A Mechanism-Based Framework for Predicting Routine Procedural Errors

Michael D. Byrne (byrne@acm.org)
Department of Psychology
Rice University, MS-25
Houston, TX 77005

Abstract

Routine procedural errors are facts of everyday life but have received little empirical study and have eluded prediction. Leading frameworks for thinking about such errors have not been successful in generating predictions, either. This paper describes the desiderata for error prediction, and notes that the MHP was a step in the right direction. Because it generally meets the criteria, ACT-R is proposed as a simulation framework for making predictions about routine procedural errors, and some of the critical mechanisms explored.

Introduction

Even in the execution of known routine procedures, people make non-random errors. Everyone has had this experience, whether it is leaving one's bank card in an ATM or failing to attach a promised file to an email message. While many such errors have little or no real cost, many such errors have dire consequences, including loss of human life (Casey, 1993). Clearly, an understanding of the cognitive and perceptual mechanisms underlying such error, and therefore knowledge about how to potentially defeat them, would be valuable. However, this problem has spawned surprisingly little research. Senders and Moray (1991, p. 2) identify probably the major explanation: "one reason for this is that error is frequently considered only as *result* or *measure* of some other variable, and not a phenomenon in its own right." Empirical work on systematic errors in the execution of routine procedures is dominated by anecdotal accounts (e.g., Casey, 1993) but controlled experiments on this subject are quite rare.

Before going into more detail, it is important to delineate scope. First, the current effort is not concerned with all forms of human error, but only routine procedural error. This is a more restrictive definition, and for the purposes of this discussion, will be taken to mean errors which occur during the execution of a routine cognitive skill, as defined by, for example, Card, Moran, and Newell (1983) and John and Kieras (1996). That is, the person involved in the activity has the correct knowledge and needs to execute that knowledge. While there are certainly a wide array of other error types, some of which even have useful functions (Ohlsson, 1996), this restricted domain of inquiry is both rich and significant. For example, this describes many of the errors made by extensively-trained and highly-motivated people in safety-critical situations, such as commercial pilots and medical professionals.

A second issue is the definition of error itself. A wide variety of definitions have been proposed, but many of them suffer shortcomings of one kind or another. For example, Reason (1990) defines human error as "the failure of a physical or mental action to achieve an intended outcome " (p. 9). The fundamental problem is that the same human cognitive-perceptual-motor system that produces "errors" is also the one which produces "correct" behaviors. Thus, it is difficult to posit an error definition based solely on either overt behaviors or internal psychological states; Reason's definition is hampered by reliance on the notion of "intended outcome." If an operator forms an "incorrect" subgoal in performing a complex task, but performs that subgoal correctly, is that an error? The answer is unclear and is not solved by the classic slip/mistake dichotomy (described below) because intentions are formed at multiple levels.

Instead, it is the joint internal and external context of task performance that defines a particular action as an "error." Thus, an error could be a failure to produce behaviors which meet a particular task demand (e.g. the correct sequence of actions is performed, but so slowly the biological sample dies), or failure to meet some requirement imposed by the tool or artifact used in the task (e.g. not selecting letter paper when printing an A4-formatted document), or even by the state of the task environment (e.g. lowering the flaps to reduce speed when airspeed is too high). Thus, one way to define an error in the context of a routine cognitive skill is *any action which causes the actor to fail to meet the performance requirements (either internal or external) of their task*. Most (but, critically, not all) performance requirements of many tasks are defined external to the agent performing the task. This will become important in later discussions of error mechanisms.

Extant Taxonomies

The dominant theoretical paradigm in this area is certainly the one proposed by Reason (1990), which is more a taxonomy than a theory. Reason classifies errors into two types: "mistakes," which are the result of forming an incorrect intention to act, and "slips," which are failures to correctly execute an intention. These are tied to Rasmussen's (1987) skill-rule-knowledge (SRK) hierarchy of skill execution.

