Ritter, F. E., Baxter, G. D., Kim, J. W., & Srinivasmurthy, S. (2013). Learning and retention.
In J. D. Lee & A. Kirlik (Eds.), The Oxford Handbook of Cognitive Engineering (pp. 125-142). New York, NY: Oxford.

Chapter: Learning and Retention

Frank E. Ritter, Gordon D. Baxter, Jong W. Kim, and Sowmyalatha Srinivasmurthy

Date: 17 Oct 2011 / Due 1 aug 2011
learning and retention v21.doc

(To appear in the *Handbook of Cognitive Engineering,* John Lee and Alex Kirlik, OUP)

Manuscript Statistics

Word count: approx. 9.9k excl. refs but including figures (target 7-8k words)
Figures: 6
Tables: 0

Corresponding Author:

Frank E. Ritter (frank.ritter@psu.edu)
College of Information Sciences and Technology
The Pennsylvania State University
University Park, PA 16802

Abstract

Learning and retention are important for design. Learning provides performance improvements, sometimes dramatic improvements with practice on a task or with an interface. These improvements can make designs that appear too slow on first use to become more usable and remove errors in performance. Learning, the rate at which it occurs, the stages of learning, how to predict it, and when learning transfers are all important aspects of human behavior for design. Retention is also important. The retention curve is similar to the learning curve in that dramatic skill loss is possible with the passage of time, which can be dangerous. The knowledge we have about learning and retention yields numerous suggestions for system design, including how to evaluate interfaces (with trained or novice testers), how and how often to train (to retain declarative knowledge and to create proceduralized knowledge where possible), and how to present information (so that it can be learned).

Keywords: Learning, retention, learning curve, transfer, skilled behavior

Acknowledgements

Preparation of this manuscript was partially sponsored by a grant from the Division of Human Performance Training, and Education at the Office of Naval Research, under Contract N00014-11-1-0275 (Ritter & Kim), and by funding from the UK EPSRC's Large Scale Complex IT Systems project (Baxter). The views and conclusions contained in this report are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government or the Pennsylvania State University.

Table of Contents

1. Introduction

People get faster and make less noticeable errors as they repeatedly perform the same task. This is what most mean when they refer to learning. More formally, learning can be seen as the relatively permanent change in response potentiality that takes place as a result of extended practice (Kimble, 1961).

By following a human-centered approach to design, we explicitly acknowledge that we are dealing with a population that encompasses users with a range of skills and abilities. We know that these skills and abilities will change over time as a result of learning, so we should explicitly support the learning process when designing systems. To do so we need to understand how learning takes place, and how it improves performance through practice, sometimes dramatically. Systems that initially appear slow and effortful to use, often become quicker and more usable with practice—learning is therefore something that users and designers can exploit to compensate for other design decisions.

In this chapter we discuss the relationship between learning and skilled behavior, noting the sorts of performance improvements that occur over time, and the process that underpins these improvements with respect to design. The performance improvements can be modeled using learning curves. There are ongoing debates about the curve's shape, with corresponding debates about the learning mechanisms that have implications for cognitive science and the underlying architecture of cognition. Performance can degrade over time too, through lack of continued practice. We describe how this relationship can be modeled using retention curves. We conclude with summary lessons about the implications for system design, and note some areas where further research is still required.

2. Learning and Skilled Behavior

Performance changes with practice, typically getting faster, less effortful, and exhibiting fewer noticeable errors. These changes apply to most tasks that people carry out, including the use of computer-based systems. As users perform tasks, they acquire new information and skills and, through practice, improve the strength of their memories. More frequent retrieval of information or learnt skills may lead to the improvement of memory strength too: retrieving a password is faster and more robust after 2 attempts than 1, and after 100 attempts than 99 (all else being equal), although the change per practice attempt decreases with successive attempts.

Human performance can be described using Neisser's (1976) Perceive-Decide-Act cycle and similar frameworks (e.g., Coram, 2002; Newell, 1990). People perceive the current situation, which leads them to make decisions about which actions to take, and then perceive the updated situation, and so on. Elements in the task environment are initially perceived individually, but related elements become grouped together with repeated exposure. Perception of the situation thus becomes more holistic with practice, and recognition consequently becomes quicker. People also come to associate decisions and responses with these learned patterns. In the same way, users can learn to

adjust their behavior to accommodate the idiosyncrasies of a particular interface: where it is not immediately obvious which actions can be performed, users typically must learn which actions to perform. Similarly, users often have some predetermined beliefs about where particular objects should be on the screen, and if they do not appear where they are expected, their location must be learned. These predetermined beliefs can be called a mental model (Johnson-Laird, 1983; Moray, 1999), and represent a theory of how the world is structured and works. If interfaces are appropriately designed—and support the user's mental model or other learning mechanisms—learning can happen without causing the user much distress; poorly designed interfaces are usually not appreciated because considerable time and effort has to be expended on learning.

We present a summary, explanatory theory of how users learn, building on existing theories. Our theory has many implications, including how user's behavior changes with learning, what extreme practice looks like, and suggestions for system design. While we tailor our discussions towards learning in the use of system design, there are many texts on psychology and cognitive science that provide more general discussions on human learning (e.g., Anderson 1982; 1995; 2004; Lindsay & Norman, 1977).

2.1 Improvements from Learning

Perhaps the biggest regularity of human behavior in general, and particularly user behavior, is the changes that occur due to learning, which leads the learners to get faster at performing a task the more they do it (Ritter & Schooler, 2001). In most tasks where it has been studied, the relationship between the number of trials and performance time follows similar shaped curves. These so-called learning curves apply across a wide range of tasks, including pushing buttons, reading strange text, doing arithmetic, typing, using a computer, generating factory schedules, all the way up to writing books (Crossman, 1959; Ericsson, Charness, Feltovich, & Hoffman, 2006; Nerb, Krems, & Ritter, 1993; Newell & Rosenbloom, 1981; Ohlsson, 1992; Seibel, 1963), and people working together in groups (e.g., Lieberman, 1984; Wright, 1936).

There are large improvements initially, although users rarely report satisfaction with these initial improvements (they call it a 'steep learning curve'!). These improvements follow a monotonically decreasing curve over time. For example, in the Seibel task (1963), participants are shown a pattern of 10 lights and have to push buttons corresponding to those lights that are illuminated. Each point on the curve for this task (see Figure 1) represents the average reaction times for 1,000 patterns. With practice, behavior improves but by a reducing amount. Over a wide variety of tasks and out to hundreds of thousands of trials, people continue to get faster (e.g., Crossman, 1959; Seibel, 1963) and to make less errors (Rabbit & Banerji, 1989).

With such long time scales it becomes difficult to see the incremental improvements in performance time. Using logarithmic axes (see Figure 1 (b)) makes it easier to perceive the incremental changes. Typically, on these log-log plots, learning curves more or less follow a straight line.

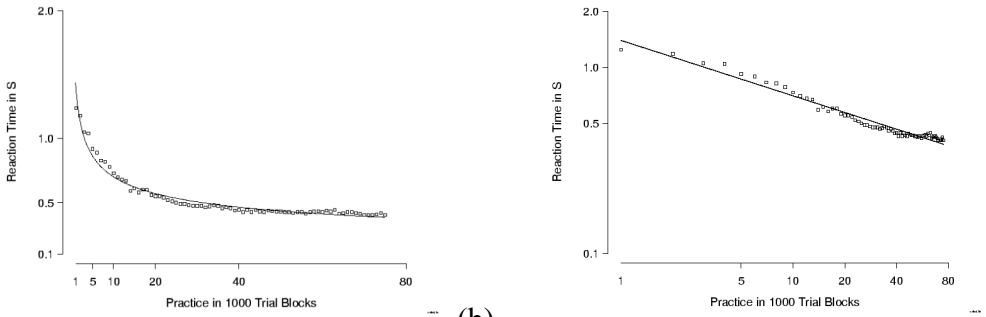


Figure 1. Time to perform a simple task (pushing a combination of buttons): (a) on a linear plot, and (b) on a log-log plot. The solid line on each plot shows a power law fit to the data. (Adapted from Seibel, 1963, and previously used in Ritter and Schooler, 2001).

The mathematical representation of the learning curve is currently disputed. While most learning curves generally follow a straight line in log-log plots, there are some systematic discrepancies. Several people have noted that the smoothness of the curve arises through aggregation, and is probably not a smooth curve on an individual level (e.g, Brown & Heathcote, 2003; Delaney, Reder, Staszewski, & Ritter, 1998). Noise in the curve may be due to differences in task difficulty (Nerb et al., 1993) and differential transfer between tasks (Ritter & Bibby, 2008), as well as other unknown factors. Also, some researchers believe that the learning curve represents an exponential function (Heathcote, Brown, & Mewhort, 2000), described by Equation 1, while others think that it is a power function (thus, called the power law of learning), described by Equation 2 (Delaney et al., 1998; Newell & Rosenbloom, 1981).

$$Time\ of\ a\ trial = Const1 + Const2 * e^{-b(numberoftrial)} \qquad \textbf{Equation 1}$$

$$Time\ of\ a\ trial = Const1 + Const2 * (number\ of\ trial + PP)^{-\alpha} \qquad \textbf{Equation 2}$$

In these equations *Const1* is the minimum time based on the machinery or external environment, the asymptotic time. It is sometimes omitted because it makes the regression harder to compute. It can be computed from equations describing the world or by using physical equipment. For example, you might measure with a camera how long it takes the equipment to provide materials to the user, or how long it takes the computer to respond to input and provide the next stimuli. *Const2* is the time that decreases with practice; it is always used.

