Byrne, M. (2005). Cognitive Architectures in HCI: Present Work and Future Directions. In Proceedings of the 11th International Conference on Human Computer Interaction. Las Vegas, NV.

Cognitive Architectures in HCI: Present Work and Future Directions

Michael D. Byrne

Department of Psychology, Rice University
6100 Main Street, MS-25
Houston, TX 77005 U.S.A.
byrne@acm.org

Abstract

This paper serves as the overview and introduction to a symposium of the same name. The symposium is made up of this introduction and six other papers on cognitive architectures in HCI. As many readers may not be familiar with cognitive architectures, a description of what cognitive architectures are is presented first. In an effort to be accessible to a wide audience, this description is fairly abstract. Once it is clear what is meant by a cognitive architecture, and what a model derived from such an architecture is, then the potential uses of such models in HCI efforts, both research and practical, can be laid out. While there is a great deal of promise, there are still challenges involved with using cognitive models in HCI, which are detailed in the third section. Finally, an overview of the other symposium papers is presented along with some orienting context.

1 What are Cognitive Architectures?

There are multiple ways to answer this question. In one sense, a cognitive architecture is a theoretical entity. That is, it is a broad theory of human cognition based on a wide selection of experimental data. Cognitive psychology is replete with theories about particular aspects of human cognition; a cognitive architecture is an attempt to build an integrated theory that encompasses a broad spectrum of what is known about human cognition and performance. Cognitive psychology has, for many years, been driven primarily by a divide-and-conquer approach, so cognitive architectures represent something of a departure from the norm. Many cognitive architectures are thus though of more properly as products of cognitive science rather than strictly cognitive psychology; there has been and continues to be a great deal of borrowing of techniques and ideas from artificial intelligence and other areas of computer science.

But a cognitive architecture is more than just a theory of cognition. It is, as defined by Young (Gray, Young, & Kirschenbaum, 1997; Ritter & Young, 2001) an embodiment of "a scientific hypothesis about those aspects of human cognition that are relatively constant over time and relatively independent of task." That is, it is an attempt to describe those aspects of the human cognitive system that are more or less universal, both across and within individuals. Thus, a cognitive architecture alone is generally not able to describe human performance on any particular task; it must be given knowledge about how to do the task. Generally speaking, a *model* of a task in a cognitive architecture (generally termed a "cognitive model") consists of the architecture and the requisite knowledge to perform the specified task. This knowledge is typically based on a thorough task analysis of the target activity being modeled.

Finally, a cognitive architecture is a piece of executable software. It is code, written by a programmer (or, more generally, by multiple programmers). This is another critical way in which cognitive architectures differ from most theories in cognitive psychology; most cognitive architectures produce not only a prediction about performance, but in fact output actual performance. That is, they produce a timestamped sequence of actions (e.g., mouse clicks, eye movements) that can be compared to actual human performance on a task. The time stamps on the actions mean that architectures produce models that are quantitative. A model could thus predict that not only is task A faster than task B, but that it is 2.2 seconds or 20% faster. This has numerous engineering implications, which are discussed further in section 2 as well as in Byrne and Gray (2003). Another important implication of this is that the

knowledge which has to be supplied to an architecture generally has to be supplied in the language of the architecture. Structuring knowledge in this form is distinctly like programming, so architecture-based modelers typically have strong programming skills.

As cognitive architectures progress and models of more and more complex and interactive tasks are built, it is increasingly common that the architecture is connected to a complex simulation of the environment in which the task is performed. In some cases, the cognitive architecture interacts directly with the actual software that humans use to perform the task. In other cases some form of connecting software must be constructed. Overall, a model of a task generally has three components: the architecture, task knowledge, and a dynamic task environment with which the model interacts. The output of this system is, as mentioned, a timespamped behavior stream, as depicted in Figure 1.

