

# Architectural Building Blocks as the Locus of Adaptive Behavior Selection

**Alonso H. Vera (alonso.vera@nasa.gov)**

Carnegie Mellon University & NASA Ames Research Center, Moffett Field, CA USA

**Irene Tollinger (irene.tollinger@nasa.gov)**

**Katherine Eng (keng@mail.arc.nasa.gov)**

NASA Ames Research Center, Moffett Field, CA USA

**Richard L. Lewis (rickl@umich.edu)**

Department of Psychology, University of Michigan, Ann Arbor, MI USA

**Andrew Howes (howesa@mac.com)**

School of Informatics, University of Manchester, Manchester UK

## Abstract

Historically CPM-GOMS has been used to predict total time for long stretches of behavior. In “Milliseconds Matter”, Gray and Boehm-Davis (2000) use CPM-GOMS to develop microstrategy variants with subtly different internal structure to explain differences observed in empirical data collected. They argue for microstrategies as the basic unit of adaptive behavior selection. While the microstrategies developed provide a good fit to the data, there is neither direct evidence for microstrategies as compared to other possible constructs nor an explicit statement of the theory underlying their construction. While the use of CPM-GOMS as an explanatory mechanism is a substantial advance, microstrategies have theoretical and practical limitations in terms of: microstrategies functioning as cognitive units, composition and structure, and dependency constraints. An alternative construct called an Architectural Process Cascade (APC) is proposed as the locus of adaptive behavior selection. An APC-based model of the Gray and Boehm-Davis button study task is presented to address the limitations of microstrategies.

## Introduction

Starting with Card, Moran, and Newell (1983), GOMS methods were developed to generate *a priori* predictions of human performance on human-computer interaction tasks. The CPM-GOMS method (Cognitive Perceptual Motor GOMS) in particular is useful in modeling routine skilled performance or “extreme expertise” (John & Kieras, 1996) at a high resolution. The analysis decomposes interactive activity into basic behaviors (clicking buttons, typing words, etc.), which are then expressed as concurrent, interleaving streams of cognitive, motor, and perceptual operators.

Historically, the GOMS methods have served as engineering models, producing approximations of performance at the level of detail chosen by the modeler, to influence system design and evaluation rather than as theoretical, explanatory models (Gray, John & Atwood, 1993; John, 1990). As Card, Moran, and Newell (1983) state, engineering models are “intended to help us remember facts and predict user-computer interaction rather than intended as a statement of what is really in the head” (p. 24).

Gray and Boehm-Davis (2000) reverse this practice by using CPM-GOMS in “Milliseconds Matter” to provide an account of what is happening in the head with respect to

behavior variants uncovered in an experimental study. In what is a significant contribution, they demonstrate that CPM-GOMS has the potential to be used as an explanatory tool. The demonstration that people make microstrategy-level adjustments is an important contribution, and in this paper we further explore that hypothesis with their data.

Gray and Boehm-Davis found an average 150-millisecond difference in task time between two different subtasks of their button task. Both subtasks require clicking a target not initially visible. In one case, the target location is known while in the other, the target location is not. Based on this study and related work (O’Hara & Payne, 1999), they argue that users optimize by selecting the most efficient microstrategy. Microstrategies are expressed as groupings of cognitive, motor, and perceptual operators into behavioral units, such as *move-click* and *click-move*.

Microstrategies are basically the same level of analysis as templates in the CPM-GOMS method (e.g., John & Kieras, 1996). CPM-GOMS templates have proven useful as a modeling method for making predictions about the time course of behavior (Gray, et al., 1993). However, this does not constitute evidence that microstrategies actually represent the strategic units selected during task execution. While microstrategies may be a useful construct for reasoning about behavior *post hoc*, there must be a *theory* of behavior composition in order to provide an explanatory account of what occurs in the head during task execution.

We have developed models using Gray and Boehm-Davis’s microstrategies within Apex-CPM and the CORE architecture over the course of the last several years (John, et al., 2002; Vera, et al., 2004). The work described here is motivated by a desire to extend the CPM-GOMS approach presented in “Milliseconds Matter” as a consequence of working with their microstrategies at a very detailed level.

