empirical question to be studied in the future.

5. Summary

This study is a preliminary step in understanding how previous knowledge is organized and used in a new learning new software in an exploratory environment. Previous understanding of terminology, software concepts and actions, and other software packages had the largest impact, both positive and negative, on learning. Knowledge of keystrokes and the keyboard, while overwhelmingly positive in terms

of total effect score, was not significantly related to learning performance. Knowing the keyboard may be useful, but not sufficient for performing well on new software. Computer ability had little impact on the type of transfer knowledge used, except with respect to software concepts and to a lesser extent, terminology. The interaction between problem type and effectiveness of a specific transfer area, suggests that identifying appropriate units of cognitive transfer rests partially on identifying common tasks among software packages. It is equally important that users understand labeling idiosyncrasies of these common tasks.

References

Anderson, J. R., Reder, L. M., & Simon, H. A. (1996). Situated learning and education. *Educational Researcher, 25*(4), 1–7.

Bakeman, R. (2000). Behavioral observation and coding. In H. T. Reis & C. M. Judge (Eds.), *Handbook of research methods in social and personality psychology* (pp. 138–159). New York: Cambridge University Press.

Bellis, M. B. (2004). Microsoft Windows. Retrieved February 29, 2004. Available from: www.inventors.about.com/library/inventors/blwindows.htm.

Bransford, J. D., Brown, A. L., & Cocking, R. R. (1999). *How people learn: Brain, mind, experience, and school*. Washington, DC: The National Academic Press.

Brown, A. L., & Palinscar, A. S. (1989). Guided, cooperative learning and individual knowledge acquisition. In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 393–451). Hillsdale, NJ: Erlbaum Associates.

Bruner, J. (1983). *Child's talk. Learning to use language*. Toronto, Canada: George J. McLeod Ltd.

Bruner, J. (1986). *Actual minds, possible worlds*. Cambridge, MA: Harvard University Press.

Carroll, J. M., & Mack, R. L. (1984). Learning to use a word processor: By doing, by thinking, and by knowing. In J. C. Thomas & M. Schneider (Eds.), *Human factors in computer systems*. Norwood, NJ: Ablex.

Carroll, J. B. (1990). *The Nurnberg funnell*. Cambridge, MA: MIT Press.

Chi, M. T. H., & Bassok, M. (1989). Learning from examples via self-explanations. In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 251–282). Hillsdale, NJ: Erlbaum Associates.

Cohen, J. (1960). A coefficient of agreement for nominal scales. *Educational and Psychological Measurement, 20*(Winter), 37–46.

Collins, A., Brown, J. S., & Newman, S. E. (1989). Cognitive apprenticeship: Teaching the crafts of reading, writing, and mathematics. In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 453–494). Hillsdale, NJ: Erlbaum Associates.

Dewey, M. E. (1983). Coefficients of agreement. *British Journal of Psychiatry, 143*, 487–489.

Ericsson, K. A., & Simon, H. A. (1980). Verbal reports as data. *Psychological Review, 87*(3), 215251.

Franzke, M., & Rieman, J. (1993). Neutral training wheels: Learning and transfer between two versions of a computer application. *Proceedings of the Vienna Conference on Human Computer Interaction* (93, pp. 317–328). Berlin: Springer.

Halpin, R. (1999). A model of constructivist learning in practice: computer literacy integrated into elementary mathematics and science teacher education. *Journal of Research on Computing in Eduation, 32*(1), 128–138.

Holmlid, S. (1997). Transfer of training: An attempt to interpret subjectively perceived usability. In *Proceedings of the STIMDI '97 Conference, Linköping, Sweden*.

Keller, J. K. (1990). Characteristics of logo instruction promoting transfer of learning: A research review. *Journal of Research on Computing in Education, 23*(1), 55–71.

Kirk, R. E. (1982). *Experimental design* (2nd ed.). Belmont, CA: Wadsworth.

