Applying Psychology to Design Better Human-Machine Systems: Crucial Goals for Cognitive Task Analysts

Cognitive Task Analysis
by Jan Maarten Schraagen, Susan F. Chipman, and Valerie L. Shalin (Eds.)
Mahwah, NJ: Erlbaum, 2000. 531 pp. ISBN 0-8058-3383-8. $99.95

Review by Marilyn Hughes Blackmon and Peter Polson

A completed cognitive task analysis (CTA) describes the skills and conceptual knowledge necessary to competently or expertly perform complex jobs or tasks, such as piloting a commercial aircraft. The primary motivation for doing a CTA is to improve the performance of human-machine systems by developing better training programs, developing tests to certify job competence, improving teamwork, or designing computer systems to support human workers.

For readers inexperienced in doing CTAs, Chapter 1 in *Cognitive Task Analysis* offers a succinct but very useful overview of how to do a modern prototypic CTA. In the preliminary phase of a CTA the analyst uses such techniques as unstructured interviews and tabletop analysis to grasp the general framework of the knowledge to be analyzed. The analyst then proceeds to the knowledge-elicitation phase, using such methods as structured interviews, thinking-out-loud protocols, and observations of subject-matter experts. In the export phase of the CTA the analyst translates the results of the CTA to software engineers, training designers, or assessment experts.

Cognitive Task Analysis captures and advances the CTA state of the art, including an annotated bibliography of previously published reviews of the CTA literature (Chapter 28) and a distinguished collection of new articles on CTA. Almost all the chapters in the book are independent case studies, each describing an application of at least one particular CTA method or tool to a specific job or task. The various case studies are written by leading-edge CTA analysts and collectively cover all the recent developments in CTA. Methods differ with respect to their focus (skills vs. conceptual knowledge), level of detail (enumeration of major tasks vs. highly detailed descriptions of the cognitive operations and physical actions required to perform each task), and theoretical foundations (atheoretical vs. being based on a modern theory of skill acquisition, notably the dominant theory of skill acquisition theory embedded within Anderson's ACT-R theory; Anderson & Lebiere, 1998).

The book resulted from a workshop, and its insightful, well-balanced introduction identifies several very important goals for the CTA field to work on over the next few years:

1. Develop a taxonomy of CTA methods.
2. Make CTA results more usable for designers.
3. Tie CTA to basic research and prompt further basic research.
4. Improve CTA of how the entire team performs.

Unfortunately, the volume reveals no evidence that the workshop participants hammered out agreement on how to accomplish these goals. Even the summary chapter (27) is a stand-alone personal reflection rather than a report of progress made at the workshop toward accomplishing these goals. Over the coming years the CTA field needs to pay attention to the goals articulated in the introduction, so each section of this review focuses on one of these goals and searches for common ground among workshop participants on how to accomplish that goal.

Develop a Taxonomy of CTA Methods

According to the introduction, the goal of building a taxonomy of CTA methods proved too ambitious for the workshop (Chapter 1). CTA analysts need to build on each other's work, but the disarray and diversity of CTA methods is liable to confuse anyone trying to select the most appropriate CTA method or methods to apply to a particular situation. Because skills are learned by doing and require a significant investment in practice, it is impossible to acquire skill performing any CTA method by simply reading a case study or a tutorial. The goal of the taxonomy should be to facilitate understanding of relevant CTAs and to enable CTA analysts to pick the method or methods most appropriate for doing a particular CTA without having to first learn how to do these methods.

It is important to distinguish CTA models (e.g., GOMS model, GOMS stands for Goals, Operators, Methods, and Selection rules) from CTA knowledge-elicitation methods (e.g., thinking-out-loud protocols). CTA analysts

JAN MAARTEN SCHRAAGEN, TNO Human Factors, Soesterberg, the Netherlands.

SUSAN F. CHIPMAN, Cognitive, Neural, and Biomolecular Science and Technology Division, Office of Naval Research, Arlington, Virginia.

VALERIE L. SHALIN, Department of Psychology, Wright State University.

MARILYN HUGHES BLACKMON and PETER POLSON, Institute of Cognitive Science, University of Colorado, Campus Box 344, Boulder, Colorado 80309-0344. E-mail: blackmon@psych.colorado.edu or ppolson@psych.colorado.edu

use knowledge-elicitation methods to collect data from subject-matter experts (people who have mastered doing the job or task), and then they can use the information collected to construct a CTA model of competence or expertise. Our tabulation of references cited in contributors' articles revealed a convergence of opinions (over 80 percent of the articles) toward some variant of goal hierarchy or task hierarchy for the CTA models. GOMS models were the most popular choice, cited by 69 percent of the computer system design articles (Part III) and 41 percent overall.

