December 2001

Exploring Differential Impacts of Autonomous Learning and Induced Learning on End-User Software Training

Xiaodong Deng
Oakland University

Balaji Rajagopalan
Oakland University

Ravi Krovi
The University of Akron

Exploring Differential Impacts of Autonomous Learning and Induced Learning on End-User Software Training

Xiaodong Deng
Oakland University
eng@oakland.edu

Balaji Rajagopalan
Oakland University
rajagopa@oakland.edu

Ravindra Krovi
The University of Akron
krovi@uakron.edu

Abstract

Effective end-user training/learning plays a critical role in organizations for successfully implementing information systems and promoting productive use of the implemented systems (Bostrom et al., 1990; Compeau & Higgins, 1995; Lim et al., 1997). This research tries to explore the differential impacts of autonomous learning (learning that happened in a natural working environment through the routine use of the information technology for work) and induced learning (learning that happened through formal efforts such as training, seminars, or workshops) on the effectiveness of end-users' learning of different software packages. The study focuses on the nature of software packages and hypothesizes that different categories of software packages need different considerations in learning/training. The development of taxonomy of software package facilitates future research activities and also helps IS managers select different training/learning strategies for end-users. The study also emphasizes two types of learning activities (autonomous and induced) for the learning process and suggests that a training program may be more effective when it combines the applications of the software package in end-users' natural working environment. This insight should provide both researchers and IS managers a broader view of end-users' information technology learning.

Introduction

End-user training/learning plays a critical role in organizations for successfully implementing information systems and promoting productive use of the systems (Bostrom et al., 1990; Compeau & Higgins, 1995; Lim et al., 1997). Effective training/learning programs will greatly expedite end-users' learning process where the users learn how to use the functions built in the systems or the associated software packages to complete their work. The knowledge and expertise received through the training thus increase the end-users' self-efficacy (Compeau & Higgins, 1995) and motivate them to use the systems more effectively, which, together with the enhanced skills, may lead to the realization of the systems' full potential (Bostrom et al., 1990).

Studies have examined the impact of learning methods and styles on learning performance. For example, Bostrom et al., (1990) examine the impact of individual learning style and training methods on learning performance. Compeau and Higgins (1995) look at the effect of prior performance and computer self-efficacy on current performance. Lim et al., (1997) examine the impact of learning methods of co-discovery and self-discovery on learning. Other perspectives offered by prior research include looking at the user training impact on the acceptance of information technology, learning style impact on the usage level and how attitude toward training effects training motivation. Examples include Nelson and Cheney's (1987) study that examines the impact of training techniques on the acceptance of information technology and the research conducted by Olfman and Mandviwalla (1994) that investigates the impact of learning style on usage.

Although several studies have focused on examining the individual learning methods and styles on performance, few have examined the learning impact by software categories; especially the differential performance resulted from a differing combination of autonomous learning and induced learning. Building on the existing research, this study first identifies taxonomy of software package along the nature (i.e., application vs. tool) and the purpose (i.e., general vs. special) of packages. It then adapts two learning types (autonomous vs. induced) from a manufacturing area to learning and hypothesizes that both learning types will improve learning proficiency but the effectiveness of the learning will differ from different types of software packages.

Taxonomy of Software

We use a classification based on two dimensions: software type and software purpose (see Figure 1). The choice of the dimensions is to facilitate the focus on the type of learning mechanism that will impact learning along the dimensions. The software type is examined at two levels: Development tools and Applications. Development tools may include programming languages, middleware, computer-aided software engineering (CASE) tools, database querying languages (SQL) and the like. The purpose of software is also viewed at two levels: General and Special (Stair & Reynolds, 2001). General-purpose software packages are those that can be used for a variety of scenarios. Special purpose software is designed for specific tasks. Now, let us examine the cells resulting from crossing these dimensions. General-purpose development tools like C or C++ will require similar learning mechanisms as their focus is on conceptual understanding and logic. On the other hand, general-purpose applications like MS Word will require more hands-on learning. Special purpose development tools like CASE tools may require an equal focus on conceptual understanding and hands-on learning. Finally, special purpose applications like computer-aided design (CAD) and enterprise resource planning (ERP) systems may also require a balance of conceptual understanding and hands-on learning.

		Purpose of Software	
		General	**Special**
Type of Software	**Development Tools**	Programming languages (e.g., C++, Java)	SQL; Cool:Gen; ColdFusion
	Applications	Office Suite (Word, PowerPoint, Excel)	CAD; CAE; ACL, ERP

Figure 1. Taxonomy of Software

An End-User Software Learning Model

The concepts of autonomous learning and induced learning were originally proposed by Levy (1965) in a manufacturing setting and extended by Dutton and Thomas (1984). Autonomous learning here means the understanding of a software package through the repetitive usage of the package for the routine work in an end-user's natural working environment. Induced learning, on the other hand, refers to the formal efforts that an end-user makes to learn the software and its application. A model for an end-user's software learning is thus proposed for empirical investigation (see Figure 2). The model hypothesizes that an end-user's autonomous learning activities and induced learning efforts determine the user's learning performance of a software package and its applications.

Autonomous learning activities include the number of same or similar tasks completed by using the software, the complexity of the completed tasks, and the variety of the completed tasks. Induced learning efforts include the formal efforts an end-user made in participating training sessions, joining focused sessions, and conducting self-studies. The learning performance here is measured by the comprehension of frequently used or job related functions of the software package, the application of these functions, and the logic/concepts behind the applications.