Over the last decade, CPM-GOMS practitioners have struggled to develop a coherent theory of behavior composition from microstrategies as evidenced by the difficulty in teaching microstrategy interleaving in courses and tutorials (B. E. John, personal communication, February 9, 2005). Similarly, the present authors and others (e.g. Vera, et al., in press) have worked toward the generation of such a theory. Although there has been substantial success

in developing computational methods to support *modeling* at the template level, it has proven difficult to generate a coherent *theory* of behavior composition to account for dynamic task execution (i.e. what is going on in the head). Therefore, the effort here pursues a different level of analysis to describe adaptive behavior composition.

In this paper we propose an alternative to microstrategies that preserves their primary strength – the potential to explain structure and variation in interactive cognitive skill – but does so in a way that is theoretically more constrained, and hence more explanatory. We mean more constrained in a very precise sense: we believe that the new approach significantly reduces the set of possible models to describe a given behavior, and thus results in more consistent *a priori* prediction of human performance. There are three features of our approach that distinguish it from the microstrategy method, and that together provide this constraint. (1) The basic unit of analysis is at a higher level than individual operators in CPM-GOMS models, but at a lower level than microstrategies. These units, called Architectural Process Cascades (APCs), are derived by identifying the smallest architecturally bound sets of communicating operators. (2) There is a richer ontology of operator relations: rather than temporal dependencies, there are resource-consuming information flows, and a distinction is made between those that are architecturally required versus a flexible response to task constraints. (3) The composition of APCs to realize task objectives is a semi-automated process of optimized scheduling over explicit objective functions, such as minimize time (Howes, et al., 2004).

In short, we are attempting to make explicit some of the theoretical assumptions that we believe guided the creation of microstrategies, introduce a more expressive ontology of operator relations for expressing that theory, carve behavior more naturally at its architectural joints, and support behavior composition through automated theory application.

In the remainder of the paper we first identify a number of issues with the microstrategy approach that we believe indicate an underlying problem: microstrategies are a highly under-constrained theoretical framework. Next, we lay out the principles of our new approach and describe how it addresses the problems identified with microstrategies.

Finally, we provide an alternative model of the Gray and Boehm-Davis *home-to-target* task, and describe how it was computationally derived from the application of APCs.

### CPM-GOMS Templates as Microstrategies

Figure 1 represents the microstrategies in the button study’s *home-to-target* task from “Milliseconds Matter”. Boxes represent operators, numbers indicate operator durations taken from psychological literature, and lines depict temporal dependencies between operators. The path of lines and boxes in bold represent the chain of dependent operators with the longest duration, or the *critical path*. The critical path determines the time required to execute the activity. This section covers microstrategies in more detail focusing on the following issues: microstrategies as cognitive units, composition and structure, and dependency relations.

### Microstrategies as Cognitive Units

Although Gray and Boehm-Davis explicitly note that they are not attempting to provide a theory of the control structure required for microstrategy selection, they nevertheless suggest that “users must have acquired selection rules for when to use one microstrategy rather than another” (p. 333). Similarly, they argue the specific details of microstrategies should be taken seriously as what is going on inside the head: “...the [microstrategy] level of description will make clear to psychological researchers the importance of integrating hitherto disparate descriptions of low-level cognitive, perceptual, and motor operations into theories of embodied cognition” (p. 334).

However, the rules and mechanisms for selecting between variants of microstrategies can be ambiguous and inaccurately describe strategic selection. For example, in the button study task, the interaction requires precision to select the target, and therefore a *slow-move-click*, which contains a verification of the cursor at the target, is selected. However, this selection at the microstrategy level is problematic in that it forces the model to decide on the use of the extra verification before the target has even been perceived. One can imagine a similar button task in which both the location and size of the target location are unknown.

