The Role of Training in Preparing End Users to Learn Related Software

CONRAD SHAYO, California State University, San Bernardino
LORNE OLFMAN, Claremont Graduate University

The aim of this paper is to determine what types of formal training methods can provide appropriate “mapping via training” of a new but related database management system (DBMS) given that “mapping via analogy” is also taking place. To this end, trainees’ existing mental models were measured prior to being exposed to one of three training methods. Training methods were varied in an experimental setting across two dimensions: the training task context (generic and/or relevant), and the number of DBMSs demonstrated (one and/or two). Outcomes were measured in terms of learning performance and perceptions of ability to transfer skills to a new but related DBMS. The results indicate that both task context and the number of software packages learned are important training variables that influence trainees’ mental models of the software, their transfer self-efficacy expectation, and their perceptions about the usefulness of the training.

Introduction

Organizations always acquire new or related versions of end-user software, but little is known about how existing end-user mental models (knowledge) acquired through previous training prepare them to learn a related package. For example, an organization that previously used a DOS-based database management system (DBMS) like dBase IV might switch to a Windows 95 environment using Microsoft Access. The two relational DBMSs perform similar functions, but the interface and specific features are different. To what extent can previous knowledge or mental models of dBase IV help or hinder users to learn Microsoft Access? Moreover, what kinds of training methods can best lead to more accurate mental models? Answers to these questions will help organizations determine their long-term software training strategy.

A user’s mental model of a package is defined as the existing structure of knowledge (declarative and procedural) which is activated into working memory at any one time when the end user thinks about using or learning a target package to accomplish some task (Rumelhart, 1980; Wilson & Rutherford, 1989). It is a mental representation of existing knowledge about the package. Mental models are acquired and reinforced through a set of processes called “mapping.” Mapping is accomplished by formal training, usage, or calling upon other mental models. This latter process is termed “mapping via analogy” (Sein, Bostrom, & Olfman, 1987; Sein, Bostrom, Olfman, & Davis, 1993). For example, computer directory structures can be thought to be analogous to filing cabinets, or word processing is analogous to typing.

The aim of this paper is to determine what types of formal training methods can provide appropriate “mapping via training” of a new but related DBMS given that “mapping via analogy” is also taking place. To this end, trainees’ existing mental models were measured prior to being exposed to one of three training methods. Training methods were varied in an experimental setting across two dimensions: the training task context (generic and/or relevant), and the number of DBMSs demonstrated (one and/or two). Outcomes were measured in terms of learning performance and perceptions of ability to transfer skills to a new but related DBMS. The results indicate that both task context and the number of software packages learned are important training variables that influence trainees’ mental models of the software, their transfer self-efficacy expectation, and their perceptions about the usefulness of the training.
followed by a description of the research methods, a presentation of results, and a discussion and conclusion of the value of these results.

**Background**

With rapid changes in hardware and software technology, organizations are constantly upgrading and changing their end-user software suite. For example, the introduction of Windows 95 created a new set of applications that were packaged with new computers (e.g., Microsoft Office 95 software). Users cannot expect to operate only one version of a particular application over their working life, but instead must confront multiple packages that are related. As such, there is need for organizations to consider the best way to train users so that the conversion process is most effective.

Successful formal software training should lead to an accurate mental model of the software (Sein et al., 1987; 1993). The “mental model” construct is defined in detail in Appendix A. An accurate mental model is expected to increase the trainees’ (a) learning performance, (b) attitude toward using the package, and (c) attitude toward using the mental model of the target software package to learn newer related packages. This last outcome has not received adequate attention in software training research. The outcome is important because the mental model of a package formed as a result of training will most likely influence future learning of similar packages. According to Gick & Holyoak (1987):

> ... any salient similarity of two situations [software packages] will influence their [trainees’] overall perceived similarity, which will in turn affect retrieval of the representation of the training during the transfer task — the greater the perceived similarity of the two situations, the more likely it is that transfer [will be attempted ... the direction of transfer [will be determined by the similarity of the two situations with respect to the features causally relevant to the goal or required response in the transfer task.... (p. 16).

It is reasonable, therefore, to expect trainers to strive to influence trainees’ beliefs about their ability to apply current concepts to future learning of related packages.

This study intends to explore how and to what extent mental models formed by end users in one training program influence their current learning performance, and their attitude toward learning a related package. Learning performance is measured in terms of perceived ease of learning (also called self-efficacy of learning), perceived usefulness of learning, satisfaction from learning, and behavioral intention to learn. This study extends previous research by Bostrom, Olfman, and Sein (1990) on software training by adding attitude toward learning a related package as one of the important outcomes of transfer of knowledge and skills acquired from software training.

---

**Relevant Previous Literature**

Previous research in human factors and software training indicates that any study that investigates transfer of knowledge from one software package to another should consider both the training task context and the number of software packages used as critical variables.

**Training Task Context**

Two studies identified in the literature, Olfman & Bostrom (1991) and Lehrer and Littlefield (1993), manipulated the training task context while maintaining the content of the training. Olfman and Bostrom’s study was based on subjects who learned to use a Lotus 123 software package. Lehrer and Littlefield’s study was based on teaching students the Logo programming language.

Olfman and Bostrom conducted a field experiment that compared construct-based versus applications-based training. The task context for the construct-based trained subjects consisted of generic problem-solving exercises using the language of a reference manual. Such context was considered less relevant to the trainees. Applications-based trainees were provided with a non-generic training task context (that was more personally relevant) by asking trainees to focus on what the software could do for them. They reasoned that providing a meaningful or relevant training task context helps adult learners to link the knowledge to the familiar and to make the familiar one instance of a more general case (Gick and Holyoak, 1987). This also increases their motivation and increases their perceived confidence (self-efficacy) to perform similar tasks (Carroll, 1990). Their study found limited direct effect between application-based training and later usage of Lotus 123. In the Lehrer and Littlefield study, one group of subjects was provided with only generic examples and ideas found in Logo-related activities (microcontext-based instruction), and the other group was provided with examples and ideas found in Logo, as well as relationships between Logo and non-Logo problems. The authors hypothesized that a more meaningful training task context provides “the likelihood for ‘mindful abstraction’ from Logo [the target software] to other related contexts” (Lehrer and Littlefield, 1993, p. 319). They showed that effective exercises are ones that provide a context that makes it possible for trainees to establish specific links to their conceptual and operational knowledge when faced with a problem situation.

**Number of Software Packages**

A review of the literature on transfer of learning in human factors shows that a number of researchers have investigated transfer of learning by studying how trainees’ knowledge of one “device” or “simulator” facilitates the learning of another new (related or unrelated) “device” or “simulator.” Dixon and Gabrys (1991) found significant transfer effects of devices that were procedurally similar but insignificant effects in devices that were conceptually simi-