PP is previous practice on the task. *PP* is either estimated, measured directly, or ignored (in the exponential equation it can be absorbed into Const2, Heathcote et al., 2000). It is usually ignored because the task studied is usually unique enough to the learner that previous practice can be ignored. In other cases, such as taking up a

familiar task, the equation does not fit as well if it is omitted. b is a constant to normalize the speed of learning.

α is typically in the range 0.1 ~ 0.5 (reviewed in Newell, 1990, Ch. 1; also see Anderson, 1995, Ch. 6, and Heathcote et al., 2000 for example values). α is usually found by plotting the data using log-log axes, and fitting a straight line using a linear equation (with a known closed form solution) to the log of the trials and the log of the time. The more accurate, but more computationally expensive approach, is to fit the power law equation, including the appropriate constants, using an iterative algorithm.

Being able to mathematically predict learning is important for several reasons. For science, it helps summarize learning, and comparing the constants (*Constant1*, *Constant2*) provides a useful way of characterizing tasks. It also has practical applications. In engineering, design, and manufacturing, it can be used to predict how fast users will become learn with practice, and thus can be used to predict factory output and profitability (e.g., Lieberman, 1984; Wright, 1936).

An alternative interpretation of the way improvements occur through learning suggests that the match to a power law is an artifact of averaging the data from multiple people and multiple series. The curve is, therefore, perhaps best described as an exponential when the data is examined on an individual learner basis (Heathcote, Brown, & Mewhort, 2000), but a powerlaw for individual by strategy (Delaney, Reder, Staszewski, & Ritter, 1998). Both equations for predicting learning have basically the same implications for users with limited lifetimes and physical equipment. (They have, however, different implications for the details of learning and its relation to the mechanisms of cognition and knowledge.)

In addition to reducing performance time, several other aspects of behavior improve with practice too (Rabbit & Banerji, 1989). The number of noticeable errors decreases, usually because people detect and correct the errors before their consequences become apparent to casual observers. The variance in task time also decreases (Crossman, 1959; Rabbit & Banerji, 1989). Some believe that this decrease in variability is the greatest contributor to improvements in speed, because the minimum time to perform a task generally remains unchanged with practice. However, there are notable exceptions such as those tasks that cannot be completed by novices.

There appear to be two instances where learning does not lead to faster performance. The first instance is a long term effect—users cannot get faster when the machinery they are using to perform the task is the limiting factor. This was first noted when cigar rollers improved up to a point and then stopped. It was found that the users were rolling faster than the machine could pass them materials (Crossman, 1959). This constraint is represented by *Constant1* in the equations above.

The second instance is a short term effect that occurs when users change strategies. As a user adopts a new strategy, they often experience a slowdown, moving back on the practice curve for that strategy. Work on deliberate practice and expertise suggests that strategy changes may be particularly important for becoming expert (Ericsson, Krampe, & Tesch-Roemer, 1993). With further practice, performance on this curve with a lower intercept improves, usually to be much better than the previous strategy (Delaney, Reder, Staszewski, & Ritter, 1998) (see Figure 2).

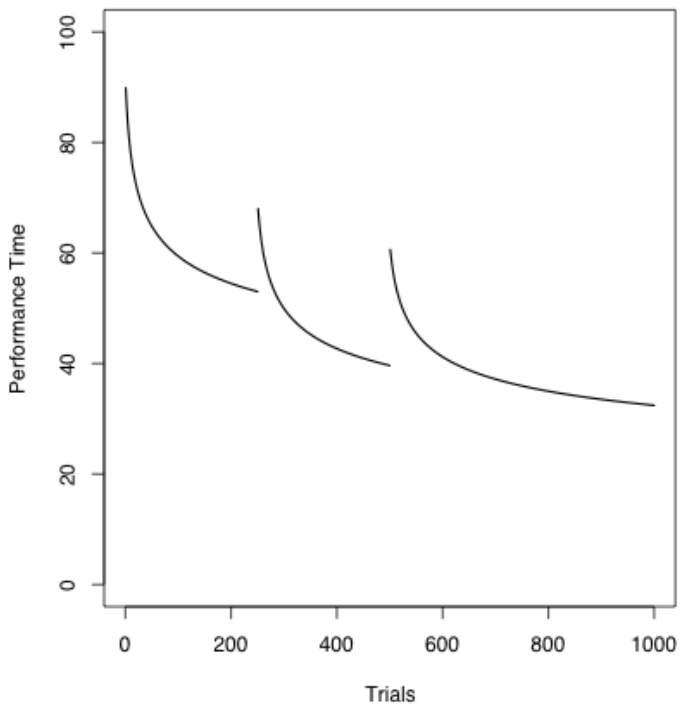


Figure 2. The learning curve for a series of better strategies, showing the initial slow down and then savings due to switching strategies. (Diagram by Ritter, learning-strat-figure.pdf)

Users often do not adopt the optimal learning strategy. In text editors, for example, there are several ways to find a target string, including scrolling line-by-line, scrolling by paragraph, and searching. In one study (Card, Moran, & Newell, 1983), most users employed a sub-optimal strategy (scrolling line-by-line) rather than searching; experts typically use searching, which can be 100 times faster than scrolling. This effect is also seen in web browsers (see, for example, Nielson http://www.useit.com/alertbox/search-skills.html), a wide range of applications (Rieman, 1996), and more recently in search engines (Jansen & Spink, 2006) and browsers[1].

Recent work has highlighted some ways to help users transition between strategies: users will use what they know to solve new problems; and provide users with operations that they can perform, perhaps teaching them new operations (Fu & Gray, 2004). Users also prefer strategies that reduce memory load, even if this makes the process slower (Gray, Sims, Fu, & Schoelles, 2006). If your system supports multiple strategies, you should consider supporting user transitions between them.

[1] Dan Russell noted in a talk in June 2011 that 30 years later many users still do not know how to search within a document, particularly when looking at documents in browsers.

2.2 Theories of Learning

Theories that describe the general process of learning have been developed in several areas, including behavioral psychology (Fitts, 1964), cognitive psychology (Anderson, 1982), cognitive engineering (Rasmussen, 1983), and computational modeling of learning (Newell, 1992; Rosenbloom & Newell, 1987). These theories all propose that learning occurs in a number of stages. We will focus our discussion mostly on Rasmussen's approach (see Figure 3), whilst noting the parallels with the theories of both Fitts and Anderson.

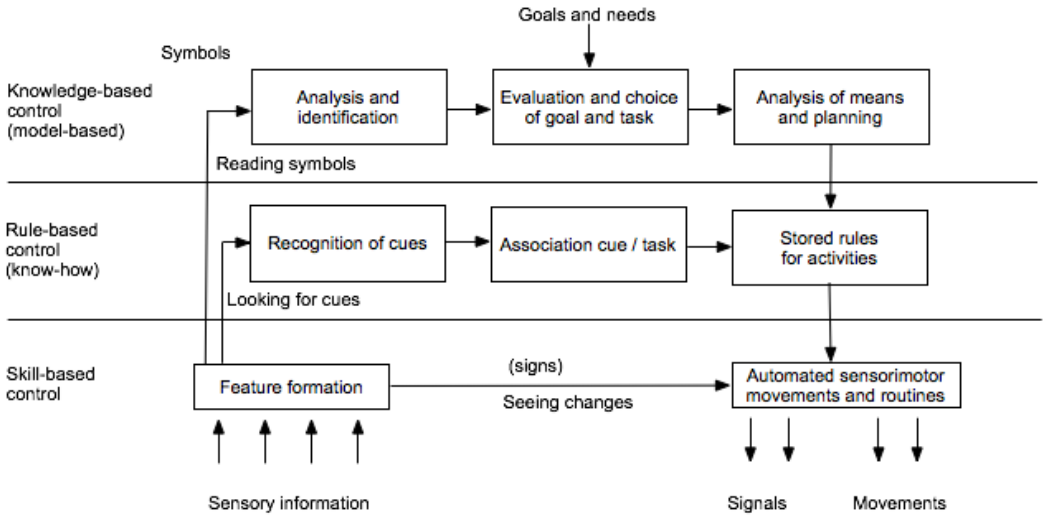


Figure 3. A graphical representation of Rasmussen's theory of knowledge levels (redrawn from Rasmussen, 1983).

2.2.1 Stage 1: Declarative knowledge acquisition

The learner first acquires declarative knowledge (facts) about the domain. Problem solving is difficult at this stage. The domain concepts may not always appear connected to each other. The learning might be considered as learning how to learn more about the task. What is learned at this point, such as the initial problem solving state and the actions—both physical and mental–that are available for problem solving, may not be enough to complete the task. Problem solving, when it is possible, is effortful, and not always correct. The user might not have much confidence in their performance. Rasmussen describes behavior at this level as knowledge-based; Anderson calls this the cognitive stage.

When learners are at this first stage, behavior occurs at the most fundamental level. There are no direct rules to inform the learner what to do next, so they usually have to work from first principles, using strategies such as trial and error. Expert users can find themselves at this level in emergencies or novel situations, when deliberate thought and reasoning about the state of the situation or system are required, based on the user's knowledge (i.e., mental model) of the system. Getting your car out of a deep rut or navigating in an unfamiliar town are examples from driving. In computer systems,

installing software that does not want to install, and changing UNIX permissions on shared files would be other examples.