At the lowest levels of skill acquisition, behavior is controlled by essentially declarative knowledge that must be interpreted on-line by the cognitive system, which is taxing. Further along the skill curve, behavioral control is dominated by explicit rules, hence behavior is said to be rule-based. Finally, behavior at the skill level is governed by stored patterns of preprogrammed instructions,

corresponding roughly to Shiffrin and Schneider's (1977) notion of automaticity. In the account of errors derived from this framework, mistakes are usually errors at the knowledge level; that is, the person making the error has incorrect knowledge about how to perform the task. While this is certainly the explanation for many errors, it does not appear to apply to many interesting forms of systematic procedural error in which the person *does* know the correct set of steps. Reason generally attributes errors at the skill or rule levels of performance to various kinds of failures of attention: inattention, overattention, informational overload, and the like. This has enormous face validity and has been useful in understanding errors in a variety of contexts. However, from the perspective of trying to predict errors, this account simply shifts the locus of the problem to another area of research, attention, which presently lacks a predictive performance theory.

Another prominent figure in the study of errors is Norman (e.g. Norman, 1988). Norman's well-known "seven stages of action" framework is depicted in Figure 1. In this framework, each goal to be achieved must be translated into an intention to act upon the world, then that intention converted into a sequence of actions, and so on. Once the action has been taken, it is necessary to assess the outcome, which requires perceiving the state of the world, then interpreting that intention, etc.

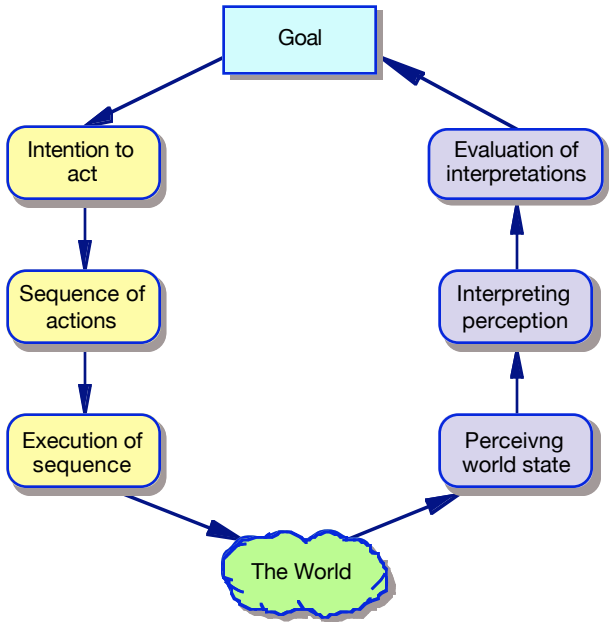

Figure 1. Norman's "seven stages of action"

It is possible to base an error taxonomy on this framework by asking the question "at what stage did the error occur?" Norman's framework is in some ways more comprehensive than the Reason scheme because it gives a larger role to the perception of the outcomes of action. However, Norman's framework is also neither mechanism-based nor predictive. However, Zhang, et al. (2002) describe a taxonomy based on an extended version of this framework that does make some attempts to be predictive. This is clearly a step in the right direction but the effort is geared more toward error classification than error prediction.

If the goal is prediction of errors, there are two fundamental problems with taxonomic efforts. First, from the standpoint of causal mechanisms, they group unlike things together and like things apart. For example, there a variety of cognitive mechanisms which can produce errors in executing an action sequence, some of which can produce errors at other stages of action. Thus, "execution errors" will be heterogeneous, while mechanisms that might produce errors in other stages will be categorized as being different from those execution errors which share their cause. This defeats the purpose of a taxonomy. Second, and more critically, these taxonomies, because they are not based on mechanisms, cannot generally be predictive.

In defense of extant taxonomies, they clearly have had value in non-predictive real-world fault diagnosis and system design, and have successfully guided various error interventions—this is not intended as a criticism of the taxonomies for the purposes for which they were developed. For the purpose of prediction, however, other methods will be required.