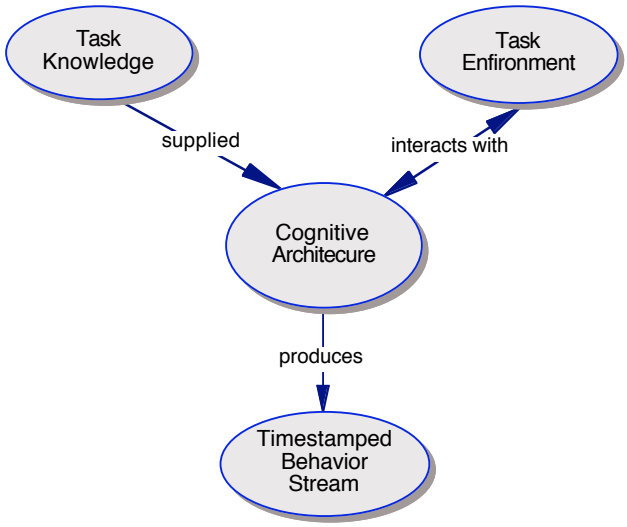

Figure 1: Structure of a model based on a cognitive architecture

Presently, cognitive architectures are primarily research tools housed in academic laboratories; however, this is changing. There are now numerous consulting/technical companies who, among other things, employ cognitive architectures in their work and even provide models developed with cognitive architectures to their clients. The Lebiere, et al. paper in this symposium is an example of a non-academic use of a cognitive architecture. Furthermore, work is being done with several architectures to reduce the usually substantial learning curve required to master the software; more will be said on this in sections 3 and 4.

2 Cognitive Architectures and HCI

It may not be immediately apparent what research on cognitive architectures has to offer HCI. On the surface, this kind of research often looks like highly theoretical cognitive science, particularly when the researchers get embroiled in debates about the ramifications of low-level features of the architecture (see also the Kieras paper in this symposium for how such issues affect HCI-level models). However, as it turns out, cognitive architectures may in fact be some of the most HCI-relevant cognitive science work available. There are multiple roles which cognitive architectures can fill in HCI research and practice, and in the other direction, there are important questions the HCI domain poses for research on cognitive architectures.

2.1 Cognitive Models as Design and Evaluation Aids

When an electrical or aerospace engineer designs, or evaluates a design, she has at her disposal a wide variety of tools, both qualitative and quantitative, to assist with the task. These engineering disciplines have a strong quantitative science base at their core which enables such tools. The same has generally not been true for the usability engineer. The usability engineer generally has to rely on some combination of guidelines, experience-based heuristics, direct empirical evaluation (e.g., a usability test), and other less-formal methods. These are certainly useful techniques that have added a great deal to the usability of a huge array of computer systems, and more traditional engineering practitioners also employ such techniques. However, the quantitative formal methods available to other engineers have a great deal of value and are generally considered central to the practice, so much so that undergraduate curricula in such areas typically requires multiple courses in the science and mathematical base before students are even permitted to take medium- and upper-level courses in their engineering discipline. In many ways, this is because the science base for these disciplines is considered more advanced than it is for the usability engineer. However, this is changing.

In *The Psychology of Human-Computer Interaction*, Card, Moran and Newell (1983) laid out a survey of the science base and concluded that many HCI design and evaluation problems could, in fact, be supported by quantitative methodology. In the last twenty-plus years, this science base has grown and current syntheses of the science base are now best represented in cognitive architectures. Thus, models based on cognitive architectures can serve to aid the usability engineer in design and evaluation. As mentioned, architecture-based models generally produce as output a time-stamped stream of actions, both overt (e.g., keystrokes and mouse clicks) and covert (e.g., retrievals from long-term memory). Thus, depending on details and assumptions made, cognitive models are able to predict user performance on a variety of measures including task times, learning rates and/or difficulties, eye scan paths, transfer, and error rates. Obviously, many of these are exactly the kinds of things usability engineers expect to get from usability tests. Furthermore, these models can often provide insights that usability tests cannot provide. For example, while it may be possible to show with a usability test that interface A produces superior task execution times relative to interface B, it is often unclear the reasons underlying the source of the difference. Cognitive models can often make such reasons clear (for an excellent example, see Kieras, Meyer, & Ballas, 2001). Furthermore, the predictions made by such models are generally quantitative. While on some non-model-based *a priori* basis it may be possible to predict that interface A will outperform interface B, it is rare that other methods provide an estimate of *how much* better A will be.