The learning of this declarative information is modeled in the ACT-R cognitive architecture (and in several other architectures following ACT-R, see Bryne's chapter, this book) through the base-level activation of a declarative memory item (a chunk). The activation or strength of a memory is dependent on how often (frequency) and how recently (recency) the item is used. Whenever an item in memory is used, the base-level activation increases and then decreases in accordance with a power function of the time since use. When the sum of the activations is above the retrieval threshold, a memory item can be retrieved. When it is below, it cannot. Higher activations lead to faster and more accurate retrievals. Thus, the base-level learning mechanism supports the Power law of learning and the Power law of forgetting (Anderson, Fincham, & Douglass, 1999). Equation 3 shows that the base-level activation B, for chunk *i,* where β is a constant, n is the number of presentations for a chunk i, t_j is the time since the j^{th} presentation, and d indicates the decay parameter.

$$B_i = \beta + \ln(\sum_{j=1}^{n} t_j^{-d}) \qquad \textbf{Equation 3}$$

2.2.2 Stage 2: Proceduralization

With learning and practice, the user progresses to Rasmussen's rule-based behavior level (what Anderson calls the associative stage). In this stage the learner can solve problems more routinely and with less effort by following a sequence of steps, or procedure. The declarative knowledge has been compiled into procedures that are relevant to the task domain and can be performed directly. Users may often be able to recognize what needs to be done and apply the appropriate procedure. For users of complex systems, behavior at this level is a consciously controlled activity and is based on familiar rules, either dictated or acquired. Lane changing in a car would be an example, as would docking a ship in an infrequently visited port. Use of office application software similarly often requires rule-based behavior to insert a figure into a document or format a paragraph.

Soar includes a chunking mechanism to describe this learning effect (Newell, 1990; Rosenbloom, 1986), and ACT-R has used several mechanisms to generate this type of learning (e.g., Jones, Ritter, & Wood, 2000; Taatgen & Lee, 2003). These mechanisms do not proceduralize the whole task, but learn based on subtask frequency and declarative memory strength (in the case of ACT-R).

2.2.3 Stage 3: Skill tuning

The final stage, where the knowledge that is applied is refined gives rise to skill-based behavior (Rasmussen); Anderson calls this the autonomous stage. At this point users still get faster at the task, but the improvements get diminishingly smaller. New declarative information is rarely learned, but small adjustments to rule priorities happen. At this point users are generally considered to be experts. Behavior is much more automatic, and occurs when users perform routine functions in their normal

operating environment. Much of their behavior is no longer consciously controlled, and hence cannot be verbalized because the behavior is unavailable to conscious thought. Users perform the task faster and, because little or no conscious control is required, this frees up attentional resources that can be allocated to other tasks. Simple lane following by an experienced motorist would be an example. Pressing a computer on button or deleting text in a word processor might be examples in HCI.

Tuning does not happen in Soar, but in ACT-R tuning could happen where declarative memories are strengthened. In other architectures, tuning would occur where the rules or declarative memory elements are further strengthened.

2.2.4 Moving between the stages

The improvements in performance time do not appear to delineate the stages of learning noted above, except in cases where individual data are studied and strategies are known to take different times (e.g., Delaney, Reder, Staszewski, & Ritter, 1998). This may be because the first stage of learning, where performance might be quite difficult, has to be nearly complete before the task can even be performed. There are hints of this in Figure 1 (b), where the initial slope is perhaps more shallow than would be expected. In complex tasks, the transition through the rule learning and rule tuning stages might lead to steady improvement, and be masked by the large number of rules and sub-tasks being improved. Looking at individual users working on well measured tasks may allow stages to be seen. When this has been done, strategy changes can be readily identified (Delaney, Reder, Staszewski, & Ritter, 1998).

2.2.5 The stages in practice

These three stages of learning have been noticed in several areas of formal reasoning. Formal reasoning is problem solving with known goals (like solve for x), using a closed set of equations or rules that can transform representations (like adding 2 to each side of an equation). Examples include physics problem solving, where the link between problem solving and learning has been studied (Larkin, 1981; Larkin, McDermott, Simon, & Simon, 1980a, 1980b; Ritter, Jones, & Baxter, 1998), proofs in geometry, and solving algebra problems.

In these types of problems, the problem solvers (and computational models) start with domain knowledge and the goals they are trying to achieve. Once novices have learned declarative knowledge, they can then work backward from the goal. In the case of physics problem solving, the task is to derive a value of a variable (like *final speed*), given some known variables (like *mass, initial speed*, and *force*), and some equations (like *force* = *mass* × *acceleration*). Novices tend to work back from the quantity they are trying to find (*final speed*), chaining inference rules together until they find known variables. If they need to compute the *final speed*, they look for an equation that contains *final speed*. Then they look for more equations to find the variables in the first equation, until they bottom out with known variables. This is known as backward chaining, or bottom-up reasoning. After each successful application, a new operation is learned that supports applying that formula to find a variable. After enough learning, experts start with the unknowns, find or recall equations that apply, and work towards the goal in a forward-chaining or top-down way.

Similar learning strategies will be seen in computer repair and troubleshooting, software usage, and other domains where formal reasoning can be applied. Better interfaces will support the novice by providing the domain knowledge needed to learn the inferences, and will support novices and experts by providing state information to reason from.

Whichever way the user learns, some key phenomena persist:

- The ability to recognize correct / incorrect items comes before the ability to generate correct items.
- Knowledge is not acquired in an all-or-nothing way. Novices go through a stage of fragile knowledge, where sometimes the correct knowledge is used, and sometimes not.
- Experts acquire a rich repertoire of representations of their knowledge. Experts not only know more than novices, their knowledge is much better organized and more readily available.

2.3 Alternative Views on Learning

There are several other ways of describing learning. One common way is to distinguish between declarative and procedural types of learning. Another common way is as implicit and explicit. The distinctions between these classifications are still subject to debate, but they represent useful and interesting differences about users that can inform system design. In addition learning can be described as massed, when it occurs in a single episode, or distributed when it occurs over several episodes, or even as a hybrid training series that includes aspects of both approaches.

2.3.1 Declarative and procedural learning

Declarative learning is learning facts (declarations), such as "the power button is on the keyboard", and "the computer manager's office is 004A". There are at least two types of declarative memories: recognition and recall. Recognition memories are easier to build than recall memories. That is why multiple choice tests seem easier—you just have to recognize the answer (recognition could also be seen as primed recall).

Procedural learning is learning actions, or how to carry out procedures. Typing, using an interface, and playing a computer game are all examples of this. Procedural memories help directly perform the task. In this process, these memories are formed after the declarative representations are available to create them.

These two types of learning are characterized by different regularities. Declarative learning can, by definition, be directly described and reported; procedural memories cannot (Ericsson & Simon, 1993). You cannot directly describe the procedural knowledge you use to ride a bike, although you can accurately report the declarative knowledge you used to generate the procedures (like keep your weight balanced), and what is in your working memory as you perform the task (there is a parked car ahead). You can also watch yourself do a task and attempt to describe what you were paying attention to as you did it. What you think you were paying attention to, however, depends on your mental model of the task and how demanding the task is. This

approach is called introspection, and it represents how people think they think, not necessarily how they think.

There are fundamental problems with introspecting in this way. While introspection can lead to useful and helpful insights, it does not lead to complete and valid theories of human thinking (Ericsson & Simon, 1993). Mainstream psychology has rejected introspection as an accurate measurement of how people think because it is subject to many biases and does not have access to most mental processes, although one may find introspection a useful inspiration for ideas that can be later validated by other approaches.

The problems with introspection can be illustrated by a study that looked at learning key bindings in a text editor (Singley & Anderson, 1989). When key bindings in the editor were changed on the second day of the study, for the first hour or two the users felt much, much slower. They were slower than they were at the end of the first day, but still faster than when they started. They regained their skill level by the end of the second day. This study illustrates three important things about users. The first is that they are always learning. The second is that introspection often leads to incorrect conclusions: they rather disliked the new interface, but adapted more quickly than they thought they did. The third is that the users were able to transfer or relearn all of what they had learned from the old interface to the new interface, at least as measured by task times. Only the key bindings changed; the underlying approach and the command structures did not, so the users were still able to apply most of what they had learned. Another similar example found that users preferred mice over light pens, even though the latter were faster to use (Charness, Holley, Feddon, & Jastrzembski, 2004).

2.3.2 Implicit and explicit learning

Memory can also be separated into implicit and explicit memories. In the implicit/explicit distinction, implicit learning seems to be automatic, is based on practice, is not improved by reflection, and produces knowledge that cannot be verbalized. This bears some similarity to the rule tuning stage. If the rules are created based on a simple theory of the domain, but a more complex theory of the domain exists, then additional learning can occur.

Implicit learning implies that learning occurs without consciousness and without hypothesis testing; people do not know that they are learning as they are learning. Implicit learning can occur when the user is working at the skill-based level, with knowledge in some cases eventually being derived at the knowledge-based level through reflection on one's own behavior. This type of learning is less effortful than explicit learning. It can often lead to, or arise from strategy changes, and is an active area of research in psychology (e.g., French & Cleeremans, 2002).