Criteria for a Framework

If the goal is an obvious one, to reduce or prevent routine procedural errors, a method which predicts which errors will be made, preferably with a frequency estimate attached, would certainly be helpful. No extant framework or model supports such prediction in general (though there have been specific tasks and errors for which this has been approached, for example see Byrne & Bovair, 1997; Gray, 2000). In fact, in the general case, there is not even a good empirical method for such an inquiry. One might try, for a particular task, to recruit or observe human operators to get error frequency counts. However, this will not necessarily yield useful predictive data unless the number of subjects that can be recruited and run is quite large. Especially in domains where the operators are highly-trained and in great demand (again, pilots and medical professionals are excellent examples), collecting even a small sample can be prohibitively expensive. A second difficulty faced by such an empirical enquiry is the determination of causality. Even when an error is observed under controlled conditions, the root cause of the error is not always clear, and thus strategies for remediation are not apparent.

One possible solution to this problem would be to develop a model based on human operators that could perform the relevant task repeatedly under a variety of conditions and faster than real time. This would allow for the collection of large amounts of data and thus more stable frequency counts and, critically, this could be done on an *a priori* basis. Note also that this idea—essentially, Monte Carlo simulation—requires some stochasticity in either the operator model, the task environment, or both. Otherwise, every simulation run would yield the same outcome, which would hardly be useful.

How could we arrive at such a system? There are probably multiple possible approaches, but it is clear that to be successful in safety-critical applications with demanding time constraints, such an approach requires a sophisticated model of the operator at a fine grain size of behavior. It has

to include end-to-end processing from perceptual to cognitive to motor, and it has to have a variety of mechanisms in it which are capable of producing errors even when the operator model is supplied with the "correct" knowledge.

The Model Human Processor (MHP) of Card, Moran, and Newell (1983) might serve as a starting point for such an endeavor. The MHP was a synthesis of the cognitive psychology and human performance of the time, cast somewhat in the form of a "modal" theory of cognitive architecture. The system consists of a number of memories and "stores" as well as cognitive, perceptual, and motor processors. The processors are themselves serial but work in parallel with one another. Each of the MHP stores had certain information-processing parameters and the processors were guided by several principles of operation.

For example, the MHP working memory has limited capacity and degrades as a function of both decay and interference. The MHP's long-term memory is cue-based, affected by frequency, recency, and similarity. The principles guiding the cognitive processor included standards such as search through a problem space and constrained adaptation supported by ubiquitous learning. In very general terms, the MHP was not especially contentious at least in part because it tended to be more inclusive than exclusive. Also, one of its great strengths was also its fatal weakness in terms of error prediction: the MHP was not implemented as a running system, so commitments were never made to the exact form taken by most of the proposed mechanisms.

The lack of commitments to specific details skirted many of the kinds of arguments about representation and process that occupy much of the cognitive science literature today; however, in some circles (primarily within the HCI community) this became a guiding conceptualization. But, of course, it does not meet the goal of allowing prediction by simulation.

What is needed is an instantiation of the MHP that does address the range of cognition, perception, and motor activity and that is also executable. This would allow for repetitive simulation to ultimately obtain frequency counts. In addition, if a trace of the system could be kept, it may be possible to record the causal mechanism which produced each error. This may lead to insight into what might be possible in terms of remediation.

A Candidate System: ACT-R

A system that appears to meet the desiderata laid out thus far is ACT-R 5.0 (Anderson, Bothell, Byrne, & Lebiere, 2002; this version subsumes all previous versions of ACT-R, including ACT-R/PM). It is important to be clear that this is not a suggestion that ACT-R is the only possible or ultimately best such system, but it currently does satisfy many of the aforementioned conditions. And though ACT-R is quite comprehensive, this is not necessarily an endorsement of either the completeness or correctness of ACT-R's mechanisms. Rather, this is a claim that ACT-R is a tenable platform for exploring the idea of predictive human error simulation for routine procedural errors.

The major components (except for the Speech and Audition Modules) of ACT-R 5.0 are depicted in Figure 2. Like the MHP, ACT-R contains multiple active processing units, which include the perceptual-motor modules and the production system pattern matcher. Also like the MHP, each one of these units is essentially serial but all the units operate in parallel with one another. As is usual for production systems, it contains two memories, a declarative memory and a procedural memory (though the Vision and Audition Modules each contain their own temporary stores as well). Communication between system components is managed through a representation of the current system state, which resides in a set of special memory elements referred to as "buffers."