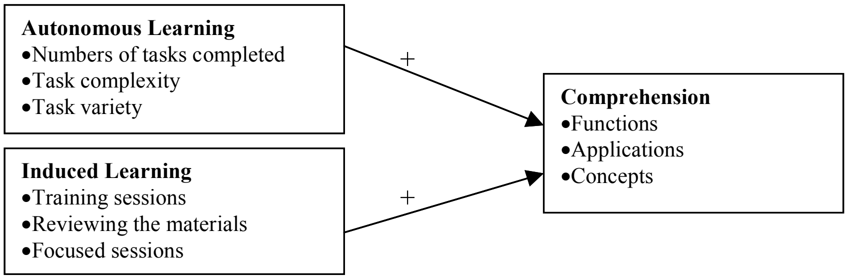

Figure 2. An End-User Software Learning Model

In the working environment, the purpose of an individual using a software package is to solve a problem, help make decisions, integrate workflows, and/or plan one's work (Doll & Torkzadeh, 1998) rather than to learn how to use the software package itself. The more an individual user use the package for different tasks, the deeper understanding of the software the user can develop.

End-users thus can improve their proficiency of using the software and its applications through the simple repetition of their routine work. This leads to following hypothesis:

> *H-A1: Autonomous learning activities an end-user makes for a software package will be positively associated with the end-user's comprehension of the software package.*

Induced learning can be achieved through participating training sessions and focused sessions (i.e., group studies) and/or self-studying the materials on the software and its application. All these types of formal/conscious learning efforts will enhance the understanding of the functions built in the software, the applications of the functions, and the logic/concepts behind the applications. Thus, the study hypothesizes:

> *H-A2: Induced learning efforts that an end-user makes for a software package will be positively associated with the end-user's comprehension of the software package.*

While studying the impacts of different individual learning styles and training techniques, previous studies have also indicated that different software packages might need different considerations (Olfman & Mandviwalla, 1994; Bostrom et al., 1990; Nelson & Cheney, 1987). For application type, general-purpose software package such as MS Office Suite, learning how to use it often requires only common sense because most of the functions are self-explained. The tasks that the software supports are normally routine-based office work. The more an end-user uses the package, the higher proficiency and more knowledge of the applications the user will obtain. Thus, following hypothesis is derived for empirical examination:

> *H-B1: The autonomous learning will have a stronger impact than the induced learning on individuals' proficiency of application type, general-purpose software packages.*

For application type, special purpose software packages such as computer-aided design (CAD), users need to learn the functions included in the package. While the users may have knowledge on tasks, the functions may not match the task and the users need to spend time to figure out how to apply the functions to a specific task. Tool type, special purpose software packages such as SQL demands more analytical/conceptual thinking in learning. End-users have to make efforts to understand the concepts of the functions before they can properly apply them. Thus for both types of software packages, the research hypothesizes:

> *H-B2: The autonomous learning will have the same impact as the induced learning on individuals' proficiency of application type, special-purpose and tool type, special-purpose software packages.*

Tool type, general-purpose software packages such as high level programming languages demand highly conceptual and analytical skills in grasping the concepts of them. Without understanding the concepts, the effectiveness of the applications of the software will be limited and the ineffectiveness will influence the individual's learning performance. The research thus hypothesizes:

> *H-B3: The induced learning will have a stronger impact than the autonomous learning on individuals' proficiency of tool type, general-purpose software packages.*

Further Researches

Future researches should develop measurement instruments to empirically test the proposed research model. This involves a rigorous process where measure items for each construct in the model have to be generated based on the theory or empirical studies, the items then have to be pilot tested and refined based on the pilot study, and then the norms can be developed through a large-scale study.

References

Bostrom, Robert P., Olfman, Lorne, and Sein, Maung K., "The Importance Of Learning Style In End-User Training," *MIS Quarterly* (14:1), 1990, pp. 101 – 119.

Compeau, Deborah R, and Higgins, Christopher A, "Application Of Social Cognitive Theory To Training For Computer Skills," *Information Systems Research* (6:2), 1995, pp. 118 – 143.

Doll, William J., and Torkzadeh, Gholamreza, "Developing a Multidimensional Measure of System-Use in an Organizational Context," *Information & Management* (33:4), 1998, pp.171-185.

Dutton, J. R., and Thomas, A., “Treating Progress Functions as a Managerial Opportunity,” *Academy Management Review* (9), 1984, pp. 235 – 247.

Levy, F. K., “Adaptation in the Production Process,” *Management Science* (11), 1965, pp. B136 – B154.

Lim, Kai H, Ward, Lawrence M., and Benbasat, Izak, “An Empirical Study of Computer System Learning: Comparison of Co-Discovery and Self-Discovery Methods,” *Information Systems Research* (8:3), 1997, pp. 254 – 272.

Nelson, R. Ryan, and Cheney, Paul H., “Training End Users An Exploratory Study,” *MIS Quarterly* (11:4), 1987, pp. 547 – 559.

Olfman, Lorne, and Mandviwalla, Munir, “Conceptual Versus Procedural Software Training For Graphical,” *MIS Quarterly* (18:4), 1994, pp. 405 – 426.

Stair, Ralph M., and Reynolds, Gorge W., *Principles of Information Systems* (5th edition), Boston, MA: Course Technology, 2001.