Explicit learning proceeds with full consciousness in a hypothesis testing way. People know that they are learning as they are doing it. Explicit learning, which is normally associated with the knowledge-based level, takes place in novel or unexpected situations, with reasoning usually being based on first principles. In these situations, learning is typified by the use of strategies such as trial and error, where one possible way of doing the task will be tried, and if it does not produce the desired results, the user notes this, reflects on the reasons for failing to achieve the desired results and uses

this to select an alternative method of performing the task. This type of learning is effortful and requires deliberate attention and control, but the net result is that the knowledge that is produced is more available for other uses and can be verbalized.

These distinctions become important when teaching users how to use an interface. Some information is presented to them as declarative knowledge to be learned (where things are, who other users are, and what the objects are), and some is presented as procedural skills, such as how to do a task or the most timely order to do subtasks. If the declarative information is not available, the learning will be implicit.

2.3.3 Massed and distributed learning

Learning can also be massed, distributed, or done in a hybrid way. Massed learning occurs at a single time, for example, cramming for a test. Distributed learning occurs with breaks in time between the learning episodes. Figure 4 shows how much better distributed learning can be for the same amount of practice for a declarative knowledge task (the plot has learning end for both curves at the same time). This figure is based on the ACT-R declarative memory equation.

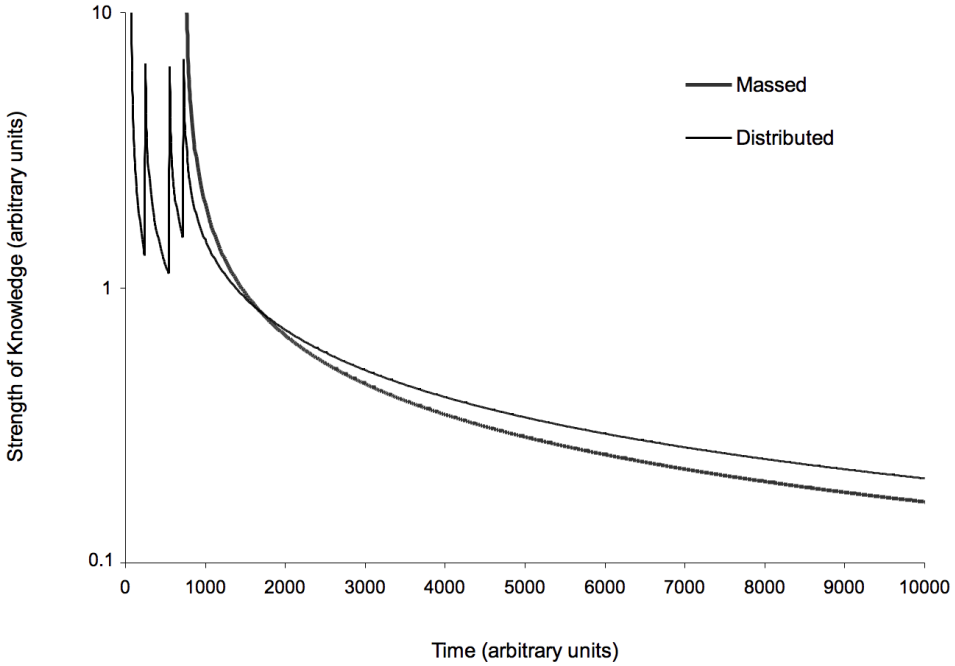


Figure 4. Massed vs. distributed practice in relation to retention of declarative learning. Given exponential decay, which is often assumed for memory, the solid line shows a given amount of practice as one block, and the dashed line shows the same amount spread out over a longer period. The distributed practice has higher activation after time t= 1,730, and will for the remainder of the curve.

Massed learning will give the best retention the next day, but decays more quickly. This may be why it is used by students. If the knowledge will only be used once, then massed may indeed be preferred.

Distributed learning takes less total time (sometimes 1/3 of the time), and the retention is better, sometimes 50% better. Distributed learning also allows more time for learning and may lead to more time spent learning (time on task). Anything you can do to help your users learn in a more distributed fashion will facilitate their retention of declarative information. Some interfaces now put up hints which appear to be a way to support distributed learning.

The theory of learning also has implications for hybrid learning schedules. For example, the ACT-R theory suggests that massing learning of declarative elements might be helpful for creating procedural learning during the second stage (Kim, Ritter, & Koubek, in press). There are not, to our knowledge, any published studies on this, but pilot work is suggestive that this effect will be found.

2.4 Transfer

One of the important issues in learning is transfer of learning. After a skill has been learned, the goal is often to reuse the knowledge or apply it to a new situation. If no transfer occurred, every situation would be a new situation; if transfer was perfect, few situations would be novel and learning would not be necessary. There are both positive and negative examples of transfer: in some cases transfer does not appear to happen very effectively; in other cases, it can be quite effective.

A telling study on transfer was done by Gick and Holyoak (1980) based on work by Duncker (1945), where, before solving a problem, subjects read about an analogous problem. The story (analogous problem) was about how a king wished to invade a town, but his army was too big to come through a single gate in the town, so he split his troops into smaller parties and entered the town through several gates. Subjects then read about the use of rays to attack cancer: the problem was that the beam of rays was too intense to be directed at the cancer cells without seriously damaging the surrounding tissue. A surprising number did not think to use multiple beams of rays, which is what the transfer of knowledge would suggest. This lack of obvious (to the experimenter) transfer effect has been repeated many times. Thus, users do not always transfer useful information unless the information is very well practiced, the transfer is very straightforward, and they have time to think.

A counterexample, however, is provided by the previously noted study on the use of perverse Emacs (Singley & Anderson, 1989). They found that transfer can occur, but users do not always see it, and they do not believe that learning has been transferred when transfer leads to small mistakes (as would have happened often in the first hour for the users of perverse Emacs). Thus, changing key bindings is likely to lead to frustration, but might not hurt users in the long run because the deeper knowledge would transfer.

When complex concepts and tasks need to be learned, they may be introduced using simple analogies: the way muscle fibers work, for example, can be likened to the way that a rowing crew operates (e.g., Spiro, Feltovich, Coulson, & Anderson, 1989). The idea is that learning can be transferred between the simpler concept or task, and the complex concept or task. Whilst this can be a useful approach initially, the simple analogies can lead to misconceptions if they are inappropriately applied to situations

where they do not properly reflect the real situation. To overcome these limitations, multiple analogies need to be used, an approach that is advocated by cognitive flexibility theory (e.g., see Feltovich, Spiro, & Coulson, 1993). So for the muscle fiber example, you also have to introduce analogies that overcome the limitations of the other analogies, and end up with a set of five analogies that together help to capture the full complexity of how muscle fibers really work.

There are tools available to measure potential transfer. One example is GOMS (Bovair, Kieras, & Polson, 1990; Kieras, 1998), a simple language that describes how to do a task. Each instruction in the language, such as how to push a button, takes about 17 ~ 30 s to learn, while declarative pieces of knowledge take about 6 ~ 10 s. Thus, the amount of differences between how to do two tasks indicates how much knowledge can be transferred and how much must be learned. GOMS analyses of popular operating systems suggest that there are real differences between them, with one being a subset of the other, and one thus being easier to learn (Kieras, 1998).

2.5 Expertise and Experts

With extended practice users become experts at their task. Generally, to become world class, it takes about 10 years of practice as well as some deliberate reflection and declarative learning, either from extensive self-tutoring or as a coach (Ericsson, 1996; Hayes, 1981, ch. 10; Schneider, 1985; Simonton, 1996) (also, Boot & Ericsson, this volume). There are exceptions, but most authors argue that these are rare or nonexistent. Less time is required for local or national prominence where there are existing practice. Whilst deliberate practice is a usual and necessary part of the development of expertise it is not, in and of itself, a sufficient condition (Ericsson, 1996). There are several factors that contribute to expertise—experiential, social, cognitive and performance-related—so variation in practice conditions is also needed, for example, to ensure that the expertise reaches a level where the user can reliably make accurate judgments, perform at a high level, and deal with rare or unanticipated situations (Hoffman, Shadbolt, Burton, & Klein, 1995).

Deliberate, intensive practice with a tutoring system can provide an usually large amount of time on the critical aspects of a task, where field users have to spend additional time to see the critical aspects, learning can be greatly accelerated. For example, the Sherlock tutor (Lesgold, 2001) abstracts the physical movements to learn a diagnostic reasoning task. Students using the tutor for approximately 25 hours learn as much as others learn after 4 years of practice in the field. Learners in the field have other tasks, but also the steps in the diagnostic sequence can take hours to perform.

There are some theories that suggest that with practice the user gains more than just speed in the task. In some studies they appear to have greater memory and more attention that they can pay to the task (Ericsson & Kintsch, 1995). In nearly all cases they have more accurate perception for and of the task details. In all cases, they have more knowledge and better anticipation of what will happen in the task.

2.6 Skilled Behavior, Users in Complex Environments

Human performance is often a complex mixture of behavior at different levels. In most cases, routine users will exhibit open-loop behavior moving seamlessly between the rule-based and skilled levels. That is, they will be able to perform most tasks in a routine way using existing, well-used knowledge without checking all of their steps. They will not close the loop by checking that things worked correctly because in most cases they no longer need to. If they do make errors, they will be able to recognize and correct them quickly, often before any adverse consequences occur. They will continue to get faster with practice, but the improvement will be minor. These skills will be easier to compose with other tasks. There will also be some closed-loop behaviors for tasks that are less well practiced, and on these tasks users will be more careful and check their work.