This is not to suggest that cognitive models will entirely supplant usability tests; real empirical tests both in the laboratory and in the field are still conducted in other engineering disciplines as well. However, cognitive models can help the usability engineer focus usability tests on features or tasks likely to be crucial and can help rule out early design alternatives, thereby reducing the number of cycles of usability testing and possibly even the number of users needed. This is particularly attractive when the user population is very small or very difficult to access due to specialization or expense, or when the tasks or situations required in the tests are dangerous or expensive. For example, testing commercial airline pilots in flight is quite challenging because pilots are difficult to recruit and their time is expensive, outfitting commercial jetliners with new equipment requires considerable time and engineering effort, and poor results can have fatal consequences. Some of these problems can be overcome by the use of simulators rather than real cockpits, but high-fidelity simulators are themselves very expensive. Similar issues come up when designing or evaluating systems for use by medical professionals (particularly advanced specialists); one can imagine any number of special populations that would raise some or all (or even more) such issues.

Now, it is certainly the goal of related techniques such as GOMS analysis (see John & Kieras, 1996) or Cognitive Walkthrough (Polson, Lewis, Reiman, & Wharton, 1992) to make predictions, often

quantitative, of many of the same things. As it turns out, these techniques were originally grounded in the same ideas as those that underlie prominent cognitive architectures and are essentially abstractions of the relevant architectures for HCI purposes. Furthermore, cognitive models provide things that such analyses do not. Cognitive models are executable and generative, which means they produce not just global execution times, but they actually generate behavior. A GOMS analysis, on the other hand, is a description of the procedural knowledge the user has to have and the sequence of actions that must be performed to accomplish a specific task instance, while the equivalent computational model actually generates the behaviors, often in real time or faster. Equally importantly, computational models have the capacity to be reactive in real time. So, while it may be possible to construct a GOMS model that describes the knowledge necessary and the time it will take an operator to classify a new object on an air traffic controller's screen, a paper-and-pencil GOMS model cannot actually execute the procedure in response to the appearance of such an object. However, a running computational model can. (It should be noted that David Kieras has done considerable work on a tool called GLEAN which in essence does allow for the execution of GOMS models; see his paper in this symposium for more information.)

2.2 Cognitive Models in Lieu of Human Users

The fact that these models are executable in real time (or faster) means they have a number of other HCI-relevant applications that may not be immediately apparent. One such use is in intelligent tutoring systems (ITSs). Consider the Lisp tutor (Anderson, Conrad, & Corbett, 1989). This tutoring system contained an architecture-based running computational model of the knowledge necessary to implement the relevant Lisp functions, and a module for assessing which pieces of this knowledge were mastered by the student. Because the model was executable, it could predict what action the student would take if the student had correct knowledge of how to solve the problem. When the student took a different action, this told the ITS that the student was missing one or more relevant pieces of knowledge. The student could then be given feedback about what knowledge was missing or incomplete, and problems which exercised this knowledge could be selected by the ITS for further practice by the student. By identifying students' knowledge, and the gaps in that knowledge, it was possible to generate more effective educational experiences. Problems that contained knowledge the student had already mastered could be avoided, to not bore the student with things they already knew. This freed up the student to concentrate on the material they had not yet mastered, resulting in improved learning (Anderson, et al., 1989). While the Lisp tutor is an old research system, ITSs based on the same underlying cognitive architecture with the same essential methodology have been developed for more pressing educational needs such as high school algebra and geometry and are now sold commercially (see http://www.carnegielearning.com for more information).

There is another important and HCI-relevant application for high-fidelity cognitive models: populating simulations or simulated worlds. For example, training a tank operator is expensive, even in a simulator, because that trainee needs to face realistic opposition. Realistic opposition consists of other trained tank operators, so training one person requires taking several trained operators away from their normal duties (i.e., operating tanks on real missions). This is difficult and expensive. If, however, the other operators could be simulated realistically, then the trainee could face opposition that would have useful training value, without having to remove already-trained operators from their duties. There are many training situations like this, where the only way to train someone is to involve multiple human experts who must all be taken away from their regular jobs. However, the need for expensive experts can potentially be eliminated (or at least reduced) by using architecturally-based cognitive models in place of the human experts. The U.S. military has already started to experiment with just such a scenario (Jones, Laird, Nielsen, Coulter, Kenny, & Koss, 1999). There are other domains besides training where having realistic opponents is desirable, such as video games. Besides things like texture-mapped 3D graphics, one of the features often used to sell games is network play. This enables players to engage opponents whose capabilities are more comparable to their own than typical computer-generated opponents. However, even

with network play, it is not always possible for a gamer to find an appropriate opponent. If the computer generated opponent were a more high-fidelity simulation of a human in terms of cognitive and perceptual-motor capabilities, then video game players would have no difficulty finding appropriate opponents without relying on network play. While this might not be the most scientifically interesting use of cognitive architectures, it seems inevitable that cognitive architectures will be used in this way.