Knowledge-elicitation methods vary depending on the grain size of the model. In the initial iteration a CTA analyst can sketch a coarse-grained GOMS model to describe the high-level goals experts accomplish while doing the job. In subsequent iterations CTA analysts can gather more detailed data from subject-matter experts and then construct a medium-grained model or iterative series of increasingly detailed models. In the final iteration CTA analysts may build a very fine-grained model of competence, revealing all the subgoals, unit tasks, and operations that experts set to accomplish each of the high-level goals. Anderson (2002) has shown that important benefits accrue from paying attention to cognition at this fine-grained level.

Make CTA Results More Usable for Designers

CTAs are commissioned to improve the design of a computer system, training program or both, and good designs are generally contingent on the availability of boundary spanners to translate among CTA analysts, subject-matter experts, and designers. CTA analysts have developed effective methods for eliciting knowledge from subject-matter experts, but CTA analysts have been less successful in communicating with designers. If designers understand CTA results, they can use CTA results to design better computer systems or training programs.

Individual contributors to this volume offer a variety of success stories about effectively communicating CTA results to designers in particular contexts (e.g., Chapters 16, 18). GOMS models and other goal or task hierarchy models have proven effective for communicating with system designers (e.g., Chapters 11–13), designers of teamwork (Chapter 24), and designers of training programs and cognitive tutors (Chapters 1, 5, 11; Anderson, Corbett, Koedinger, & Pelletier, 1995). Coarse-grained models are appropriate for designing the basic functionality of a computer system or the curriculum structure for a training program (Kieras, 1996). Medium-grained models are for designing and testing a prototype. Fine-grained models can improve the usability of the fully implemented system design or training program, or assessment program.

There are better ways to communicate with designers. Potter, Roth, Woods, and Elm (Chapter 20) argue that CTA analysts must do much more than simply hand off their CTA to the software engineers. CTA analysts must have a deeper impact on system design by understanding software engineers' artifacts (object model, system requirements), capturing the design rationale, and developing scenarios. The real value of software to support CTA, they argue, will be in producing software engineering artifacts that directly feed the CTA results into the design of systems and training programs.

We urge CTA analysts to consider mapping CTA results to use case modeling that has been increasingly applied by object-oriented software designers (Constantine & Lockwood, 2001). Current books about how to write use cases (e.g., Armour & Miller, 2001; Cockburn, 2000) are strikingly devoid of any attention to the psychological validity of user goals, they do not cite any of the relevant psychological literature, and they totally neglect human information-processing constraints that could make it difficult or impossible for actual human beings to accomplish their goals via the system. Use case modeling has developed in isolation from parallel work by psychologists, but psychologists now have an exciting opportunity to feed CTA models into use case modeling, thereby translating CTA results into terms that software engineers understand. Scenarios (Rosson, 1999) capitalize on narrative structure to communicate effectively to designers and can be incorporated into use case models, or substituted for use case models, whenever informal models are more useful.

Tie CTA to Basic Research and Prompt Further Basic Research

The originators of hierarchical task analysis and human factors were mainly pragmatic and experience-based rather than theoretical, but a theoretical foundation for CTA has since emerged from basic research in cognitive psychology and cognitive science (Kieras, 1996). An alternate theoretical foundation comes from cognitive systems engineering (Rasmussen, Pejtersen, & Goodstein, 1994), which has produced some of the best CTAs of human performance and error in safety critical and other complex domains and systems (Chapters 6, 7, 20, 21) and should be mapped over into cognitive psychology wherever possible. Psychology theories are mostly microtheories developed to explain a single phenomenon, but cognitive architectures—ACT-R and EPIC are the ones most cited in this book—now provide a unified theory of cognition realized in a computational model that can simulate human performance. Cognitive architectures are becoming the preferred route for giving human–computer interaction research a solid grounding in cognitive psychology theory (Byrne, 2001, 2002; Gray, Young, & Kirschenbaum, 1997).

Several chapters in this book (notably Chapters 15, 22) advocate using cognitive architectures to build executable computational models capable of performing the task being analyzed and adapting to variations of the task. Implementing a CTA in a cognitive architecture makes it possible to predict human performance on the tasks described and test how well the model matches expert performance. This enables CTA analysts to evaluate alternative designs during early stages of the product development cycle, long before products are ready for actual user testing.

The practical demands for CTAs have revealed serious gaps in the theory and research base in several areas, and it is hoped that these gaps will spur further basic research and the integration of the basic research into the cognitive architectures used to test CTA models. The practical needs of CTAs call for basic research in several areas as follows: three-dimensional spatial representation and spatial thinking that are critical in many jobs, such as aviation and ship-based combat control (Chapters 4, 6, 17, 18, 20–25); event-drive expertise (Chapter 17), including adaptations to novel situations and responding to perception of meaningful patterns (Chapter 6, 7); multitasking, high cognitive load, and decision making under time pressure (Chapters 6, 7, 8, 20, 21); role of mental models (declarative knowledge) in relation to skills (procedural knowledge) in expert performance (Chapters 24–27); and how to train the capacity to solve novel problems, not just routine problems (Chapters 7, 17, 19).