Rasmussen (1983) argues that good design needs to support all three levels of behavior, not just one. Aircraft pilots, for example, often operate at all three levels. Some aspects of their behavior are automatic, for others they refer to rules and procedures, whilst for others, particularly in emergencies, they reason on the basis of their knowledge about the plane and learn. For most emergencies, they will use the quick reference handbook (QRH), which provides them with compiled knowledge in the form of checklists. In unusual emergencies, however, which are not documented in the QRH, they have to reason from first principles.

We can also note a social human factor here: if there are multiple team members, they may operate at different levels at the same time due to differences in experience. This may give rise to conflict or consensus, depending on how these differences are resolved. In an emergency situation in an aircraft, for example, the more experienced pilot (the Captain) may be working at one level of reasoning, while the more junior pilot (the First Officer) may be using a different level that may not be appropriate to the unusual situation (or vice versa). Crew Resource Management (Wiener, Kranki, & Helmreich, 1993) is an approach to reduce this problem.

The SwissAir crash at Halifax, Nova Scotia in 1998 illustrates the point about pilots operating at different levels (e.g., see Carley, 21 January 1999; Dekker, 2005). After a burning smell was detected, there were reports of smoke in the cockpit. The first officer, who was flying the aircraft, suggested that they should immediately start jettisoning fuel to make the aircraft light enough to descend and land at the earliest opportunity. This instinctive behavior is typical of the skill-based level. The captain, however, decided that they should first follow the predefined procedures for dealing with smoke and fire. He was reasoning at the rule-based level. The decision about dumping fuel was delayed. Meanwhile the fire continued to develop, and the aircraft became uncontrollable, and eventually crashed into the sea, killing all 229 people on board.

Example approaches to creating computational models of these situations may help resolve the effects of different factors. These models are reviewed by Byrne in his chapter (this volume). In addition to modeling the learning, it is also important that the models interact with the environment, which is a non-trivial problem (Ritter, Baxter, Jones, & Young, 2000).

Learning in these situations not only occurs on the individual level but also on the organizational level. Hummerdal, Wilhelmsson, and Dekker's chapter in this volume explores some of these issues.

2.7 Summary

There are several theories of the process of learning (and we note only the most cognitive ones here, Anderson 2000, reviews further theories). Many theories propose that there are three stages: acquisition, proceduralization, and then tuning. Deliberate practice helps move between these stages, and learning new strategies will complicate the description but also help learners perform better.

Learning can also be viewed with respect to the type of knowledge learned, declarative or procedural, and explicit and implicit (and these distinctions are likely to be related).

There are different learning schedules. These give rise to different forgetting curves (covered in the next section). In many cases distributed learning is to be preferred, but there are situations where massed learning may be better. It is possible that hybrid learning schedules will be explored and found to be useful.

How knowledge transfers determines where learners start and how their learning transfers to new situations. How learners gauge this transfer has not been much studied, but probably is not accurate.

With deliberate practice learners can change strategies and become exposed to a wide range of situations. In complex environments, deliberate practice becomes more important because the stakes are higher and because rare but important situations can be difficult to encounter without practice.

Taken together, these effects and issues suggest that learning is rarely complete. It is easy to think that if you have learnt to operate a system your learning about that system is complete. This need not be so; many users can sometimes perform very competently with very little knowledge. These sorts of systems lie at one end of the spectrum of performance: the systems have a simple structure, or they only exhibit a small range of behaviors. To operate these systems successfully, the user may only need a small amount of knowledge and a small repertoire of actions. At the other end of the spectrum lie complex systems, such as computer operating systems, large scale IT systems, vehicles, and so on. The user may not be able to rely on a structural mental model of the system to help them control it, and the range of system behaviors is so large that the user may never see all of them. So even though the users may be competent in operating the system, particularly under routine conditions (where the users' behavior may become largely proceduralized), they continue to learn new ways to operate them as they encounter new conditions and new behaviors.

3. Retention

Pilots have to go back to training school on a flight simulator every several months, and they have to fly at regular intervals to maintain their accreditation to fly particular types of aircraft. Many professionals are in this situation. In this section, we discuss this

maintenance, or retention of knowledge and skills. Schmidt and Bjork (1992) argue that learning is an imperfect indicator of later performance, and that learning and retention should consequently be considered together. It is therefore important to understand how acquired knowledge and skills are retained and should be maintained in the longer term.

3.1 The Process of Retention

Earlier, we discussed the process of learning—there is a consensus understanding of how learning occurs in three different stages—which is supported theoretically by the ACT-R cognitive architecture. Based on this theoretical foundation, we can describe how forgetting and retention might vary reliably across those corresponding stages of learning. Figure 5 provides a summary of learning and retention accounts.

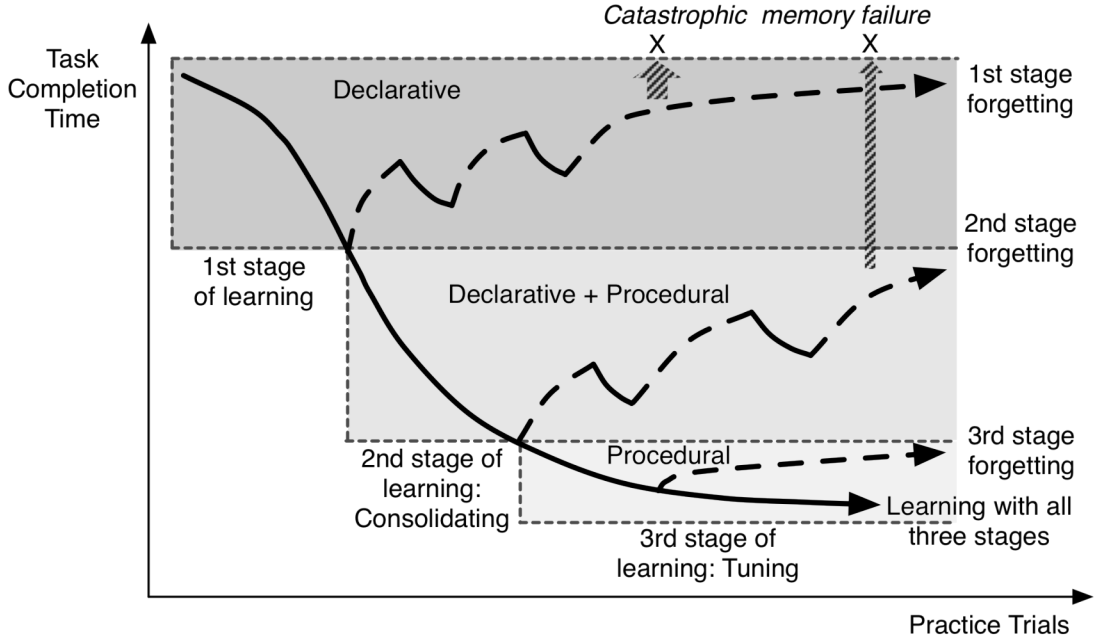


Figure 5. A theory of skill retention, showing the three stages of learning and forgetting. Rasmussen called these stages knowledge-based, rule-based, and skill-based. Fitts called these stages cognitive, associative, and autonomous. (Figure taken from Kim, Ritter, & Koubek, in press).

The First Stage: Declarative. For this first stage of learning and forgetting, knowledge in declarative memory degrades with lack of use, perhaps catastrophically as indicated by X's in Figure 5, leading to the inability to perform the task. In this stage, learning and forgetting are accounted for by the activation mechanism for declarative knowledge in ACT-R: with lack of use, the strength of declarative memory items declines. Decreased memory strength leads to increased response times and decreased retention and performance accuracy.

The Second Stage: Associative. In the second stage of learning, task knowledge is represented using a mix of declarative and procedural memory. With lack of use, the declarative knowledge is forgotten, leading to mistakes and missed steps. Procedural

memory, on the other hand, is basically immune to decay. The slope of the forgetting curve in this stage could vary by subtask because it would vary in their knowledge mix, and different mixes would decay at different rates. In the first two stages, catastrophic memory failure can occur because the declarative knowledge is not fully activated. This finding suggests that in this mixed stage training is necessary to keep the declarative knowledge active and also to support further proceduralization because the activation of declarative memory is required to generate procedural rules.

The Third Stage: Procedural. In the third stage of learning, task knowledge is available in both declarative and procedural forms, but procedural knowledge predominantly drives performance. Practice will compile declarative knowledge into procedural knowledge. We describe this type of task knowledge as a *proceduralized skill*. Major cognitive theories predict that procedural knowledge will not decay (reviewed in Kim et al., in press), and reviews of forgetting (e.g., Rubin & Wenzel, 1996) typically examine declarative memory. With lack of use, declarative knowledge may degrade. Nevertheless, the learner can still perform the task if all the knowledge is proceduralized and thus not forgotten with time, and performing the task does not require new declarative inputs.

3.2 Design Rationale in the Three Stages

Learners in the three stages will vary with respect to how forgetting affects their performance. Information access costs (e.g., measured by accuracy or time lag) will vary across those stages. Gray et al. (2006) observed a progressive switch from more interaction-intensive to more memory-intensive strategies as information access costs increased in their experiments.

More specifically, a tradeoff between interaction-intensive and memory intensive strategies in a system design exists. It depends on factors such as the time required to encode task knowledge in memory, the time required to retrieve the knowledge from memory, and the probability of retrieving the encoded task knowledge in memory (if not forgotten).