This is by no means a complete list; there are many other potential uses for cognitive architectures in HCI. However, this should provide enough so that the value (or at least the potential value) of such endeavors to the HCI community is apparent. This is not an entirely promissory note, either, as there have been many successful applications of cognitive architectures in HCI beyond those mentioned here. Most issues of journals such as *Human-Computer Interaction* and each of the proceedings of the ACM SIGCHI conference contain papers which use architecture-based cognitive models. There was a special section of the journal *Human-Computer Interaction* devoted to such models in 1997 (see Gray, Young, & Kirschenbaum, 1997), a similar special issue of the *International Journal of Human-Computer Studies* in 2001 (see Ritter & Young, 2001), and a special section of the journal *Human Factors* in 2003 (see Byrne & Gray, 2003) that contained several papers based on such models.

2.3 HCI Challenges for Cognitive Science

As it turns out, the relationship between cognitive architectures and HCI is reciprocal. Just as cognitive architectures can contribute positively to research and practice in HCI, tackling HCI problems is useful and informative for research in cognitive architectures. As mentioned, cognitive science has taken a divide-and-conquer approach to the study of human cognition; models and theories in one domain of study (e.g., memory) often make little or no contact with models and theories in other domains (e.g., vision). Thus, the tasks used in many laboratories are often highly artificial, stripping away a great deal of other influences so the researcher can focus on their particular domain. People in HCI do not have this luxury. Even fairly straightforward Web browsing requires users to engage a wide range of cognitive, perceptual, and motor capabilities. Thus, HCI is an excellent domain to push on the integration capabilities of cognitive architectures. Furthermore, HCI is a very natural domain for people who construct cognitive architectures. Such people often have not only extensive training in cognitive psychology or computer science, but generally have a fair amount of cross-training in the other discipline. Since these are two of the central disciplines reflected in HCI, there tends to be a substantial overlap in interests between people in cognitive architectures and people in HCI. And since most architecture builders also spend inordinate amounts of time engaged with their computers, they are directly affetcted by advances in HCI.

3 Open Issues and Future Work

While the track record of cognitive architectures in HCI is impressive, they are in many ways still works in progress. There are many challenges that have yet to be addressed by cognitive architectures. The good news is that research has been initiated on many of these issues and in many cases early reports of progress are promising.

3.1 Theoretical Coverage

While cognitive architectures strive to be comprehensive, incorporating the full range of perception, cognition, and motor control is obviously a significant challenge. Unsurprisingly, there are a great many gaps in what has been covered by cognitive architectures. Generally speaking, perception and motor control are only modeled at a very abstract level; these systems generally do not cover detailed aspects of how light is turned into perception of objects and surfaces, nor are the details of complex motor movements modeled (usually only the time course is modeled). Even within the realm of cognition, there

are gaps. For instance, there has been little work in domains such as probabilistic reasoning, many aspects of metacognition, and individual differences (though see Daily, Lovett, and Reder, 2001 for an impressive account of individual differences in working memory capacity). However, with multiple architectures being driven to increasingly wider and wider ranges of tasks, these gaps are gradually being filled in.

Another aspect of cognitive architectures that does not quite meet the full range of capabilities necessary to model HCI tasks is that such architectures are almost entirely cognitive; they model primarily aspects of thinking, and just enough perception and motor control to support that thinking. They generally do not take into account factors like affect and social influence. However, as affect and emotion have been topics which have received considerable attention lately in the HCI community, work on integrating affect with cognitive architectures is emerging rapidly (Gratch & Marsella, 2004; Norman, Ortony, & Revelle, in press). Along similar lines, models developed in cognitive architectures do not express aesthetic or subjective preferences, something real users obviously do (often vociferously!).

