

# Developing Human Performance Models Using Apex / CPM-GOMS for Agent-Based Modeling and Simulation

Seung Man Lee, Roger Remington, Ujwala Ravinder, and Michael Matessa

NASA Ames Research Center  
MS 262-4  
Moffett Field, CA 94035

{smlee, rremington, uravinder, mmatessa} @mail.arc.nasa.gov

**Keywords:** Human performance, Apex, GOMS, Agent-based, Air traffic control (ATC)

## Abstract

This paper reports on the development of agent models with human-like performance characteristics for use in agent-based modeling and simulation. The goal of our effort is to develop a computational human agent model that can be incorporated into an agent-based simulation to evaluate the impact of new technologies, such as datalink technology, on the workload and situation awareness of air traffic controllers. To provide models of advanced airspace, domain and task knowledge must be integrated with human performance characteristics to achieve a specification of controller functionality. Our approach combines a task analysis, which provides a functional description of the domain, with a human performance architecture, from which detailed performance predictions can be made. This paper presents how a standard task analysis can be paired with a human performance model to generate behavior that approximates that of the human operator. This paper will also outline the issues of integrating human performance models into a large-scale distributed agent-based simulation.

## INTRODUCTION

Modeling and simulation have emerged as critical tools for the design and safety analysis of large-scale complex systems such as the National Airspace System (NAS) [1]. To be a useful design and analysis tool, simulation requires suitable element-level models, such as dynamic models of physical machines, to predict the evolution of complex interactions of the elements. While this approach has been successfully applied to physical system components, only recently have modeling and simulation of human behavior begun to be explored. Humans are integral system components and critical elements to the performance and safety of the large-scale complex systems. Human capabilities and limitations must be taken into account when developing procedures, interfaces, and systems. Therefore, system modeling and simulation will be more effective if it includes human characteristics as an essential feature.

Though the scientific understanding of human behavior is far from complete, current theories and findings have enabled successful modeling of human-computer interaction (HCI) [2]. However, this success has not generalized to the modeling and

simulation of large-scale dynamic systems. In part this is because the development of computational human performance models requires spending a great deal of time, cost, and expertise on human cognition and behaviors. Also, human performance models typically execute in their own unique simulation environments. For widespread use it will be preferable to integrate human models into system simulations along with other system components.

In simulating large-scale complex systems agent-based modeling and simulation (ABMS) has proven to provide robustness and flexibility. Also, by combining multiple agents it is possible to observe emergent properties of interactive behavior [3]. Agent-based simulation integrates human-like agent models in a dynamic representation of their environments, which includes detailed models of the physical and technical systems. However, given the complexity of human performance models, creating the ability to interact with other simulation models can require significant adaptations.

Several recent studies have examined using such human performance models as agents in large-scale agent-based simulations to evaluate air transportation systems [4-6]. These human agent models drive high-level behaviors by applying domain knowledge. However, they do not attempt to model the capabilities and constraints on human performance. The recent developments in computational modeling of cognitive human agents provide a new scientific paradigm for addressing and understanding fundamental aspects of human behavior and cognition at the individual level and system level. Our goal is to extend the capability of human agent models by developing an agent that produces human-like behaviors. We have designed this agent for the evaluation of new advanced air transportation concepts. By capturing performance characteristics we believe it will be possible to estimate factors such as workload, multitasking, and operator throughput.

There are several issues in developing agent models of human behavior and cognition. First, the representation of human behavior and cognition must be able to provide insight into the impact of human performance on overall system performance to predict global consequences of system changes [7]. Second, agent models of human behavior and cognition must be able to be easily modified and quickly adapted to new situations. Existing human performance models typically require deep knowledge of human behavior and cognition and a great amount of time and efforts to develop a new human performance model for a specific domain of interest. Third, agent models of human behavior and cognition should be implemented in a computational architecture with the capability for representing the range of behavior found in complex

task domains, and that can be incorporated into a large-scale simulation environment to interact with other agents.

In this paper, we describe a framework for developing agent models to resolve those issues using Apex (Architecture for Procedure Execution) developed at NASA Ames [8]. Apex is designed for generating adaptive, intelligent human behavior in complex, dynamic environments such as the air transportation system. Apex incorporates many high-level aspects of cognition such as action selection under uncertainty, managing multi-tasking, and task interleaving. Apex includes a range of components for modeling, simulating and analyzing human behavior, so Apex makes it easier to create human performance models. By facilitating model development and modification Apex makes it possible to test a wide variety of procedures and interfaces and to examine the impact of new technologies such as data link on the overall performance and safety of the NAS.

Our approach combines a task analysis, which provides a functional description of the domain, with a human performance architecture, from which detailed performance predictions can be made. We have previously reported on a method that seamlessly links task analysis with elementary human performance components. This linkage is the basis for much of the usability of Apex. Before describing our air traffic control model, we briefly describe our approach to human performance modeling.