A recent study (Kim, 2008) compared performance (i.e., task completion time) of users in different interfaces (a keystroke-driven and a menu-driven interface). In this study, users completed several learning sessions and their forgetting and relearning performance with varying retention intervals were measured. One of the interesting implications is that a menu-driven interface can help the user to relearn how to perform a task. That is, in our everyday interaction, a menu-driven interface can allow the user not only to perform the task as a novice but also to complete a task even after an arbitrarily long time lag of knowledge disuse. This indication can help to explain why menu-driven interfaces (i.e., graphical user interfaces) are prevalent in interactive system designs; it is not just that they are easier to learn, it may be that they are easier to relearn.

4. Conclusion: Implications for System Design

The theories and results noted here have several implications for system design. These implications apply to many levels of system design, including direct interface design (see Byrne, this volume), training materials, and system level applications.

4.1 Learning and Design

All users will learn and hence get faster at doing a task with repeated practice. This is an important way that users fit themselves to a task, covering up errors in design and compensating for trade-offs made in interface design that do not favor the user. Interfaces that initially appear too slow may become faster with practice, but may require training (to learn the task) and practice (to get faster). These times can be roughly predicted by comparison with other tasks or after several initial trials to get the slope of the improvement. There are theories that predict this, based on a simple task analysis (Kieras & Bovair, 1986) where a step (rule) takes about 30 s to learn. Work continues on more complete theories to predict how long it takes to learn a task, both from a design perspective (Christou, Ritter, & Jacob, 2009; Bovair et al., 1990) and from a cognitive architecture perspective (Anderson, 2007; Anderson et al., 1999; Ritter & Bibby, 2008).

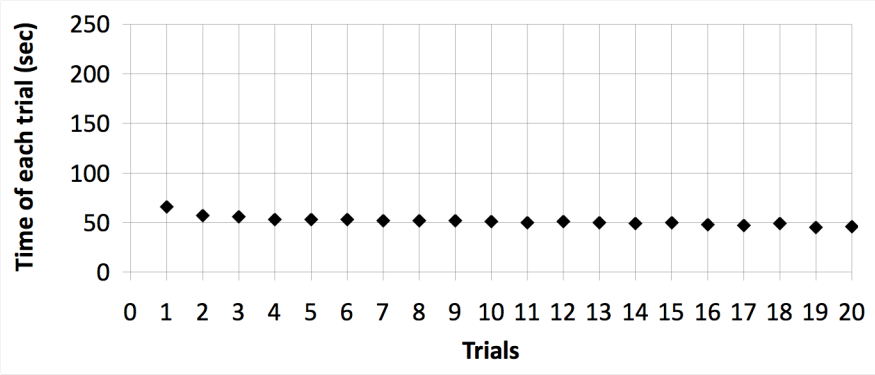
It is important to consider the different levels of behavior when designing systems. Systems need to support users learning the basic skills at the knowledge level and as they progress to rule-based and skill-based behavior. The support needs to include the provision of appropriate information that will help them operate on the rule-based and skill-based levels.

Users must have some declarative information about a task before they can perform it. This means that interfaces and their associated system, including manuals, online help, web sites, and other users, should provide new users with enough information to perform the task, or at least to start to explore the environment.

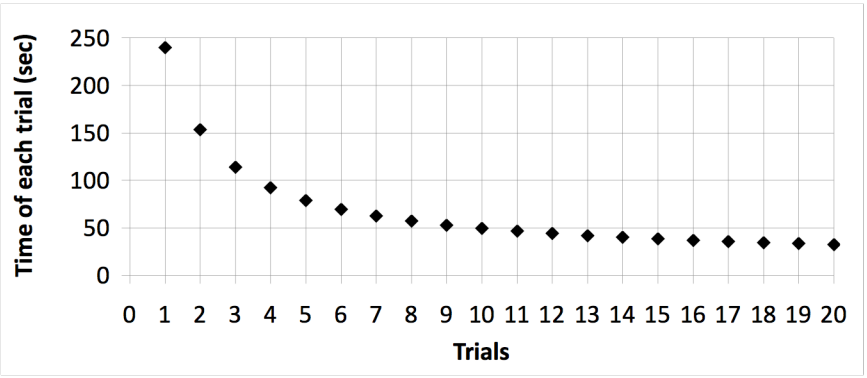
Both procedural and declarative memory usage depend on how the memories are represented. If the user has a more general representation and declarative knowledge associated with it, the procedural knowledge can transfer to new problems more easily. Interfaces and systems (including instructional materials) that support more general representations and more task-oriented representations (vs. tool specific representations) will allow users to transfer their knowledge both in to and out of the interface more readily.

The learning curve that can be created by observing learners can directly provide insights to system design. The start of the curve shows how difficult the interface appears to novices. This amount of time might be too high or at an acceptable level. Figure 6 (a) shows a relatively shallow curve where performance times for novices are not so different than for experts. Figure 6 (b) shows a curve where novices would be less happy but experts happier, which might be appropriate for a game or an often used interface.

The slope of the curve can also provide insights. If the curve is very steep, and the system will be used often, the initial task times might not be an issue. If the interface is used only a few times, then the slope needs to get the user's task time to an acceptable level more quickly or start out at a reasonable time. It can also be the case that 150 s is an acceptable time, or it can be that 50 s is not, and even the well learned portion of the curve is not yet fast enough—it will depend on the context. For an example of a task with thresholds, see Bryan and Harter (1897), who studied telegraph operators. There were minimum speeds to send and receive with telegraphs 'on the main line'.



(a) A relative shallow learning curve.



(b) A relatively steep learning curve.

Figure 6. Two learning curves with approximately the same final speed: (a) a shallow learning curve and (b) a steeper and longer curve. A relatively shallow curve but with a low intercept may be more appropriate for interfaces not used often, while a relatively steep curve but with a lower final time may be more appropriate for interfaces used by experts.

Some users like to learn. This is probably most often seen in games, which have their explicit goal to teach the user some procedural skill (like driving a racing car), or some declarative knowledge (like the way around a dungeon). Many other users prefer not to learn, but will if the task is important enough. This is likely related to a trait called need for cognition (Cacioppo & Petty, 1982), which notes that some people like to think and learn more than others. When users want to learn it may be appropriate to give them more to learn.

Systems that allow users to recognize actions they want to do (e.g., by using menus) will be easier initially to use than those that require them to recall keystroke based commands. There is a tradeoff, however, as novices become experts. The experts will be able to recall the keystrokes or command names and may not wish to wade through the choices. For example, Linux experts like to use keystrokes in the command shell, while novices on Windows prefer a graphical user interface. The transition from novice to expert involves changes in strategy, as described earlier in Section 2.1. When designing systems, it is important that the changes in strategy required to move from novice to expert are not hindered by the system, and are preferably actively supported and facilitated.

Systems that encourage users to reflect on how to perform the task may lead to different types of learning (Boot & Ericsson, this volume). Where the learning is incidental and not tied to educational intent or work systems, this may not matter. On the other hand, tutoring systems should be designed carefully so that learners do not have to learn unnecessary information (Sweller, 1988) and what users learn is what is intended. For example, the interface should not encourage users to just try key presses or button clicks to see what happens, which can lead to implicit learning that is tied to the representation of the interface and not to the domain.

4.2 Retention and System Design

In various systems (or interfaces), the process of learning occurs as users interact with those systems. Wherever learning is considered to be important, retention should be also taken into consideration. For example, training systems are an important domain in which improving the user's performance and skill retention are the goal of the system.

We can analyze a training system in terms of the aforementioned three stages of learning and retention. This approach provides a way to evaluate the effectiveness of the training system. We present potential principles for a cognitively engineered training system.

During the first stage of learning and retention, the trainee's knowledge in declarative memory would degrade with lack of use, leading to the inability to perform the task. With lack of use, the strength of declarative memory items declines, following an activation formula (previously shown in Equation 3). A cognitively engineered training system can optimize the scheduling of necessary training (e.g., Atkinson, 1972; Pavlik, 2007), helping the trainee to achieve long-term retention by helping them focus their learning on the least learned material.

In the second stage of learning, training is needed to avoid catastrophic failures in task performance resulting from the learner's inability to retrieve declarative knowledge for steps still requiring declarative knowledge elements. This is either because the task is incompletely proceduralized or because declarative inputs vary for the task (e.g., the radio frequencies or flight numbers used by pilots as they enter or exit controlled airspace). The higher decay rates associated with declarative memories also suggest that achieving more complete expertise entails training not only common but also uncommon tasks to the procedural level (e.g., emergency procedures for pilots).

Furthermore, while the periods of time between retraining sessions can increase with practice, catastrophic failures can still occur if completing the task requires declarative memory elements (i.e., changing inputs like frequencies, coordinates, or names). As an example, in learning typing skills, this stage would correspond with the learner possessing a strong declarative representation of the key locations while also possessing some procedural rules regarding the task as retrievals increasingly get combined with key presses (in other words the greater association of conditions with actions).

The third stage of learning and retention would correspond to the knowledge about how to type to being fully represented in production rules, and the declarative knowledge would not get used any more and would decay. Infrequently used characters (e.g., # or ^), however, might still require declarative retrievals if these characters are insufficiently practiced.