Improve CTA of How Entire Team Performs

Most CTAs have described the jobs of individuals, but individual goals are in fact nested within the mission and goals of the organization and often within the goals of the team within which the individual works. Both the introduction and summary articles explicitly set a goal of adapting CTA methods to deal with teams, and the five articles in Part IV report CTAs devoted to CTAs of entire teams. For the bulk of the chapters in this book the theoretical foundation for CTA rests quite consistently on Anderson's cognitive architecture, GOMS, Hierarchical Task Analysis (HTA), or Rasmussen's work on cognitive systems engineering, but none of the five articles on team CTAs cite Anderson, GOMS, or Rasmussen, and only one cites HTA. Thus, the five articles in Part IV are poorly integrated with the rest of the CTA field. Distributed cognitition (Hollan, Hutchins, & Kirsh, 2000) and Groupware Task Analysis (van der Veer, van Welie, & Thorborg, 1999; van Welie, 2001) are promising recent developments that do rest on the shared theoretical foundation.

Implicit Goals of This Book Deserving Future Attention

In addition to the explicit goals already covered, several goals were articulated only in scattered chapters but are worthy of becoming the explicit focus of future books and workshops on CTA:

1. Reach agreement on how to acquire professional competence in CTA.
2. Insist on empirical evaluation of CTAs to verify their reliability and validity (Chapters 4, 15, 16, 20).
3. Make CTA more cost effective (Chapters 5, 9, 11, 14).
4. Find ways to efficiently update CTA results to accommodate rapid evolution of jobs (Chapter 22).

Conclusion

Cognitive Task Analysis is an important book. The book accurately presents the state of the CTA field, featuring representative contributions from a distinguished group of CTA professionals, and it explicitly or implicitly articulates the goals that must occupy the CTA profession at workshops and conferences over the next few years. The CTA field is fragmented, and we are disappointed that this particular workshop and book made little progress toward accomplishing the important goals that are explicitly and implicitly presented in the book. We hope readers of this book will pick up where the book left off, building on the common ground and actually accomplishing these important goals. ❑

References

Anderson, J. R. (2002). Spanning seven orders of magnitude: A challenge for cognitive modeling. *Cognitive Science, 26*, 85–112.

Anderson, J. R., Corbett, A. T., Koedinger, K. R., & Pelletier, R. (1995). Cognitive tutors: Lessons learned. *Journal of the Learning Sciences, 4*(2), 167–207.

Anderson, J. R., & Lebiere, C. (1998). *Atomic components of thought.* Mahwah, NJ: Erlbaum.

Armour, F., & Miller, G. (2001). *Advanced use case modeling: Software systems.* Reading, MA: Addison-Wesley.

Byrne, M. D. (2001). ACT-R/PM and menu selection: Applying a cognitive architecture to HCI. *International Journal of Human-Computer Studies, 55*, 41–84.

Byrne, M. D. (2002). Cognitive architecture. In J. Jacko & A. Sears (Eds.), *Handbook of Human-Computer Interaction.* Hillsdale, NJ: Erlbaum.

Cockburn, A. (2000). *Writing effective use cases.* Boston: Addison-Wesley.

Constantine, L. L., & Lockwood, L. A. D. (2001). Structure and style in use cases for user interface design. In M. van Harmelen (Ed.), *Object modeling and user interface design.* Boston: Addison-Wesley.

Gray, W. D., Young, R. M., & Kirschenbaum, S. (1997). Introduction to this special issue on cognitive architectures and human-computer interaction. *Computer-Human Interaction, 12*(4), 301–309.

Hollan, J., Hutchins, E., & Kirsh, D. (2000). Distributed cognition: Toward a new foundation for human-computer interaction research. *ACM Transactions on Computer-Human Interaction, 7*(2), 174–196.

Kieras, D. E. (1996). Task analysis and the design of functionality. In A. B. Tucker, Jr. (Ed.), *The computer science and engineering handbook* (pp. 1401–1423). Boca Raton, FL: CRC Press.

Rasmussen, J., Pejtersen, A. M., & Goodstein, L. P. (1994). *Cognitive systems engineering.* New York: Wiley.

Rosson, M. G. (1999). Integrating development of task and object models. *Communications of the ACM, 42*(1), 49–56.

van der Veer, G. C., van Welie, M., & Thorborg, D. (1999). Modeling complex processes in GTA. *Ergonomics, 42*(11), 1572–1587.

van Welie, M. (2001). *Task-based user interface design.* Unpublished doctoral dissertation, Free University, Amsterdam.