Another set of factors which has not yet received adequate attention are factors like fatigue, stress, sleep deprivation, and the like. While these certainly may carry with them affective or emotional components, such factors also have direct effects on human cognition and performance. Unfortunately, cognitive architectures have generally not been used to model the effects of such moderators. Again, though, research into how such effects can be modeled with cognitive architectures is underway (for an example see Ritter, Reifers, Klein, Quigley, & Schoelles, 2004).

Lastly, most cognitive modeling has been aimed at modeling the behavior of a single user at a time. Clearly, many tasks of great interest to HCI researchers and practitioners involve groups or teams of users. While there is nothing in principle which prevents the use of cognitive architectures to construct multiple models and have those models interact with one another, such usage has not been the norm and it is not clear the extent to which these would really capture the richness of human social interaction. However, there has been some promising work on this front as well presented by Kieras and Santoro (2004), though that work did not delve deeply into social factors.

3.2 Technical and Practical Hurdles

There are other barriers to the widespread adoption of cognitive architectures in HCI beyond theoretical coverage. Just as with those issues, however, progress is being made on a number of fronts. Probably the largest problem in principle and in practice is the knowledge engineering problem. Consider Figure 1 again. One of the things that must be supplied to a cognitive architecture as one constructs a model is the knowledge about the task possessed by the person or people being modeled. In some cases, this is not a huge hurdle. In a huge array of laboratory psychology experiments, the tasks are both simple and novel and people have little knowledge to bring to bear. This is almost certainly true of some user interfaces as well; the expectation is that users will come to the interface with little or no direct experience. Automated teller machines (ATMs) and many information kiosks are of this general type. However, even in those cases, users generally do come to the task with a surprisingly substantial amount of knowledge, even if that knowledge is not specific to that particular interface. For example, novice users of an ATM are assumed to be able to read and to understand the mapping between instructions printed on the display and the actions they are to take. While this may seem like fairly trivial knowledge, it is so implicit for most people that the subtleties involved are easy to under-appreciate. What cognitive architectures need for situations like this are large “background knowledge” bases along with accurate models of human learning processes. Work to address the issue of learning from simple instructions was initiated with Soar architecture some years ago (see Lewis, Newell, & Polk, 1989) and more recently, this has become a major focus of some of the researchers in the ACT-R community (Anderson, Bothell, Byrne, Douglass, Lebiere, & Qin, 2004).

However, even if learning simple novel tasks from instructions becomes straightforward, this will not solve the knowledge engineering problem in general. This is because expert human users have a great deal of knowledge, earned over years of experience with their task, which cannot be easily captured by a simple set of instructions. Many commercial aircraft pilots, for example, have logged literally thousands of hours in the cockpit. It is unfortunate that the kinds of populations that are difficult to recruit as participants in usability tests also tend to be the kinds of populations for which the knowledge engineering problem is also the most severe. There is no in-principle solution for this problem; human experts will always have a great deal of knowledge and it will always be challenging to extract this knowledge and put it into a form which can be used by a cognitive architecture. However, in many cases it may be worth this effort if it reduces the need for extensive testing using such experts.

Another issue that arises with some regularity when dealing with cognitive architectures is the other part of the diagram in Figure 1, which is the connection between the architecture and the task environment. An excellent summary of this problem can be found in Ritter, Baxter, Jones, and Young (2000). In brief, for very simple environments, this can be a very simple representation of the environment written in the same simulation language as the architecture itself. In other cases, where that environment is both complex and dynamic, this can be a very difficult problem. In general, most user interfaces are "closed" pieces software with no built-in support for supplying a cognitive model with the information it needs for perception (i.e., what is on the screen where) or accepting input from a model. Somehow, the interface and the model must be connected. Work on making this easier is ongoing (see Section 4) and there is reason to be optimistic that this hurdle can be straightforwardly overcome, but for the time being this is still often a problem for cognitive modelers.

Finally, there is the issue of the usability of cognitive architectures themselves. These systems tend to be large, complex, and unwieldy pieces of software, generally with limited documentation. This does not make them especially approachable by anyone other than those researchers who have extensive experience with a particular architecture. There is indeed a certain irony to this situation. However, work is underway to help with this problem as well. One very exciting line of work is presented in John, Prevas, Salvucci, & Koedinger (2004). They have combined an HTML graphical interface builder with a programming-by-demonstration interpreter and a system for automatically synthesizing ACT-R models from simple execution traces. This is done to make the generation of cognitive models straightforward for certain classes of interfaces and tasks. While space considerations prevent a more detailed presentation of this work, the promise held here is substantial.