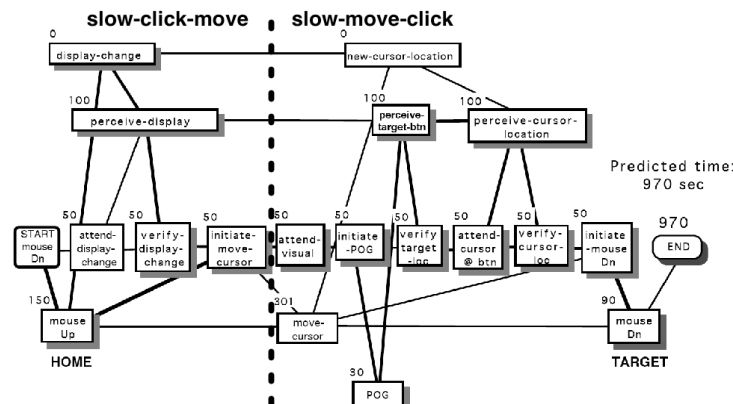


Figure 1. The *home-to-target* task (Gray & Boehm-Davis, 2000) is composed of two microstrategies, divided by a dotted line.

In this context, the user would not know whether a *slow-move-click* should be used until after the microstrategy must be selected (i.e. until after the target has been perceived and verified earlier in the *move-click* microstrategy). Requiring the model to make decisions before all of the necessary information is available indicates that microstrategies are not of an appropriate granularity to model strategic selection in a task. Although microstrategies may still explain the behavior *post hoc*, their inflexibility and large size makes them inadequate for modeling the non-deliberate, zero-time strategic selection that Gray and Boehm-Davis propose to explain behavior.

This problem could be addressed by postulating the availability of a smaller microstrategy that would deal with this specific condition. However, this solution is essentially a move toward making increasingly smaller microstrategies, which begs the question of why the current strategy is not simply composed of two smaller ones in the first place, especially if microstrategy selection has no associated cost. Without a theory of microstrategy structure, there is no reason why microstrategies would not simply end up being the individual operator boxes.

### Microstrategy Composition and Structure

The microstrategies, as composed to achieve the button study task, appear to result in incongruous model behaviors. Redundant model actions are an example of these problems. In the *target-to-home* sequence shown in Figure 1, the *display-change* indicating the target location is perceived and verified as part of the *slow-click-move* microstrategy. Then the target location is perceived and verified again in the first *attend-perceive-verify* cycle of the following *slow-move-click* microstrategy. This is unnecessary because the exact same information is being gathered twice. Along similar lines, the first *attend-perceive-verify* cycle in the task shown in Figure 1 does not include an eye movement operator to shift the point of gaze (*POG*) so that the eyes are at the target when perception starts. This is also a problem because the eyes cannot already be at the target because it was not previously visible. Then, in the redundant *attend-perceive-verify* cycle already described above, there is an eye movement that is unnecessary because the eyes would already be at the target if the first *attend-perceive-verify* cycle were correct (included an shift *POG* operator).

This may be the consequence of an incomplete specification of a set of microstrategy-generation rules. While it is, of course, possible to create a version of *click-move* with a shift eyes (e.g. *slowest-click-move*), this method of creating microstrategies for each subtle task context does not seem like the right approach to capturing the subtleties of dynamic adaptive behavior.

### Microstrategy Dependency Constraints

In CPM-GOMS, and as used by Gray and Boehm-Davis, temporal dependencies between the operators that make up a microstrategy indicate the ordering relationships between those operators. Historically, dependencies have been

treated as homogenous though they often represent different types of relationships. For example in *slow-move-click* (shown in Figure 1), the cognitive cycle to *initiate-mouseDn* must complete before the *mouseDn* can occur. Along these same lines, the *verify target-loc* must occur before the *attend cursor@target*. The dependency approach makes no distinction between these two constraints. However, a closer look reveals that there is a relevant distinction. Motor operators *must* be preceded by cognitive operators that initiate them. This is an architectural requirement for all motor operators. However, it is not the case that *attend* always requires a preceding *verify*. For example, in *slow-move-click*, the first *attend-display-change* is not preceded by anything. Therefore, the *verify-attend* relationship is not architectural but is imposed by the task.