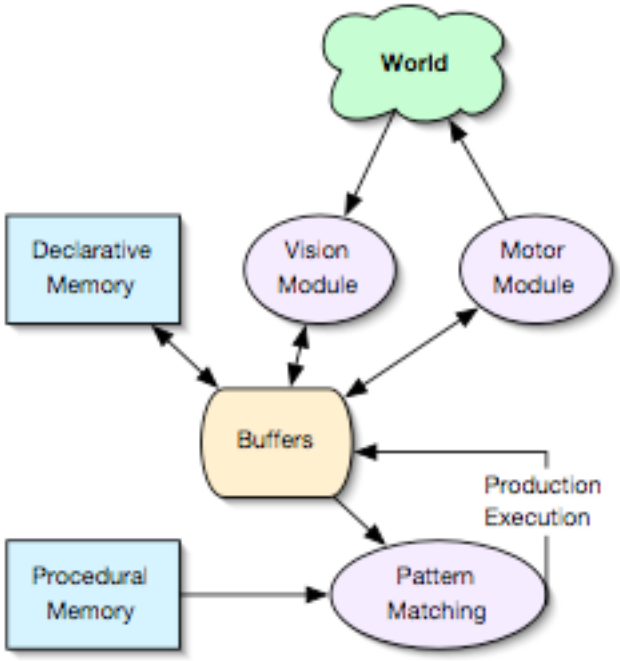

Figure 2. Major components of ACT-R 5.0. Boxes represent memories, ovals active processes.

Unlike the MHP, however, the workings of the various processes and memories have been specified in detail (see Anderson, Bothell, Byrne, & Lebiere, 2002 for a more complete description; see Anderson, 1990, for the "rational" justification for many of these mechanisms) and the system is runnable, producing as its output a timestamped sequence of behaviors, both overt (e.g. keystrokes) and covert (e.g. retrieval from declarative memory). In order to produce this output, ACT-R requires two inputs: knowledge and an environment with which to interact. The knowledge consists of declarative memory elements (termed "chunks") and production rules, as well as some parameters which affect how that knowledge will be used (more on this shortly). ACT-R also requires a runnable environment which responds to its actions and produces stimuli.

So, ACT-R meets the "runnable" criterion. The next question is whether or not ACT-R is capable of producing errors in the execution of routine procedures. And if so, how might that lead to an improved taxonomy or remediation strategies?

The answer to the first question appears to be "yes." ACT-R contains a number of mechanisms that can produce errors despite the presence of "correct" knowledge. Furthermore, in places where it does not appear that ACT-R has appropriate error-producing mechanisms, modifications that would produce them appear to be straightforward. Thus, while it most certainly is not a complete or necessarily correct theory everywhere, it might serve as a starting point for systematic explorations into routine procedural errors. To make the case more clear and concrete, two mechanisms will be elaborated in some depth.

Procedural Memory Mechanisms

In ACT-R, the basic unit of procedural memory is the production rule (or simply production). This is a condition-action pair, IF a particular pattern is present in the buffers, THEN take one or more actions. These actions include modifications of the contents of a buffer (e.g. a change in the state of the current goal) and requests of the other subsystems (e.g. retrieve a declarative memory, initiate some perceptual-motor action). It seems apparent that if the correct productions are in place, then the correct action sequence should be output.

However, this is not always the case. It can be (and often is) the case that more than one production will match on a given cycle. Since the production system is serial—that is, only one production at a time can fire—there needs to be a way to arbitrate when multiple productions match. In ACT-R, this is done with a simply utility calculation. Each production has a utility (U) associated with it, and the matching production with the highest utility is the one selected.

The utility is computed with the formula PG–C, where P represents the estimated probability that the firing of that production will lead to success in pursuing the current goal and C the estimated cost (in terms of time) until that success if the production is allowed to fire. G represents the value of the current goal. Importantly, this computation is noisy. This means that the production with the highest utility will not always be chosen. In fact, the system delivers what is sometimes referred to as "soft max" behavior; the probability of a production being selected is higher when its utility is higher, and thus the one with the highest utility is the one mostly likely to be selected. But it will not always be. Thus, when in a situation where there are multiple viable alternatives, ACT-R will choose stochastically from among them. This can lead to the selection of strategies or actions which are not as well-matched as they might be. This, in turn, can lead to failure to meet task requirements, even when the actions taken are by some definition "correct."