Despite these and other challenges beyond the scope of this paper, there are many questions, both research questions and practical questions, for which cognitive architectures are well suited.

4 Recent Advances Represented in this Symposium

While numerous advances in a variety of areas were described in the previous section, this symposium serves to highlight a number of other research programs in cognitive architectures with a particularly strong connection to the HCI world. The remainder of this paper serves as an introduction and overview of that work in an effort to frame the other papers in the larger context of cognitive architectures and HCI. Most of this work touches on issues that were raised in the previous section

For instance, one of the issues raised is the representation of background knowledge, not specific to any interface, held by most users. This is particularly pertinent to understanding and predicting the behavior of users in environments where the basic mechanics (i.e., pointing and clicking) of the interface are straightforward, but the information environment faced by users most definitely is not. The paradigm example for such an interface right now is the World-Wide Web. With even moderately well-constructed Web pages, the mechanics of pointing and clicking links to navigate the Web are straightforward.

However, the problem of choosing *which* link to follow in order to find particular pieces of information can be a difficult one for users, and an even more challenging problem to predict with any kind of model. This is because this kind of behavior is largely a function of the huge amount of knowledge users bring with them to the task. Not only is there a lot of knowledge to represent, but that knowledge is highly-structured and semantically rich. Even a few years ago, representation of such knowledge in a cognitive model was a challenge that was hard to fathom. However, an enormous amount of progress has been made on this problem in recent years, which is represented in this symposium by two papers, one by Kitajima, Blackmon, and Polson and the other by Pirolli, Fu, Chi, and Farahat. These papers differ in many important and interesting ways, but both showcase how cognitive architectures have been scaled up to address the semantic demands of the Web.

Another issue raised in the previous section is the issue of knowledge engineering. One of the things which makes knowledge engineering for cognitive architectures so labor-intensive is that the knowledge typically has to be specified and debugged at a very detailed level, down to the level of individual eye movements. While the problem of understanding what people, particularly experts, know about a task is not particularly easy to solve, there are multiple researchers who are actively looking at ways to make the representation of this knowledge easier. The paper in the symposium by Lebiere, Archer, Warwick and Schunk is an example of such an approach. This paper describes their work on unifying a detailed cognitive architecture, ACT-R, with a higher-level task network simulation tool, IMPRINT. Task network tools use a task representation that is at a much higher level of abstraction than those typically used in cognitive architectures. Such models have been a part of the human factors toolbox for some time, so this work represents the unification of two important human performance modeling traditions.

Another approach to this problem is to simply forego the level of detail typically found in cognitive architectures. For many HCI-level tasks, such detail is not necessary and forces the model-builder to confront issues on or outside the boundaries of the science base. While resolution of those issues is ultimately important for both scientific and practical reasons, it is not in the best interests of the HCI practitioner to have to confront those issues on a routine basis. Such is the argument presented in the paper by Kieras. He outlines a two-pronged approach to cognitive modeling in HCI, a “low fidelity” approach based on a tool called GLEAN aimed at keeping such details away from modelers and a “high fidelity” approach based on an architecture called EPIC aimed at addressing these detailed issues in the longer term. An interesting alternative middle ground can be found in a paper by St. Amant, Freed, and Ritter (2005) in which they present a tool, G2A, which takes a GLEAN-level specification and translates it directly into a model basd in the ACT-R architecture, which is at approximately the same level of analysis as EPIC.

However, this is not the problem at which the symposium paper by St. Amant, Reidl, and Ritter is aimed. Their symposium paper concerns the problem of how to connect a cognitive architecture to a closed software systems that were not designed with communication with a cognitive model in mind. In particular, they describe a system called SegMan that applies image processing techniques to raw bitmaps to allow cognitive models to “see” and manipulate arbitrary Windows applications, without having to modify the Windows application in any way. The promise of this tool is tremendous and could potentially go a long way toward solving the software integration problems faced by cognitive modelers in a wide variety of domains.