This distinction is important because it greatly reduces the degrees of freedom with respect to how microstrategies can be constructed. There is a basic set of invariant architectural constraints, leaving only those constraints imposed by the task to vary. This view suggests that microstrategies may be thought of as conglomerations of architecture-level units (see John & Kieras, 1996, for examples of CPM-GOMS templates at this level), where the inclusion and order of those building blocks in the model is determined, not by the microstrategy-level structure but rather by the information requirements of the task. Adaptive behavior may then be seen as the point selection of a minimal set of architecturally invariant building blocks based on the information requirements imposed by the task. This is in contrast to the position that adaptive behavior results from the selection of increasingly optimized microstrategies.

### Architectural Process Cascades

In response to the limitations of microstrategies described above, we propose the construct of Architectural Process Cascades. APCs are motivated by a commitment to the hypothesis that there is a cognitive architecture and that it provides functionality that can be adaptively configured for task performance. APCs are invariant functional units constrained by the cognitive architecture. There are two strong claims being made by adopting APC-level task decomposition. First, APCs are the locus of dynamic adaptation at task execution time. It is APCs that are selected during task execution, not microstrategies of the granularity presented in “Milliseconds Matter”. The granularity of APCs is consistent with the granularity of production rules in all the major cognitive architectures. Second, the information flow requirements within each architectural unit implicitly carry the information about how units interact with one another in a way that temporal dependencies cannot. Both of these claims are unpacked below in the following sections covering APC type, selection, composition and structure, information flow, and cascade constraints.

## APC Types

There are four APCs total in the button study task: *motor-simple*, *motor-Fitts*, *perception-simple*, and *perception-plus-shift*. The *motor-simple* includes a cognitive initiate followed by a motor operator representing a mouse-up, mouse-down, press key or other action that uses a parametric estimate. *Motor-Fitts* specifically represents motor actions predicted by Fitts's Law, which is not well represented in the Gray and Boehm-Davis microstrategies. In both *slow-move-click* and *fast-move-click*, a perception cycle of 250 milliseconds gathers information about the target's location during the move, but there is no representation of the perception necessary to do the multiple, successively more fine course corrections required to move the mouse to the target location (Card, et al., 1983). To address this issue, the *motor-Fitts* APC includes the following: an initiate that kicks off both motor and perception, a move, and concurrent perceptual and fixate operators that span the duration of the move to represent the motor-visual feedback loop required for the many course corrections to reach the target.

There are two perception APCs. *Perception-simple*, the case for audition and visual perception in which the eyes are already at the target, is composed of: *attend*, *perceive*, and *verify*. *Perception-plus-shift*, the visual case in which the eyes must move to the target, adds a cognitive operator to initiate eye movement and the eye movement itself to the *perception-simple* operators so in total it includes: *attend*, *initiate POG*, *shift POG*, *perceive*, and *verify*.

## APC Selection

APCs address the ambiguous selection rules that arise in a microstrategy-based model (e.g. choosing *slow-move-click* versus *fast-move-click*). APCs represent the most fundamental building blocks and can be selected and applied exactly when needed. They allow selection based on the information requirements of the task, which include both the sequence of operators and what information is available, and do not force the model to predict which strategy to apply before all of the necessary information is available. In the example of a button study task in which both the location and size of the target location are unknown, the model can select an additional visual *perception-simple* APC to verify the cursor at the target only if the size of the target has been perceived to be small in the previous vision APC. This does not require the model to make predictions at the start of the move based on incomplete knowledge as a microstrategy-based approach does. APCs therefore more accurately capture how and when strategic selection occurs, resulting in a consistent method for predicting information-driven strategies.

## APC Composition and Structure

There are five different microstrategy variants in the existing *click-move* and *move-click* families, and some variants in the current families are not represented (e.g.

*slowest-click-move* proposed above). In addition, many entire families of commonly executed HCI behaviors have yet to be constructed including: *move-without-click*, *double-click*, *click-and-drag*, etc. In order to generate these variants, the individual operators would be strung together with no underlying theory to govern microstrategy construction and composition. However, APCs are consistent in construction as architectural invariants which leads to more consistent longer sequences of composed behavior both in the button study task and in others contexts. For example, *move-without-click* is composed of one motor and one perception APC and a *double-click* is composed of two motor APCs. APCs offer a more consistent explanation for adaptive behavior that does not require additional theoretical work to create novel behaviors as APCs are reused. In this case, simplified compositional theory supports more consistent prediction of dynamic adaptive behavior.

