Cognitive Systems Engineering: The Hype and the Hope

Gary Klein and Sterling Wiggins, Applied Research Associates
Steven Deal, Deal Corp.

Cognitive systems engineering is a value-added technology offering many benefits that outweigh its costs.

Pilots are assigned a runway as they approach the destination airport, and they enter the runway into the flight management system along with related data such as altitude constraints at different waypoints. However, the air traffic controller might later instruct pilots to use a different runway, in which case they enter this information into the FMS and receive confirmation that the system understood the change.

But will the system remember to use the original altitude constraints, which are still in effect? That's what a good teammate would do. In at least one major airplane, the Airbus A-320, the system deletes those altitude constraints when the pilot enters a new runway, increasing the risk of a midair collision.

Will your pilots anticipate this problem? Two cognitive systems engineers, Nadine Sarter and David Woods, have some good news for you. Only four of the 18 professional pilots they tested in a simulator missed this problem, and of the 14 who either anticipated it or noticed it, 12 recovered in time to reenter the altitude constraints ("Team Play with a Powerful and Independent Agent: A Full-Mission Simulation Study," *Human Factors*, vol. 42, no. 3, 2000, pp. 390-402).

So the odds are heavily in your favor. Of course, you might wonder why the system was originally programmed like this.

Lack of support for users' cognitive requirements is evident in more familiar situations, such as the Blue Screen of Death associated with the Windows operating system. To someone intimate with Windows, the BSoD might be a useful operator message. But for most of us, who regard computers as tools, the message leads to questions, complications, and lost productivity. A next-generation "transparent solution" that automates the restart and remedial procedures is just as opaque and only further undermines confidence in the OS.

COGNITIVE SYSTEMS ENGINEERING

We define *cognitive systems engineering* as the effort to support the cognitive requirements of work. CSE is primarily applied to design information technologies to make them easier to use and more likely to be adopted. Developers must produce a system that not only "works," but that also factors in operators' varying skill levels and attentiveness as well as nonstandard conditions such as last-minute changes in a runway assignment.

In 1994, the Standish Group issued its first CHAOS report documenting the billions of dollars wasted on failed software development projects and examining why so many run into trouble (www.standishgroup.com/chaos_resources/chronicles.php). The report concluded that the two primary reasons were incomplete requirements and lack of user involvement. CSE tries to tackle both of these problems.

The origins of CSE can be traced back to the Three Mile Island accident in 1979. Jens Rasmussen studied nuclear power plant control rooms, particularly the operator interfaces, and found that designs that mechanically piped information to operators sometimes interfered with control room staff members' attempts to understand what was happening and adapt to circumstances (*Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*, Elsevier Science, 1986).

Rasmussen's research directly influenced a generation of CSE specialists including Morton Lind, Erik Hollnagel, David Woods, and Kim Vicente. Donald Norman has also shaped the development of CSE.

CSE practitioners on a design team typically first identify the most important cognitive functions of the system under consideration. They then determine how the user performs these functions, what makes these functions difficult to execute, what kinds of errors users commonly make, and what kinds of tacit knowledge the user needs.

CSE MYTHS

Several popular beliefs have emerged about CSE as it has matured as a discipline. While not entirely wrong, these beliefs are misleading and can lead to confusion about what CSE has to offer, thereby reducing the opportunities for effective CSE application.

Myth 1

The first myth is that *CSE should support the systems engineering community*. Because systems engineers are often in charge of IT projects, CSE is sometimes seen as a way to facilitate systems engineering activities by showing how to better fit the software tools with the nature of the cognitive work.

This is accurate but insufficient. CSE also can aid *project managers* by describing the kinds of cognitive functions to support. This improves the overall concept of the software tools, structures the team, and informs tradeoff decisions about cost, schedule, workload, manning, capability, and training. CSE also can support the *sponsors* by establishing cognitive requirements as acceptance criteria for the technology.

Thus, CSE can influence the design process at several points, not just the connection with systems engineers.

Myth 2

Another myth is that *CSE is a strategy*. Several practitioners have articulated CSE strategies to explain how people can apply CSE.

CSE strategies generally include the same types of activities. They identify the cognitive requirements for doing the work as well as barriers, types of errors, and critical cues and relationships and their design implications. However, these activities can be performed in various ways and sequences, and not all are needed on every project. Therefore, there's no "right" or "best" strategy.

For the most part, each CSE strategy is followed only by its close adherents, leading potential customers to wonder whether one strategy is better than another, what the implications of the differences between them are, and how to choose the right one for their needs. This confusion makes it less likely that an enterprise will apply CSE.

CSE aims to support, not minimize, cognitive requirements.

Myth 3

A third myth is that *CSE should minimize the user's cognitive requirements*. Some developers—especially those who want to automate tasks as much as possible—try to use CSE for this purpose. While some tasks benefit from automation, others need experienced decision-makers. CSE aims to support, not minimize, cognitive requirements; what needs to be minimized is cognitive overload.