Finally, there is the paper by Vera, Howes, Lewis, Tollinger, Eng, and Richardson. This represents a different approach to cognitive modeling, one based on enumerating the constraints on human performance and then reasoning over those constraints to predict expert human behavior. This is not exactly a cognitive architecture as defined in section 1 of this paper, but it shares a great deal of the same intellectual tradition and aims to produce quantitative models of human performance with HCI

implications. Their approach is both novel and interesting, and may ultimately provide insights which are later incorporated into the kinds of cognitive architectures described in the other papers.

This is by no means a complete picture of HCI-relevant research in cognitive architectures. There are numerous other architectures, research questions, and task domains being explored by cognitive modelers, many of which have either near-term or long-term connections to research and practice in HCI. A complete survey of all such projects would be far beyond the scope of this paper and this symposium. What should be evident is that this is a vibrant research area which will have an increasing impact on HCI as it grows in scientific and technical scope.

Acknowledgments

I would like to thank all of the symposium authors for their excellent contributions to the session. I would also like to acknowledge the financial support of the Office of Naval Research under grant number N00014-03-1-0094 and the National Aeronautics and Space Administration under grant number NCC2-1219. The views and conclusions contained herein are those of the authors 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.

References

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

Anderson, J. R., Conrad, F. G., & Corbett, A. T. (1989). Skill acquisition and the LISP tutor. *Cognitive Science, 13*(4), 467-505.

Byrne, M. D., & Gray, W. D. (2003). Returning human factors to an engineering discipline: Expanding the science base through a new generation of quantitative methods—preface to the special section. *Human Factors, 45*, 1-4.

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

Daily, L. Z., Lovett, M. C., & Reder, L. M. (2001). Modeling individual differences in working memory performance: A source activation account. *Cognitive Science, 25*, 315–353.

Gratch, J., & Marsella, S. (2004). A domain-independent framework for modeling emotion. *Journal of Cognitive Systems Research, 5*, 269-306.

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

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.

John, B. E., Prevas, K., Salvucci, D. D., & Koedinger, K. (2004). Predictive human performance modeling made easy. In *Human factors in computing systems: Proceedings of CHI 2004* (pp. 455-462). New York: ACM.

Jones, R. M., Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P., & Koss, F. V. (1999). Automated intelligent pilots for combat flight simulation. *AI Magazine, 20*(1), 27-41.

Kieras, D. E., Meyer, D. E., & Ballas, J. A. (2001). Towards demystification of direct manipulation: Cognitive modeling charts the gulf of execution. In *Proceedings of ACM CHI 01 Conference on Human Factors in Computing Systems* (pp. 128-135). New York: ACM.

Kieras, D. E., & Santoro, T. P. (2004). Computational GOMS modeling of a complex team task: Lessons learned. In *Human Factors in Computing Systems: Proceedings of CHI 2004* (pp. 97-104). New York: ACM.

Lewis, R. L., Newell, A., & Polk, T. A. (1989). Toward a Soar theory of taking instructions for immediate reasoning tasks. In *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society* (pp. 514-521). Hillsdale, NJ: Lawrence Erlbaum Associates.

Norman, D. A., Ortony, A., & Revelle, W. (in press). Effective functioning: A three-level model of affect, behavior, and cognition. In J. M. Fellous & M. A. Arbib (Eds.), *Who needs emotions? The brain meets the machine*. New York: Oxford University Press.

Polson, P. G., Lewis, C., Reiman, J., & Wharton, C. (1992). Cognitive walkthroughs: A method for theory-based evaluation of user interfaces. *International Journal of Man-machine Studies, 36*, 741-773.

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

Ritter, F. E., Reifers, A., Klein, L. C., Quigley, K., & Schoelles, M. (2004). Using cognitive modeling to study behavior moderators: Pre-task appraisal and anxiety. In *Proceedings of the Human Factors and Ergonomics Society 48th Annual Meeting* (pp. 2121-2125). Santa Monica, CA: Human Factors and Ergonomics Society.

Ritter, F. E., & Young, R. M. (2001). Embodied models as simulated users: Introduction to this special issue on using cognitive models to improve interface design. *International Journal of Human-Computer Studies, 55*, 1-14.

St. Amant, R., Freed, A. R., & Ritter, F. E. (2005). Specifying ACT-R models of user interaction with a GOMS language. *Cognitive Systems Research, 6*, 71-88.