9-2005

Prepare your mind for learning

Deborah K. Smith
University of Nevada, Las Vegas

Trevor T. Moores
University of Nevada, Las Vegas

Jerry Cha-Jan Chang
University of Nevada, Las Vegas, jchang@unlv.nevada.edu

By **Deborah K. Smith, Trevor Moores, and Jerry Chang**

Prepare Your Mind for Learning

The learning process must evolve and expand throughout one's IT career. Most would agree that's often easier said than done. Here are some ways professionals can overcome mental blocks that may prevent learning.

One of the greatest challenges facing information technology professionals is the need to stay current. The body of knowledge generated by researchers and practitioners is growing at a mind-boggling rate, while the business environment in which we must ply our multidisciplinary trade is constantly changing. For IT professionals, in industry, academia, and public service, this situation creates an ethical imperative to engage in continuous learning for the purposes of acquiring new knowledge and updating or enhancing old skills. Unfortunately, there are forces in our environment and in our very nature that conspire against us, often making it all but impossible for us to learn.

One of the most important and least understood impediments to learning is *metacognitive miscalibration*, or MM. Fortunately, psychologists and philosophers offer sound advice for recalibrating our metacognitive judgments and overcoming the ill effects of MM. Here, we explain what MM is, how it affects our ability to learn, what causes it, and how we can recalibrate our metacognitive judgments.

IT professionals encounter a number of meta words in their work. Metalanguage is language used to describe language, a useful tool for compiler developers. Metadata is data used to describe data, a requirement for databases. Metacognition is, literally, thinking about thinking. Metacognition plays a part in both monitoring and controlling our cognitive functions [5]. In its monitoring role, metacognition informs us of our knowl-

edge and assesses our performance.

For instance, if someone asks, "Can you create a Web-enabled time and services billing application?," your metacognitive processes quickly assess your experience with Web-enabled applications and your knowledge of time and services billing applications, form a metacognitive judgment, and prompt you to respond, "Sure, no problem." Throughout the project, your metacognitive processes inform you of your progress and indicate whether you have the ability to complete the task at hand. These metacognitive judgments are input to the control activities of metacognition, providing your cognitive processes with the information needed to change and adapt.

The trouble is, our metacognitive judgments can often be wrong—the MM phenomenon [2]. Evidence of MM can often be found in the classroom, but it certainly occurs in the real world as well. Typical manifestations go something like this: "Professor, I need to talk to you about my grade. I think there must be some kind of mistake. I took computer classes in high school. I just know I didn't fail that test!" Sometimes students exhibit a more alarming form of MM: "I don't see why they make us take these courses. We already know all this stuff. It's just a waste of time!" Casual observation and some preliminary research indicate that MM students tend to learn less than their peers [6].

MM doesn't disappear after college graduation. In fact, you might recognize some of its symptoms—reluctance to attend training programs or participate in professional organizations, inability to stick to a task long enough to master it, aversion to admitting ignorance, and chronic boredom. It is often tempting to dismiss these behaviors as merely expressions of arrogance. While arrogance may indeed play a part, poor metacognitive judgment—MM—generally lies at the root of the problem.

What are the Effects of MM?

Metacognition, while not primarily affective, does have an effect on motivation [4]. Programmers who believe they have the ability to create the next revolutionary piece of software will devote countless hours to the task; those who do not share this belief quickly give up. Students who believe they are ready for their final exams will get a good night's sleep before the test; those who do not will cram.

There is evidence that metacognition, and particularly MM, has a deeper influence on learning than simply motivating study [2]. Both the incompetent and the highly miscalibrated lack the ability to assess others' performance as well as their own, limiting their ability to learn from example. The most obvious demonstration of this is one you likely experience at least twice each day: driving. We are all superlative drivers; everyone else, however, is so bad they should not be allowed to own, much less drive, a car. Logically, this cannot be true, although we can all fall into the trap of believing everyone else is so highly miscalibrated they are unable to learn from our most excellent examples.

Though he doesn't use the term metacognition, Armour also suggests that an inability to deal with what we do not know has serious implications for our ability to develop software systems [1]. Ignorance is defined in terms of five orders, numbered zero through four. The zeroth order of ignorance, 0OI, is defined as the absence of ignorance, in which knowledge is present and demonstrable, such as the ability to build an information system that satisfies the requirements of the end user. The first order, 1OI, is ignorance in the usual sense, recognized lack of knowledge; there is something I do not know, but at least I recognize the fact and hopefully this will prompt me to try and find the answer. For instance, I may know the project deadline has slipped but I cannot say by how much until I reestimate the time to complete all remaining tasks.