Lampert, M. (1986). Teaching multiplication. *Journal of Mathematical Behaviour, 5*, 241–280.

Larkin, J. H. (1989). What kind of knowledge transfers? In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 283–305). Hillsdale, NJ: Erlbaum Associates.

Lehrer, R., Lee, M., & Jeong, A. (1999). Reflective teaching of logo. *Journal of the Learning Sciences, 8*, 245–288.

Lehrer, R., & Littlefield, R. (1993). Relationships among cognitive components in logo learning and transfer. *Journal of Educational Psychology, 85*(2), 317–330.

Lombard, M., Snyder-Duch, J., & Bracken, C. C. (2004). Practical resources for assessing and reporting intercoder reliability in content analysis research projects. Retrieved September 2004. Available from: www.temple.edu/mmc/reliability.

Mayer, R. E., Quilici, J. H., & Moreno, R. (1999). What is learned in an after-school computer club? *Journal of Educational Computing Research, 20*, 215–227.

Norman, D. A. (1986). Cognitive engineering. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives on human–computer interaction* (pp. 31–65). Hillsdale, NJ: Lawrence Erlbaum Associates.

Owen, D. (1986). Naive theories of computation. In D. A. Norman & S. W. Draper (Eds.), *User centered system design: New perspectives on human-computer interaction* (pp. 187–200). Hillsdale, NJ: Lawrence Erlbaum Associates.

Reder, L., & Klatzky, R. L. (1994). The effect of context on training: Is learning situated? In D. Druckman & R. A. Bjork (Eds.), *Learning, remembering, believing: Enhancing human performance* (pp. 25–56). Washington, DC: National Academic Press.

Resnick, L. B. (1989). Introduction. In L. B. Resnick (Ed.), *Knowing, learning, and instruction* (pp. 1–24). Hillsdale, NJ: Erlbaum Associates.

Rieman, J. (1996). A field study of exploratory learning strategies. *ACM Transactions on Computer–Human Interaction, 3*(3), 189–218.

Shneiderman, B. (1989). Designing the user interface. In T. Forester (Ed.), *Computers in the human context* (pp. 166–173). Cambridge, MA: The MIT Press.

Shayo, C., & Olfman, L. (1998). The role of conceptual models in formal software training. In *Proceedings of the 1998 ACM SIGCPR conference on computer personnel research, Boston, MA*.

Simpson, H. K., & Pelligrino, J. W. (1993). Descriptive models in learning command languages. *Journal of Educational Psychology, 85*(3), 539–550.

Singley, M. K., & Anderson, J. A. (1989). *The transfer of cognitive skill*. Cambridge, MA: The Harvard University Press.

Shih, Y., & Alessi, S. M. (1993). Mental models and transfer of learning in computer programming. *Journal of Research on Computing in Education, 26*(2), 164–175.

Sternberg, R. J. (1989). Domain-generality versus domain-specificity: the life and impending death of a false dichotomy. *Merrill-Palmer Quarterly, 35*(1), 115–130.

Stine, W. D., & Wildemuth, B. M. (1990). The training of microcomputer users: insights from two disciplines. *Journal of Education for Library and Information Science, 33*(2), 100–109.

Thorndike, E. (1913). *Educational psychology: The psychology of learning*. New York: Teachers College Press.

Turkle, S. (1984). *The second self*. New York: Simon & Shuster.

Underwood, D. M., & Underwood, G. (1990). *Computers and learning*. Cambridge, MA: Basil Blackwell Ltd.

Voss, J. F. (1989). Problem solving and the educational process. In A. Lesgold & R. Glaser (Eds.), *Foundations for a psychology of education* (pp. 251–294). Hillsdale, NJ: Lawrence Erlbaum Associates.

Vygotsky, L. S. (1978). *Mind in society*. Cambridge M.A: Harvard University Press.

Yamnill, S., & McLean, G. N. (2001). Theories supporting transfer of training. *Human Resource Development Quarterly, 12*(2), 195–208.