## APC Information Flow

Within APCs, individual operators are linked via buffers that represent a constraint and an information cascade rather than by temporal dependencies. The goal is to model the flow of the information (but not its actual content). Buffers are modeled as resources and are useful here for two reasons (Vera, et al., 2004). First, the explicit creation of buffers deters the modeler from creating implausible information flows (e.g. a buffer in which the hand sends information directly to perception). Second, the explicit use of buffers yields more consistent models of inter-microstrategy and now inter-APC relationships. A temporal dependency model permits undesirable interleaving; for example, the schedule ordering *init(x)*, *init(y)*, *click(y)*, *click(x)* is legal but cognitively implausible (Howes, et al., 2004). Buffers reduce the number of constraints required to link the operators and ensure that the resource constraints, along with the task constraints, drive the behavior predictions rather than temporal dependencies that can more easily either over-constrain or under-specify the real architectural relationship.

## APC Cascade Constraints

APCs are built around architectural relationships between operators. For example, that a motor operator (hand or eyes) requires a cognitive operator to initiate it is an architectural relationship between operators. Task relationships are those required by a particular combination of task and user interface. For example, in the button study task the user must click on home followed by the target button and in an ATM banking task the user must enter her pin in a particular order followed by the 'OK' button.

Given the construction of APCs, there are two possible types of relationships: 1) intra-APC meaning between operators within a single APC, and 2) inter-APC meaning between two APCs. As stated above, APCs are constructed such that intra-APC operator relationships must be architectural. Inter-APC relationships are task constraints

that specify a particular sequence of behaviors required to accomplish the task. Task-level ordering does not assume borders at any point in the longer sequence of activity other than those between the APCs. That is, task-level ordering does not imply microstrategy-level composition of the task: the *home-to-target* and *target-to-home* sequence is achieved, not by four microstrategies, but by four APCs, each repeated several times. The distinction between task and architectural relationships is more than a clarifying formalism in that it provides a theoretical bound on the construction of APCs and the composition of APCs into tasks.

### CORE Model of Button Study Tasks

To both test and demonstrate the APC construct, we briefly introduce the CORE architecture and then cover an APC-based model implemented in CORE. The Cognitive Constraint Modeling (CCM) approach, instantiated in software as CORE (Constraint-based Optimal Reasoning Engine), is characterized by two principles that distinguish it from existing approaches. 1) The system takes as input a description of the constraints on a particular task and on the human cognitive architecture. 2) Based on those constraints, the system performs a search to automatically generate an optimal prediction of human performance. For a more complete description of CCM and CORE, the reader is referred to Howes, et al. (2004).

The APC model addresses many of the issues covered in the *CPM-GOMS Templates as Microstrategies* section above. The time prediction is quite close both to Gray and Boehm-Davis’s original predictions and to the experimental data they collected. This supports the empirical evidence that users are sensitive to subtle differences in task context and challenges the argument for microstrategies as a cognitive unit. The APC model, shown in Figure 2, is composed of the APC types described above: a *motor-simple* APC for the click on home, a *perception-plus-shift* to perceive the target, a *motor-Fitts* to move to the target, a

*perception-simple* to check that the cursor is at the target, and a *motor-simple* to click on the target. This model also addresses the inconsistencies described in the *Microstrategy Composition and Structure* subsection of the *CPM-GOMS Templates as Microstrategies* section above. Though this model is a departure from the assumptions of “Milliseconds Matter”, the predicted time for *home-to-target*, according to the model, is 1031 milliseconds. This value is comparable to the Gray and Boehm-Davis models (970 milliseconds) as well as the empirical measurements (994 milliseconds). For the purposes of comparison, times were measured from the beginning of the *mouse-up-home* to the end of the *mouse-down-target* as measured in “Milliseconds Matter”.