Conversely, less practiced or infrequently used skills, such as responding to unusual errors, may exhibit the mixed curve shown in Figure 5. Consequently, these skills require concerted and structured practice to proceduralize. Unlike the common ASCII keys in the typing task, there is no assurance that routine task execution will compile and proceduralize the declarative memory elements associated with the task, meaning that training is most likely necessary to achieve proceduralization (noted in Figure 5 as crossing the dashed grids that represent the stage thresholds).

4.3 Learning and Higher Order Cognition

Learning also can be used to help with higher order cognition constructs such as attention and situation awareness (SA, also see chapter in this handbook by Endsley). With greater learning people begin to recognize situations and generate responses faster. This allows them to allocate spare attentional resources to a complex task or to a secondary task.

Learning has a secondary effect on SA by supporting many of the activities related to SA and generally improving SA. If we take Endsley's (1988, also noted in the chapter in this volume) definition of SA as "the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future" we can see that learning makes the process faster and more accurate. As novices learn about the task domain, they learn where to look to find the information they need, they learn how the various elements in the domain are related so they can process them together, and they learn how to make more accurate predictions about the effects of their actions because of their increased knowledge. As the stages of perceive, decide, and act become more proceduralized, they require less attentional resources. The net effect is that more resources are available for monitoring other aspects of the situation, and less working memory decay occurs while performing secondary tasks.

4.4 Summary

Learning of knowledge and skills and subsequent retention of these skills are fundamental aspects of users. There are many known regularities in learning and retention. The ones reviewed here provide a brief introduction. There are more topics

and factors in learning and retention that are worth understanding, including chunking, compilation of knowledge, conscious control, expertise, learning from reflection and self-explanation, scaffolding, and learning from errors. Some of these topics are covered in other chapters in this handbook, including Expertise (Boot & Ericsson), and learning from failure (Hummerdal et al.). All of these regularities have implications for design that are both feasible and powerful.

4.5 Further Outside Readings

We can recommend the following further readings in learning and retention in addition to the cited works.

Anderson, J. R. (2000). *Learning and memory*. New York, NY: John Wiley and Sons.

Anderson (2000) provides an overview of multiple learning theories and sources of data on learning.

Jonassen, D. H., & Grabowski, B. L. (1993). *Handbook of individual differences, learning, and instruction*. Hillsdale, NJ: Erlbaum.

Jonassen and Grabowski (1993) has several chapters discussing the application of learning theory to education and explains many related topics.

Reigeluth, C. M. (2007). Order, first step to mastery: An introduction to sequencing in instructional design. In F. E. Ritter, J. Nerb, T. M. O'Shea & E. Lehtinen (Eds.), *In order to learn: How the sequences of topics affect learning* (pp. 19-40). New York, NY: Oxford.

Reigeluth's (2007) chapter introduces many basic ideas in instructional design, and highlights the strengths and weaknesses of several commonly used ways to order instruction.

4.6 Future Directions

We conclude this chapter noting several questions representing future directions for the field, difficult problems to be solved, and topics that remain to be addressed.

(a) Users often use an inefficient strategy (Fu & Gray, 2004; Rieman, 1996). How do you get users not just to learn how to perform an existing strategy faster, but to change their strategy, to learn a new, better strategy, when one is available? This is also related to deliberate practice (Boot & Ericsson, this volume).

(b) Can we create a model learner that tests interfaces? Fu (this volume) and other researchers (e.g., Kieras, 2003; Ritter, Baxter, Jones, & Young, 2000)(Byrne, this volume) have worked on this problem for years. Related to this is how to build useful tools to help predict learning (Jastrzembski, Addis, Krusmark, & Gluck, 2010; John & Jastrzembski, 2010).

(c) Current computational theories (reviewed in Kim et al., in press) often do not predict forgetting procedural skills (but some data suggests that procedural skills might decay, Anderson et al., 1999). It would be interesting to gather data on forgetting skills that had been proceduralised. Also, adding this theory then into tutors, human, systems (e.g., educational systems), and artificial systems, would be possible.

(d) How do people learn to operate large scale systems of systems? Do existing methods of training scale up? We already know that people use functional mental models more than structural mental models because the structure of systems is getting more and more complicated (too complicated to keep in your head). We are already seeing people talk about building systems that are so complex that we do not really understand them (Pew & Mavor, 2007). The old methods of software engineering do not seem to scale up for building these systems, so do we also need to think about new ways of learning to operate these systems?

References

Anderson, J. R. (1982). Acquisition of cognitive skill. *Psychological Review, 89*, 369-406.

Anderson, J. R. (1995). *Learning and memory*. New York, NY: John Wiley and Sons.

Anderson, J. R. (2000). *Learning and memory: An integrated approach*. New York, NY: John Wiley and Sons.

Anderson, J. R. (2007). *How can the human mind exist in the physical universe?* New York, NY: Oxford University Press.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Anderson, J. R., Fincham, J. M., & Douglass, S. (1999). Practice and retention: A unified analysis. *Journal of Experimental Psychology: Learning, Memory, and Cognition, 25*(5), 1120-1136.

Anderson, J. R., Greeno, J. G., Kline, P. J., & Neves, D. M. (1981). Acquisition of problem-solving skill. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition*. Hillsdale, NJ: Erlbaum.

Atkinson, R. C. (1972). Optimizing the learning of second-language vocabulary. *Journal of Experimental Psychology, 96*, 124-129.

Bovair, S., Kieras, D. E., & Polson, P. G. (1990). The acquisition and performance of text-editing skill: A cognitive complexity analysis. *Human-Computer Interaction, 5*, 1-48.

Brown, S., & Heathcote, A. (2003). Averaging learning curves across and within participants. *Behavior Research Methods, Instruments and Computers, 35*, 11-21.

Bryan, W. L., & Harter, N. (1897). Studies in the physiology and psychology of the telegraphic language. *Psychological Review, 4*, 27-53.

Cacioppo, J. T., & Petty, R. E. (1982). The need for cognition. *Journal of Personality and Social Psychology 42*(1), 116-131.

Card, S. K., Moran, T., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum.

Carley, W. M. (1999, January 21). Airlines: Swissair pilots differed on how to avoid crash. *Wall Street Journal,* B1.

Charness, N., Holley, P., Feddon, J., & Jastrzembski, T. (2004). Light pen use and practice minimize age and hand performance differences in pointing tasks. *Human Factors, 46*(3), 373-384.

Christou, G., Ritter, F. E., & Jacob, R. J. K. (2009). Knowledge-based usability evaluation for reality-based interaction. In G. Christou, E. L.-C. Law, W. Green & K. Hornbaek (Eds.), *Challenges in the evaluation of usability and user experience in reality-based interaction (workshop proceedings). At CHI 2009 Conference on Human Factors in Computing Systems, Boston, MA, 2009. CHI 2009 Workshop: Challenges in Evaluating Usability and User Experience in Reality Based Interaction* (pp. 36-39). Toulouse, France: IRIT Press.

Coram, R. (2002). *Boyd: The fighter pilot who changed the art of war*. New York, NY: Back Bay Books/Little, Brown and Company.

Crossman, E. R. F. W. (1959). A theory of the acquisition of speed-skill. *Ergonomics, 2*, 153-166.

Dekker, S. (2005). *Ten questions about human error: A new view of human factors and system safety*. Mahwah, NJ: Lawrence Erlbaum.

Delaney, P. F., Reder, L. M., Staszewski, J. J., & Ritter, F. E. (1998). The strategy specific nature of improvement: The power law applies by strategy within task. *Psychological Science, 9*(1), 1-8.

Duncker, K. (1945). On problem solving. *Psychological Monographs, 58*(5), (Whole No. 270).

Endsley, M. (1988). Design and evaluation for situation awareness enhancement. In *Proceedings of the Human Factors Society 32nd Annual Meeting*, 97-101. Human Factors Society: Santa Monica, CA.

Ericsson, K. A. (Ed.). (1996). *The Road to excellence: The acquisition of expert performance in the arts and sciences*. Mahwah, NJ: Erlbaum.

Ericsson, K. A., Charness, N., Feltovich, P. J., & Hoffman, R. R. (Eds.). (2006). *The Cambridge handbook of expertise and expert performance*. Cambridge: Cambridge University Press.

Ericsson, K. A., & Kintsch, W. (1995). Long-term working memory. *Psychological Review, 102*, 211-245.

Ericsson, K. A., Krampe, R. T., & Tesch-Roemer, C. (1993). The role of deliberate practice in the acquisition of expert performance. *Psychological Review, 100*, 363-406.

Ericsson, K. A., & Simon, H. A. (1993). *Protocol analysis: Verbal reports as data* (Vol. 2nd ed.). Cambridge, MA: MIT Press.

Feltovich, P. J., Spiro, R. J., & Coulson, R. L. (1993). Learning, teaching, and testing for Complex Conceptual Understanding. In N. Frederiksen, R. J. Mislevy & I. I. Bejar (Eds.), *Test theory for a new generation of tests* (pp. 181-217). Hillsdale, NJ: Erlbaum.

Fitts, P. M. (1964). Perceptual-motor skill learning. In A. W. Melton (Ed.), *Categories of human learning* (Vol. 47, pp. 381-391). New York: Academic Press.

French, R. M., & Cleeremans, A. (2002). *Implicit learning and consciousness: An empirical, philosophical and computational consensus in the making*. Hove, UK: Psychology Press.