The second order, 2OI, is unrecognized absence of knowledge; we do not know what it is that we do not know, like the missing or misunderstood requirements for an information system. For example, the software development project continues with the development team blissfully unaware that the system they are building is not the system the users want. The third order, 3OI, is absence of a process for recognizing ignorance; in other words, we have no method that will determine if there is something we do not know. We could just build the system and let the users tell us what we forgot or did not understand, but it would be better to have a process by which these problems are avoided. The fourth order, 4OI, is meta-ignorance: ignorance of the orders of ignorance.

Armour's second order of ignorance is what we would consider MM: someone suffering from 2OI is ignorant and unaware of it—clueless. Not knowing that you do not know—incompetence so complete it is not recognized is the ultimate expression, both cause and effect, of MM. Formal education is one process by which we remedy both 1OI and 2OI. For those who teach, the implication is that you must first move students from 2OI to 1OI—recalibrate their metacognitive judgments—before you can expect them to learn. For those who are primarily self-taught, including many industry practitioners, you must first know that you do not know, acknowledge

your ignorance, before you can begin to learn. To self-taught individuals, this may seem intuitive; why would you seek instruction, from a book or other source, if you were not aware of some deficiency in your knowledge? The problem is, you may need instruction and be unaware of this need.

What Causes MM?

According to Kruger and Dunning [2], the relationship between metacognition and ignorance is such that MM impairs learning and ignorance begets MM. The same cognitive processes that make competence possible are required not only for accurate metacognition but for accurate assessment of others' performance. The miscalibrated are unable to distinguish between good and poor examples, unable to assess their own performance, and, as a result, unable to learn from their mistakes. This would seem to be an impasse.

One explanation of the source of miscalibration is presented by George Leonard [3], who describes learning as a repeating cycle of knowledge acquisition and knowledge assimilation. When knowledge is plotted against time, there is a stair-step learning curve in which each period of increase is followed by a plateau. Knowledge assimilation takes place on the plateau. Using characteristic learning curves, Leonard profiles four different types of learners: dabblers, hackers, obsessives, and masters.

Dabblers are the epitome of the maxim "jack of all trades, master of none." They embark upon new experiences with all the exuberance of puppy love, abandoning the romance when they reach a plateau and no longer recognize performance improvements. Hackers are content to reach a plateau and stay there, deluding themselves that they know enough. Obsessives, on the other hand, can never learn enough fast enough, often exhibiting a meteoric rise followed by burnout when their efforts to achieve constant performance improvements fail. Masters, according to Leonard, understand the value of the plateau, working their way through the plateau secure in the knowledge that there is more to learn at its end.

Leonardian masters are the only type of learner that does not suffer from MM. Dabblers and hackers both believe a little learning is enough; they suffer from 2OI but either do not care (dabblers) or feel that it does not matter anyway (hackers). Obsessives are miscalibrated and 3OI; their knowledge is never reality-checked, so they lack a process for discovering their ignorance.

Existing within a culture of dabblers and hackers who lionize the obsessive, the IT profession seems particularly vulnerable to environmentally induced miscalibration. Who has not been subjected to tales of the workaholic software engineer who practically lives at the office, subsisting on soda and junk food, slinging code until he passes out at his desk? Who among us did not succumb to planner mania in the 1990s, bragging as we drank our designer coffee about how many hours we worked last week, tacking on a few extra for good measure? And who has not, at some point, felt completely overwhelmed by the sheer volume of information we must handle, not to mention the rate at which this information changes?

> Knowledge acquisition is often akin to building a beach by adding grains of sand. Celebrate the buckets of sand when they come; learn to recognize and cherish the individual grains.

Those who feel the pressure of deadlines in the midst of leading-edge projects often seem forced into the role of Leonardian hackers and may be pushed into 2OI as a result. When industry rewards hacking, it reinforces the mistaken notion that a little knowledge is enough, colluding with, if not actually contributing to, MM. Yet, for those of us who truly enjoy working in IT, asking us not to dabble in the new is like asking a kid not to play.

How Can We Recalibrate our Metacognitive Judgments?

Kruger and Dunning [2] suggest that domain incompetence and MM are so inextricably intertwined that acquisition of competence is the only real cure for MM. This would seem to create a Möbius Loop: If I cannot learn because I am miscalibrated and I cannot recalibrate without learning, am I not doomed to perpetual ignorance? To defeat this conundrum, we will follow Armour's advice [1] and eliminate 3OI by developing a process for rec-

ognizing 2OI, then apply that process to eliminate 2OI and ultimately 1OI.