The second issue is with the PG–C quantities themselves. P and C for each production can be set by the modeler or learned by the system over time; they are intended to represent the true probabilities and costs in the history of the agent being modeled. G is generally left fixed. However, this leads to a potentially interesting situation: what happens when the environment changes? This might be best illustrated with a somewhat simplified example.

Suppose the model has two strategies for executing a simple procedure, say, driving home from the office. There are two possible routes, with known probabilities and costs, which leads the model to consistently choose route A. However, on a particular trip home, unbeknownst to the model, there is utility construction along route A. Because of the model's preferences based on P and C, it will choose route A. This can be an error if the construction yields a delay long enough that the performance requirements (e.g., get to day care by 6:30 pm) are not met. Thus, the system can err despite the presence of the correct knowledge. Note that the situation described above pushes on the definition of "correct;" in the changed environment, it might be reasonable to say that the model's knowledge is no longer correct. However, this is a fine line, since the model does have the correct knowledge about how to navigate route A. Ultimately, this semantic question is probably not that important in predicting errors; we'd like to know where human errors are likely to occur, and humans are likely to have similar imperfections in the tuning of their knowledge when the world changes.

Another important point here is that while ACT-R has default values it will supply for P, G, and C, these can be taken with a grain of salt. With Monte Carlo techniques, it is be possible to simply repeatedly sample the space of reasonable values and from that derive a distribution of predicted error frequencies. Obviously, the better the values selected, the better the model output will be, but supplying even a good approximate range should yield useful outputs.

Declarative Memory Mechanisms

ACT-R's other memory system is an even richer potential source of error. To understand why, some detail regarding how it works will be necessary. First, ACT-R's declarative memory system is engaged when a production requests the retrieval of a chunk matching a particular pattern. If the system is able to retrieve such a chunk, then the retrieved chunk is placed in one of ACT-R's buffers so it can be accessed directly by productions.The time that it takes is a function of the retrieved chunk's activation, with more active chunks taking less time to be retrieved. Chunk activation plays other key roles as well. First, because there is a system-wide threshold for chunk activation, it determines whether a chunk can be retrieved at all. Second, when multiple chunks match the pattern specified by the retrieval, the chunk with the highest activation is the one actually retrieved. Thus, chunk activation is a critical quantity. The activation of chunk *i* is given by the following equation:

$$A_i = B_i + \sum W_j S_{ji} + \varepsilon \qquad [1]$$

where epsilon represents stochastic noise added to all chunk activations at each retrieval request. Like with production utilities, this leads to soft max behavior; the chunk with the highest pre-noise activation is the one most likely to have the highest post-noise activation, but this is not guaranteed. The other terms require explanation as well. The summation is across each chunk which is an element of the current goal (termed a "source;" note that goals in ACT-R are also chunks), where W_j is the total source activation W (usually W is set to 1) divided by the number of such

chunks. S_{ji} is the strength of association between the source chunk *j* and the target chunk *i*. In essence, this is the summed strength of the chunk's relationship to the current context, scaled to account for the amount of such context.

The other equation of importance is the one determining base-level activation of a chunk, B_i in equation 1:

$$B_i = \ln\left(\sum t_j^{-d}\right) \quad [2]$$

where the summation is across each previous access *j* of the chunk, *t* represents the time since that access, and *d* is system-wide constant (usually 0.5). Thus, the base-level activation of a chunk is a function of both frequency and recency of access, with more accesses leading to higher activation, and more recent accesses weighing more heavily than older accesses.

Taken together, what this means is that despite the presence of the "correct" chunk in declarative memory, it may not be accessible when requested. Furthermore, an "incorrect" chunk may be retrieved in its place. The fact that such an event can occur is certainly promising for the prospective error modeler, but it is more interesting to consider the conditions that are most likely to lead to such failures.