Fu, W.-T., & Gray, W. D. (2004). Resolving the paradox of the active user: Stable suboptimal performance in interactive tasks. *Cognitive Science, 28*(6), 901-935.

Gick, M. L., & Holyoak, K. J. . (1980). Analogical problem solving. *Cognitive Psychology, 12*, 306-355.

Gray, W. D., Sims, C. R., Fu, W.-T., & Schoelles, M. J. (2006). The soft constraints hypothesis: A rational analysis approach to resource allocation for interactive behavior. . *Psychological Review, 113*(3), 461-482.

Hayes, J. R. (1981). *The complete problem solver*. Philadelphia: The Franklin Institute Press.

Heathcote, A., Brown, S., & Mewhort, D. J. K. (2000). Repealing the power law: The case for an exponential law of practice. *Psychonomic Bulletin & Review, 7*, 185-207.

Hoffman, R., Shadbolt, N., Burton, A., & Klein, G. (1995). Eliciting knowledge from experts: A methodological analysis. *Organizational Behaviour and Decision Processes, 62*, 129-158.

Jansen, B. J., & Spink, A. H. (2006). How are we searching the World Wide Web? A comparison of nine large search engine transaction logs. *Information Processing and Management, 42*(4), 248-263.

Jastrzembski, T. S., Addis, K., Krusmark, M., & Gluck, K. A. (2010). Prediction intervals for performance prediction. In D. D. Salvucci & G. Gunzelmann (Eds.), *Proceedings of ICCM - 2010- Tenth International Conference on Cognitive Modeling* (pp. 109-114).

John, B. E., & Jastrzembski, T. S. (2010). Exploration of costs and benefits of predictive human performance modeling for design. In *Proceedings of the 10th International Conference on Cognitive Modeling*, 115-120. Drexel University: Philadelphia, PA.

Johnson-Laird, P. N. (1983). *Mental models*. Cambridge: Cambridge University Press.

Jones, G., Ritter, F. E., & Wood, D. J. (2000). Using a cognitive architecture to examine what develops. *Psychological Science, 11*(2), 93-100.

Kieras, D. E. (1998). A guide to GOMS model usability evaluation using NGOMSL. In M. Helander, T. K. Landauer & P. Prabhu (Eds.), *Handbook of human-computer interaction* (Vol. 12, pp. 391-438). Amsterdam: Elsevier.

Kieras, D. E. (2003). Model-based evaluation. In J. Jacko & A. Sears (Eds.), *Handbook for human-computer interaction* (pp. 1139-1151). Mahwah, NJ: Erlbaum.

Kieras, D. E., & Bovair, S. (1986). The acquisition of procedures from text: A production system model. *Journal of Memory and Language, 25*, 507-524.

Kim, J. W. (2008). *Procedural skills: From learning to forgetting*. Department of Industrial and Manufacturing Engineering, The Pennsylvania State University, University Park, PA.

Kim, J. W., Ritter, F. E., & Koubek, R. J. (in press). An integrated theory for improved skill acquisition and retention in the three stages of learning. *Theoretical Issues in Ergonomics Science*.

Kimble, G. A. (1961). *Hilgard and Marquis' Conditioning and Learning* (2nd Edition ed.). New York Appleton-Century-Crofts.

Larkin, J. H. (1981). Enriching formal knowledge: A model for learning to solve textbook physics problems. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition* (pp. 311-334). Hillsdale, NJ: Erlbaum.

Larkin, J. H., McDermott, J., Simon, D. P., & Simon, H. A. (1980a). Expert and novice performance in solving physics problems. *Science, 208*, 1335-1342.

Larkin, J. H., McDermott, J., Simon, D. P., & Simon, H. A. (1980b). Models of competence in solving physics problems. *Cognitive Science, 4*, 317-345.

Lesgold, A. M. (2001). The nature and methods of learning by doing. *American Psychologist*(November), 964-973.

Lieberman, M. B. (1984). The learning curve and pricing in the chemical processing industries. *The RAND Journal of Economics, 15*(2), 213-228.

Lindsay, P. H., & Norman, D. A. (1977). *Human information processing* (Vol. 2nd). San Diego, CA: Harcourt Brace Jovanovich.

Moray, N. (1999). Mental models in theory and practice. In D. Gopher & A. Koriat (Eds.), *Attention and performance XVII: Cognitive regulation of performance: Interaction of theory and application* (pp. 223-258). Cambridge, MA: MIT Press.

Neisser, U. (1976). *Cognition and reality*. San Francisco: W. H. Freeman.

Nerb, J., Krems, J., & Ritter, F. E. (1993). Rule learning and the power law: A computational model and empirical results. In *Proceedings of the Fifteenth Annual Conference of the Cognitive Science Society*, 765-770. Erlbaum: Hillsdale, NJ.

Newell, A. (1990). *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Newell, A. (1992). Unified theories of cognition and the role of Soar. In J. A. Michon & A. Akyurek (Eds.), *Soar: A cognitive architecture in perspective*. Dordrecht, NL: Kluwer Academic Publishers.

Newell, A., & Rosenbloom, P. S. (1981). Mechanisms of skill acquisition and the law of practice. In J. R. Anderson (Ed.), *Cognitive skills and their acquisition* (pp. 1-51). Hillsdale, NJ: Erlbaum.

Ohlsson, S. (1992). The learning curve for writing books: Evidence from Professor Asimov. *Psychological Science, 3*(6), 380-382.

Pavlik, P. I. (2007). Timing is in order: Modeling order effects in the learning of information. In F. E. Ritter, J. Nerb, E. Lehtinen & T. O'Shea (Eds.), *In order to learn: How the sequences of topics affect learning* (pp. 137-150). New York, NY: Oxford University Press.

Pew, R. W., & Mavor, A. S. (Eds.). (2007). *Human-system integration in the system development process: A new look*. Washington, DC: National Academy Press. http://books.nap.edu/catalog.php?record_id=11893.

Rabbit, P., & Banerji, N. (1989). How does very prolonged practice improve decision speed? *Journal of Experimental Psychology: General, 118*, 338-345.

Rasmussen, J. (1983). Skills, rules, knowledge: Signals, signs and symbols and other distinctions in human performance models. *IEEE Transactions: Systems, Man, & Cybernetics, SMC-13*, 257-267.

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

Ritter, F. E., Baxter, G. D., Jones, G., & Young, R. M. (2000). Supporting cognitive models as users. *ACM Transactions on Computer-Human Interaction, 7*(2), 141-173.

Ritter, F. E., & Bibby, P. A. (2008). Modeling how, when, and what learning happens in a diagrammatic reasoning task. *Cognitive Science, 32*, 862-892.

Ritter, F. E., Jones, R. M., & Baxter, G. D. (1998). Reusable models and graphical interfaces: Realising the potential of a unified theory of cognition. In U. Schmid, J. Krems & F. Wysotzki (Eds.), *Mind modeling—A cognitive science approach to reasoning, learning and discovery* (pp. 83-109). Lengerich, Germany: Pabst Scientific Publishing.

Ritter, F. E., & Schooler, L. J. (2001). The learning curve. In W. Kintch, N. Smelser & P. Baltes (Eds.), *International encyclopedia of the social and behavioral sciences* (Vol. 13, pp. 8602-8605). Amsterdam: Pergamon.

Rosenbloom, P. S. (1986). The chunking of goal hierarchies. In J. Laird, P. S. Rosenbloom & A. Newell (Eds.), *Universal subgoaling and chunking: The automatic generation and learning of goal hierarchies*. Boston, MA: Kluwer.

Rosenbloom, P. S., & Newell, A. (1987). Learning by chunking, a production system model of practice. In D. Klahr, P. Langley & R. Neches (Eds.), *Production system models of learning and development* (pp. 221-286). Cambridge, MA: MIT Press.

Rubin, D. C., & Wenzel, A. E. (1996). One hundred years of forgetting: A quantitative description of retention. *Psychological Review, 103*(4), 734-760.

Schmidt, R. A., & Bjork, R. A. (1992). New conceptualizations of practice: Common principles in three paradigms suggest new concepts for training. *Psychological Science, 3*(4), 207-217.

Schneider, W. (1985). Training high-performance skills: Fallacies and guidelines. *Human Factors, 27*(3), 285–300.

Seibel, R. (1963). Discrimination reaction time for a 1,023-alternative task. *Journal of Experimental Psychology, 66*(3), 215-226.

Simonton, D. K. (1996). Creative expertise: A life-span developmental perspective. In K. A. Ericsson (Ed.), *The Road to excellence: the acquisition of expert performance in the arts and sciences* (pp. 227-253). Mahwah, NJ: Erlbaum.

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

Spiro, R. J., Feltovich, P. J., Coulson, R. L., & Anderson, D. R. (1989). Multiple analogies for complex concepts: Antidotes for analogy-induced misconception in advanced knowledge acquisition. In S. Vosniadou & A. Ortony (Eds.), *Similarity and analogical reasoning* (pp. 498-531). Cambridge, UK: Cambridge University Press.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science, 12*, 257-285.

Taatgen, N. A., & Lee, F. J. (2003). Production compilation: A simple mechanism to model complex skill acquisition. *Human Factors, 45*(1), 61-76.

Wiener, E., Kranki, B., & Helmreich, R. L. (Eds.). (1993). *Cockpit Resource Management*. London, UK: Academic Press.

Wright, T. P. (1936). Factors affecting the cost of airplanes. *Journal of the Aeronautical Science, 3*(2), 122-128.