For Kruger and Dunning, education is the preferred process for eliminating 3OI, but they admit learning is hampered by the existence of 2OI. If the preferred solution is not feasible, then is there a less-optimal solution that is feasible? We believe there is. There are a number of things we can do to prepare our minds for learning. These things may seem obvious to some, but are what the best learners do almost instinctively on their way to 0OI.

Cultivate a beginner's mind. No matter how much we know there is always something more we do not know. To recognize there are holes in our knowledge is the first step to doing something about it. No matter how easy we believe the task at hand to be, we should always assume there is something we can learn from doing it. Knowledge acquisition is akin to building a beach by adding grains of sand. Celebrate the buckets of sand when they come; learn to recognize and cherish the individual grains.

Question your knowledge. Sometimes we simply have the wrong information. We may have read or been told something wrong, misunderstood, made invalid assumptions, reached an erroneous conclusion, or suffered faulty recall. In any event, it is a good idea to review what we think we know and verify its accuracy. This can be accomplished through reading, discussing with others, and attending lectures at users' groups, conferences, and training courses. Think how much better it feels to uncover your own mistakes rather than have them pointed out.

Seek criticism of your work. This is the flip side of the previous suggestion. Have others evaluate your work and make suggestions, whether a co-worker, supervisor, or someone from outside your organization. A good reviewer is one who will provide constructive criticism and value the opportunity to learn from the experience.

Review the work of others. We are occasionally called upon to review conference proceedings and journal articles prior to publication. More often, we review the work of students. Though these appraisals are primarily evaluative in nature, we try always to learn something when we do them. Seeing how someone else approaches a task and comparing their approach to our own can be a very enlightening, and sometimes humbling, experience.

Teach. It is an old maxim, often repeated in academia, that the best way to really learn a subject is to teach it. Most faculty in IT disciplines have experienced the thrill and the terror of teaching a new course. There is nothing quite like having to explain a concept to a group of students to make us realize that we do not know as much as we thought, and there is almost always a student in the class who will raise an issue we have never really thought about before; these students make teaching especially enjoyable. Even teaching the same course from term to term can be a real learning opportunity. Updating course materials, searching for better textbooks and better teaching materials, and discussing teaching methods with other instructors all offer opportunities for the teacher to learn.

Those outside academia have opportunities to teach as well. Mentor a junior. Offer to hold an informal training class on some subject you feel others in your work group might find useful. Offer to guest lecture at your local college; most faculty welcome this, and students generally enjoy having people from industry talk with them.

Relish the plateau. This is perhaps the most difficult for IT professionals and academics alike. There is so much to know, learn, do, and so little time; we often feel forced into the life of the obsessive hacker. The Leonardian plateau is a resting place for our minds: a period of time in which the new grains of knowledge find their place among, or displace, the old. This is also when we can look forward to the next upward climb along the path of knowledge and prepare our minds for learning. C

References

1. Armour, P.G. The five orders of ignorance. *Commun. ACM 43,* 10 (Oct. 2000), 17–20.
2. Kruger, J. and Dunning, D. Unskilled and unaware of it: how difficulties in recognizing one's own incompetence lead to inflated self-assessments. *J. Personality Soc. Psych. 77,* 6 (Dec. 1999), 1121–1134.
3. Leonard, G. *Mastery: The Keys to success and Long-Term Fulfillment.* Plume, NY, 1991.
4. Metcalfe, J. and Shimamura, A.P. Editors' Preface. *Metacognition: Knowing about Knowing.* MIT Press, Cambridge, MA, 1996.
5. Nelson, T.O. and Narens, L. Why investigate metacognition? *Metacognition: Knowing about Knowing.* J. Metcalfe and A.P. Shimamura, Eds. MIT Press, Cambridge, MA, 1996, 1–26.
6. Smith, D.K. Metacognitive recalibration in a computer literacy course. In *Proceedings of the ISOneWorld Conference (*Las Vegas, NV, Apr. 4–5, 2002).

Deborah K. Smith (deborah.smith@ccmail.nevada.edu) is an assistant professor of MIS at the University of Nevada, Las Vegas.
Trevor Moores (trevor.moores@ccmail.nevada.edu) is an assistant professor of MIS at the University of Nevada, Las Vegas.
Jerry Chang (jchang@unlv.nevada.edu) is an assistant professor of MIS at the University of Nevada, Las Vegas.