Myth 4

Yet another myth is that *CSE is a way to get user opinions into the design process*. Almost everyone in the software engineering community appreciates the importance of user input throughout the development cycle. However, user opinions are often a source of problems. Users may advocate for specific types of solutions instead of explaining the nature and difficulty of their work. Worse, they frequently can't articulate the tacit knowledge that underlies their expertise. CSE can help discover this tacit knowledge and thereby clarify the cognitive requirements.

Myth 5

The final myth is that *CSE is essential to good system design*. Many well-designed systems, even those involving information technologies, didn't use CSE. Therefore, CSE isn't a critical path process; rather, it's a value-added technology.

CSE practitioners must make a case that the costs of the additional effort required are balanced by the benefits of improved design, increased chance of successful adoption, and the time and budget savings accrued through reduced design iterations.

However, even if CSE isn't essential to good system design, considering a system's cognitive aspects is. If developers believe they can adequately address these aspects without relying on CSE tools and approaches, we wish them luck.

CSE CONTRIBUTIONS

So what *does* CSE contribute to the design process? CSE practitioners help the design team understand the cognitive requirements of the work and how technology helps or hinders users in meeting those requirements. They broker design discussions by describing the impact of various design choices on the execution of cognitive work.

No one else on the team has this capability or responsibility. The designers are usually charged with shaping the system itself and thus don't always appreciate how users will interact with the system. They often feel that, being human, they have a human perspective, but they often miss key implications. Potential operators on the design team aren't always prepared to advocate for themselves—to reflect the tacit knowledge needed to do a good job.

CSE makes its contributions in several ways.

Expertise about system usage

CSE practitioners assess the context of system usage—the nature of the work and the environment in which it will occur. Many technology concepts seem feasible in a well-ordered situation but often turn out to be brittle as complexities arise. Drawing on their experience with comparable projects, CSE practitioners try to anticipate the kinds of unusual events, contingencies, and ripple effects users might confront.

Applying rather than managing information

CSE practitioners seek to ensure that information can be used to make sense of events. Software designers sometimes focus on information management as the end point and thus miss the importance of using information to make sense of events.

Cognitive measures

CSE practitioners recommend ways to evaluate how a technology supports the required cognitive functions. This includes describing scenarios and helping to define the cognitive attributes of test events. Measuring the operator's performance as well as the technology's lets the sponsor establish cognitive measures up front as test and evaluation criteria.

CSE tools

CSE practitioners use many *cognitive task analysis* methods to capture tacit knowledge (B. Crandall, G. Klein, and R.R. Hoffman, *Working Minds: A Practitioner's Guide to Cognitive Task Analysis,* MIT Press, 2006). Another common tool is the *abstraction-decomposition matrix,* which links user requirements to physical constraints (K.J. Vicente, *Cognitive Work Analysis: Towards Safe, Productive, and Healthy Computer-Based Work,* Lawrence Erlbaum, 1999).

CSE research

Some people wonder if CSE is ready for "prime time" and whether practitioners are stuck in endless research loops. However, CSE is being successfully applied today. And just as IT innovations proceed from ongoing research, so do CSE successes.

One common mistake designers make is to believe that people adhere to textbook decision strategies, set up decision tables, and engage in labor-intensive option comparisons. Several decades of research by CSE specialists have demonstrated otherwise: People use their experience with patterns to rapidly size up a situation, identify a plausible course of action, and evaluate this through a mental simulation process (G. Klein, *Sources of Power: How People Make Decisions,* MIT Press, 1998).

This recognition-primed decision strategy helped shape the design of information technologies in the Defense Advanced Research Project Agency's Command Post of the Future system, which the US Army first used in Iraq (http://en.wikipedia.org/wiki/Command_Post_of_the_Future).

Cognitive systems engineering imposes certain costs: adding a member to the design team and studying cognitive requirements. However, several benefits balance these costs, including

- increased system performance, which can only be achieved by understanding the cognitive requirements;
- reduced risk of additional iterations, project cancellations, or rejected deliverables;
- less time required to code systems by minimizing trial and error; and
- lower training, personnel, and manpower costs.

Successful integration of CSE into existing project activities such as modeling, simulation, and prototyping can mitigate these costs.

CSE supports the design process at different levels. It helps system engineers understand how tradeoff and design decisions will affect user performance. It helps program managers make tradeoffs throughout the life cycle. And it helps sponsors establish cognitive measures up front as the basis for test and evaluation criteria.

__Gary Klein__ is chief scientist of the Klein Associates Division of __Applied Research Associates__ Inc., an applied cognitive psychology provider based in Dayton, Ohio. Contact him at gary@decisionmaking.com.

__Steven Deal__ is vice president of __Deal Corp.__, an integration engineering company based in Yellow Springs, Ohio. Contact him at steven_deal@dealcorp.net.

__Sterling Wiggins__ is a senior scientist at __Applied Research Associates__. Contact him at sterling@decisionmaking.com.

Portions of this material are based upon work supported by the United States Air Force under contract no. FA8650-06-C-6638. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Air Force.

Editor: Richard G. Mathieu, Dept. of Computer Information Systems and Management Science, College of Business, James Madison Univ., Harrisonburg, VA; mathierg@jmu.edu