Though neither the Gray and Boehm-Davis models nor the APC models can be considered more “correct” without further empirical work, the APC model, composed of four types of APCs and a task-level ordering of those APCs, is closer to the level of architectural invariants and is more compositionally consistent than the Gray and Boehm-Davis models.

### Conclusion and Discussion

“Milliseconds Matter” is an important piece of work because it is a compelling application of the insight – perhaps established most clearly in the Meyer and Kieras (1997) work on EPIC – that people adapt their interactions in an extremely fine-grained way in order to achieve greater efficiency while satisfying task demands. What we are questioning here is the use of microstrategies as the vehicle for bringing to bear this insight on problems in human-computer interaction. Our principal conclusions about microstrategies may be summarized as follows. First, microstrategies do not comprise a psychological theory of cognitive task units: there is no empirical evidence for them, and there is no explicit theory of how to carve up behavior into microstrategies.

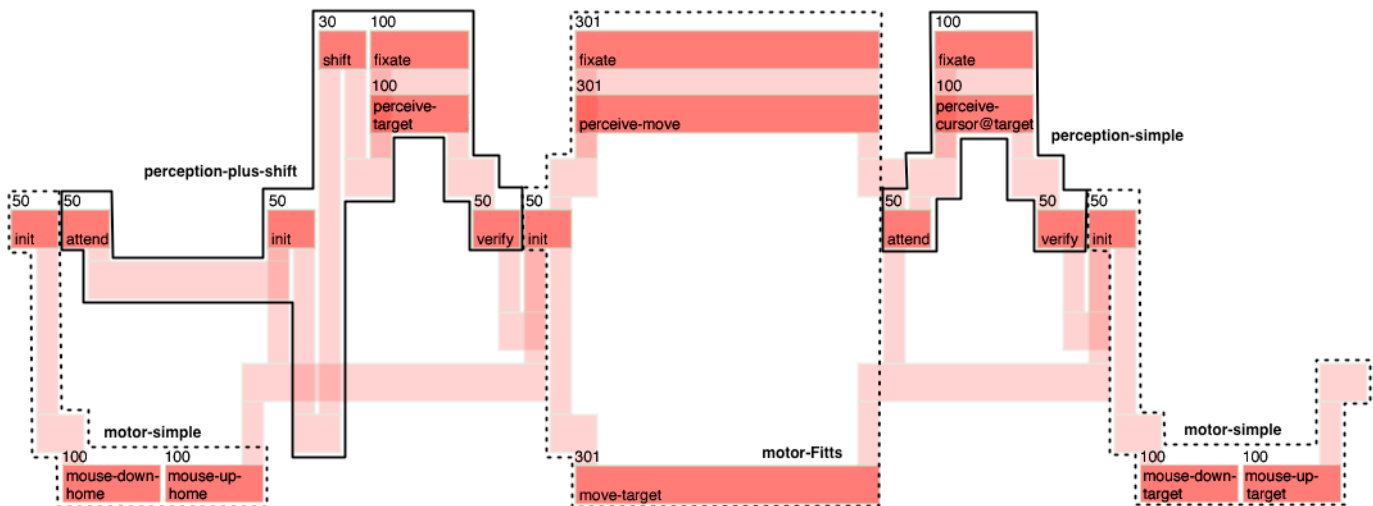


Figure 2. Model of the *home-to-target* task implemented in CORE.

Furthermore, microstrategies as units leave no residue in the final task models. Speculation about the mechanisms underlying dynamic behavioral adaptation (e.g., whether the selection is deliberate or not, as Gray and Boehm-Davis discuss) is at best premature.

Second, even if microstrategies are interpreted only as a notational convenience, because there is no explicit theory of either their internal construction or their composition into larger behaviors, there are negative consequences for the modeler. The most serious from an applied psychology point of view is that, even though the present microstrategies and their composition were shown to fit the data, the lack of an explicit theory base makes applying the general approach and even these specific microstrategies to new situations more of an error-prone art rather than a rigorous engineering method.