First, note that chunks' base-level activations decay over time. This can be overcome with additional accesses (e.g., rehearsal), but of course additional access takes time and, since the declarative memory system can only retrieve one chunk at a time, this prevents it from being used for other parts of task performance. This makes the obvious prediction that dynamic pieces of task information (e.g. partial products, state information) will be forgotten if not accessed enough; more importantly, it specifies the likelihood of forgetting under various conditions.

Additionally, for most chunks, they require more than just base-level activation to be above threshold, they require the activation spread from the current context (the slots of the current goal). Retrievals can fail if there is a shortage of this activation as well. What conditions lead to this? The two components in the equation tell us. First, there is the strength of the sources (W_j). As the goal chunk gets bigger and bigger in an attempt to store more state or context, spreading activation gets more diffuse, making all retrievals both slower and more likely to fail. This is essentially a workload effect; the greater the workload of the system, the more likely it is that a retrieval will fail, possibly leading to an error.

Second, there are the effects of the strength of association, which is essentially an index of the specificity of the cues. Cues that are too general are less effective. For example, adding a red indicator light to cue the pilot that something is in an error condition, even if the pilot sees the indicator, is unlikely to help if many things are associate with a red indicator light. Again, this is hardly a novel idea or prediction. But again, ACT-R makes specific predictions about the effects of such manipulations, and also makes predictions about how all three factors discussed trade off with one another.

Thus, even a model with the "correct" set of productions can fail if it cannot access the declarative information it needs to meet the task requirements successfully. Moreover, this effect is stochastic and depends on the values of some system-wide and memory-specific parameters. Again, a space of reasonable values could be sampled repeatedly to generate error frequency predictions.

Timing, Combinations, and Cascades

Space constraints prohibit an exhaustive discussion of other potentially error-generating mechanisms in ACT-R, but such mechanisms, or the potential for them, certainly exist in the other ACT-R subsystems such as the visual system. For example, the visual system has limited bandwidth (only one set of eyes) meaning, among other things, difficult visual searches can be time-consuming.

In fact, timing is another crucial error source. Many tasks in dynamic environments (e.g., again, aviation and medicine) have stringent timing requirements, and performance that is simply too slow may also be considered erroneous. Timing plays a critical role internally in ACT-R as well. Since the various systems act asynchronously, this raise the possibility of the right thing being done at the wrong time, especially, too late. For example, some visual search takes longer than expected, after which time the critical piece of state information is no longer accessible because it has decayed. This points out another important aspect of the system's behavior: it is often potentially the case that no single mechanism is "responsible" for a particular error, but rather that multiple mechanisms were involved. This suggests that one of the reasons that taxonomies of errors may be so difficult to successfully construct, is that the lines of demarcation are not clear at a mechanistic level.

Furthermore, when searching for the cause of a particular error, in some sense the cause may not always be proximal. ACT-R's behavior is both non-linear (see equation 2 for an example) and stochastic, and thus chaotic in the mathematical sense. That is, a small perturbation in the activation of a chunk at time *t* may produce a much larger shift in behavior at time $t + n$. Anyone who has run multiple people through an experiment where the same conditions are presented multiple times can appreciate the face validity of such a system as a model of human behavior. And again, this points to the necessity of Monte Carlo simulations.

Discussion

The central point here is not that ACT-R is the ultimate solution to predicting human error, rather, that it is a candidate system that at least potentially meets the requirements for making such predictions, at least in the case of routine procedural errors. The availability of such a system is a relatively recent innovation; now is the time to take seriously the idea of simulation modeling for human error prediction.

Obviously, the amount of effort and resources required to do such modeling is not trivial. The knowledge engineering required to model people with even moderate real-world expertise in executing complex but routine (for them) procedures is significant, and running large Monte Carlo

simulations is also resource-intensive. However, as noted, even traditional empirical methods in such domains can be quite expensive to administer and often provide quite limited results. For safety-critical applications where millions of dollars and/or human lives are at stake, the costs of such simulation-based prediction may well pay off.