Our own experience in attempting to automate the composition of temporal dependency-based microstrategies (Vera, et al., in press) suggests that the CPM-GOMS notation is itself too impoverished to support a systematic theory of composition. But rather than completely abandon the advantages of clarity and simplicity that attracted Gray and Boehm-Davis to the CPM-GOMS notation in the first place, relative to complete architectural simulations such as ACT-R (Anderson, et al., 2004), we have offered a richer ontology of operator relations based on resource-bound information cascades which leads to the postulation of a new modeling primitive (APCs) that sits at a higher level than CPM-GOMS operators. The advantage is that APCs contain the required architectural constraints to properly bound the composition of larger behaviors. The combination of APCs with optimizing constraint satisfaction yields a powerful modeling framework that we believe goes some way toward reducing theoretical degrees of freedom, and moves closer to the vision of deriving detailed cognitive behavior directly from architectural theory plus task constraints (Howes, et al., 2004).

Finally, it should also be clear that we have taken only the first steps here. A full demonstration of the generative power of APCs and Cognitive Constraint Modeling with respect to microstrategic variation is to demonstrate how a taxonomy of interactive strategies such as that identified in Table 2 of "Milliseconds Matter" can emerge (and be computationally derived) from searching a generative, architecturally constrained space of behaviors with respect to an explicit objective function. We are pursuing this path and believe it will yield substantial gains in both theoretical explanation and applied cognitive modeling methodology.

### Acknowledgements

This research was supported by the NASA *Aviation Operations Safety* and *Intelligent Systems* Programs.

### References

Anderson, J., R., Bothell, D., Byrne, M., D., Douglas, S., Lebiere, C., & Qin, Y. (2004). An Integrated Theory of the Mind. *Psychological Review*, 111 (4), 1036-1060.

- Card, S. K., Moran, T. P., & Newell, A. (1983). *The Psychology of Human-Computer Interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- Gray, W. D., & Boehm-Davis, D. A. (2000). Milliseconds Matter: An Introduction to Microstrategies And To Their Use In Describing And Predicting Interactive Behavior. *Journal of Experimental Psychology: Applied*, 6(4), 322-335.
- Gray, W. D., John, B. E., & Atwood, M. E. (1993). Project Ernestine: Validating GOMS for predicting and explaining real-world task performance. *Human Computer Interaction*, 8(3), 237-309.
- Howes, A., Vera, A., Lewis, R., & McCurdy, M. (2004). Cognitive Constraint Modeling: A Formal Approach to Supporting Reasoning About Behavior. *Proceedings of the 26th Annual Meeting of the Cognitive Science Society*, (pp. 595-600). Hillsdale, NJ: Lawrence Erlbaum Associates.
- John, B. E. (1993). *A quantitative model of expert transcription typing* (Tech. Rep. CMU-CS-93-120). Pittsburgh, PA: Carnegie Mellon University, School of Computer Science.
- John, B. E. (1990). Extensions of GOMS Analyses to Expert Performance Requiring Perception of Dynamic Visual and Auditory Information. *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 107-116). New York, NY: ACM Press.
- 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(4), 320-351.
- John, B. E., Vera, A., Matessa, M., Freed, M., & Remington, R., (2002). Automating CPM-GOMS. *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 147-154). New York, NY: ACM Press.
- Lewis, R.L., Howes, A., & Vera, A. (2004). A constraint-based approach to understanding the composition of skill. *Proceedings of the 6<sup>th</sup> International Conference on Cognitive Modeling* (pp. 148-153), Mahwah, NJ: Lawrence Erlbaum.
- O'Hara, P. K., & Payne, S. J. (1999). Planning and the user interface: the effects of lockout time and error recovery cost. *International Journal of Human-Computer Interaction Studies*, 50, 41-59.
- Vera, A. H., Howes, A., McCurdy, M., & Lewis, R. L. (2004). A constraint satisfaction approach to predicting skilled interactive cognition. *Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 121-128). New York, NY: ACM Press.
- Vera, A. H., John, B., E., Remington, R., Matessa, M., & Freed, M. (in press). Automating Human-Performance Modeling at the Millisecond Level. *Journal of Human-Computer Interaction* (in press).