This is something of a departure from how such errors have been approached in the past. Systematic exploration of human error has traditionally been the domain of human factors researchers, while this simulation-based approach requires a great deal of expertise in computational cognitive modeling, an area more familiar to the cognitive scientist. Fortunately, these communities overlap and my hope is that people from both groups will be drawn to this kind of approach.

For that to happen, the approach will have to demonstrate some empirical success. Such efforts are have been initiated (Schoppek, Boehm-Davis, Diez, Hansberger, & Holt, 2000; see also Byrne & Kirlik, 2002, 2003) and look promising. Hopefully other candidate systems (such as APEX, Freed & Remington, 1998) can be identified and put into use this way to provide a broader gauge of the ultimate tractability of this approach.

Acknowledgements

I would like to acknowledge the support of the Office of Naval Research under grant number N00014-03-1-0094 and the National Aeronautics and Space Administration, grant number NDD2-1321. The views and conclusions contained herein are those of the author and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ONR, NASA, the U.S. Government, or any other organization.

I would also like to thank Alex Kirlik and Wayne Gray for numerous helpful discussions and Stellan Ohlsson for helpful comments on an earlier draft.

References

Anderson, J. R. (1990). *The adaptive character of thought*. Hillsdale, NJ, USA: Lawrence Erlbaum Associates, Inc.

Anderson, J. R., Bothell, D., Byrne, M. D., & Lebiere, C. (2002). An integrated theory of the mind. Manuscript submitted for publication and available at http://actr.psy.cmu.edu/papers/403/IntegratedTheory.pdf.

Byrne, M. D., & Bovair, S. (1997). A working memory model of a common procedural error. *Cognitive Science, 21*, 31–61.

Byrne, M. D., & Kirlik, A. (2002). Integrated Modeling of Cognition and the Information Environment: Closed-Loop, ACT-R Modeling of Aviation Taxi Errors and Performance. Technical Report AHFD-02-19/NASA-02-10, Institute of Aviation, University of Illinois at Urbana-Champaign.

Byrne, M. D., & Kirlik, A. (2003). Integrated Modeling of Cognition and the Information Environment: A Closed-Loop, ACT-R Approach to Modeling Approach and Landing with and without Synthetic Vision System (SVS) Technology. Technical Report AHFD-03-4/NASA-03-3, Institute of Aviation. University of Illinois at Urbana-Champaign.

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

Casey, S. (1993). *Set phasers on stun*. Santa Barbara, CA: Aegean Publishing.

Freed, M. and Remington, R. (1998) A conceptual framework for predicting errors in complex human-machine environments. In *Proceedings of the Twentieth Annual Conference of the Cognitive Science Society* (pp. 356–361). Mahwah, NJ: Erlbaum.

Gray, W. D. (2000). The nature and processing of errors in interactive behavior. *Cognitive Science, 24*(2), 205-248.

John, B. E., & Kieras, D. E. (1996). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction, 3*, 320-351.

Norman, D. (1988). *The design of everyday things*. New York: Doubleday.

Ohlsson, S. (1996). Learning from performance errors. *Psychological Review, 103*, 241–262.

Rasmussen, J. (1987). The definition of human error and a taxonomy for technical system design. In K. D. J. Rasmussen, & J. Leplat (Ed.), *New technology and human error (pp.* 53–62). Chichester: John Wiley & Sons.

Reason, J. T. (1990). *Human error*. New York: Cambridge University Press.

Schoppek, W., Boehm-Davis, D. A., Diez, M., Hansberger, J. T., & Holt, R. W. (2000, August). Letting ACT-R fly-A model of the interaction between trained airline pilots and the flight management system. Paper presented at the 7th Annual ACT-R Workshop, Carnegie Mellon University, Pittsburgh, PA

Shiffrin, R. M., & Schneider, W. (1977). Controlled and automatic human information processing II: Perceptual learning, automatic attending and a general theory. *Psychological Review, 84*, 127–190.

Zhang, J., Patel, V. L., Johnson, T. R., & Shortliffe, E. H. (2002). Toward an action based taxonomy of human error in medicine. In W. D. Gray & C. D. Schunn (Eds.), *Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society* (pp. 970–975). Mahwah, NJ: Erlbaum.