## APEX HUMAN AGENT MODELING

In modeling human performance, economy can be achieved by representing the human agent in terms of fundamental characteristics constant across situations. This aspect of the agent can be reused across domains. The model must also represent domain and procedural knowledge, typically represented in a task analysis. Our modeling approach draws on previous work [2, 9] that integrates task with human characteristics in a unified representation. This representation extends the task analysis by including a fundamental human resource architecture at the level of cognitive, perceptual, and motor (CPM) operators.

The task analysis consists of a hierarchical task decomposition based on the Goal, Operators, Methods, and Selection (GOMS) technique [2]. The resource architecture currently implemented is based on CPM-GOMS [10], an extension of GOMS. CPM-GOMS is a modeling method that combines a hierarchical task decomposition with a resource architecture. The GOMS analysis terminates in low-level Cognitive, Perceptual, and Motor operators. CPM-GOMS models have made accurate, zero-parameter predictions about skilled user behavior in routine tasks. CPM-GOMS specifies the parallelism and timing of elementary cognitive, perceptual, and motor operators [11].

The inclusion of a fixed processing architecture at this level of detail allows our agents to capture important characteristics of human performance that remain fixed across tasks and domains. Capturing resource demands is a crucial component of predicting how effectively displays and controls are designed, or how efficiently two concurrent tasks can be done. Resource allocation policy is central to learning how to perform in a task context, and the competition for limited resources determines how tasks are sequenced at a detailed level of performance. The method we describe is being applied to simulate the performance of a human operator for the purpose of evaluating new concepts of advanced air transportation system.

We have recently described an approach for automatically generating CPM-GOMS analyses using the Apex computational architecture [12]. The automation of CPM-GOMS in Apex makes it practical for the first time to derive detailed predictions of human performance with complex tasks and interfaces. We have focused initially on evaluating routine human-system interaction by predicting the time and resource demands of accomplishing common interface tasks.

## Apex Human Agent Architecture

In our approach performance on a task is constructed from elementary human cognitive, perceptual, and motor operators, whose characteristics are relatively constant across domains. Key to our success with the compositional approach has been the development of a method for automatically scheduling these elementary human resources, which was achieved using the Apex computational architecture. A high-level architecture of Apex is shown in Figure 1.

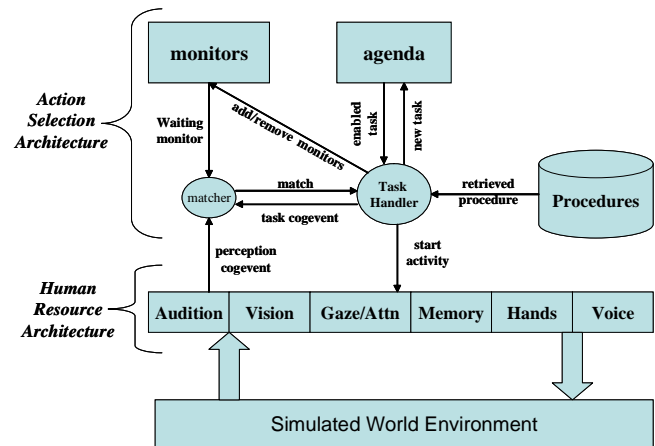


Figure 1. The Architecture of Apex

The modeling framework includes a *Resource Architecture*, an *Action Selection Architecture*, and a *Procedure Library*. The resource architecture defines the limited-capacity cognitive, perceptual, and motor components as shown in Figure 1. The action selection architecture coordinates the activity of the resources and applies knowledge in the form of procedures. The procedure library contains the knowledge the agent applies to perform in the target domain. All knowledge in Apex is in the form of procedures. Information about the world is input through perceptual processes, which have limited capacity. Incoming information is matched against the specifications of procedures in the procedure library. If conditions for a procedure are detected then the Action Selection Architecture schedules the steps of the procedure in accordance with the constraints specified below.

As discussed earlier the model combines the CPM-GOMS task analysis with the Apex computational architecture, which provides the underlying simulation framework. Apex is a software tool for creating, running and analyzing simulations of intelligent agents carrying out human-computer interaction tasks. Apex treats the problem of modeling behavior as a problem of scheduling an agent's limited resources. The agent architecture incorporates a reactive planning and execution mechanism with integrated online resource scheduling and other capabilities needed to handle

multiple tasks. Both the general capabilities of a reactive planner and the multitask management extensions specific to Apex have proven central in automating CPM-GOMS models.

The reactive planner recursively decomposes high-level goals into subgoals and primitive operators based on stored plans. It does not generate the goal hierarchy all at once. Instead, it waits until all preconditions associated with a given goal are satisfied before retrieving a plan to specify subgoals. Deferring goal decomposition until near execution time enables the planner to choose how to decompose (i.e. which procedure to use) with as much situation information as possible. This strategy is essential for tasks such as air traffic control where uncertainty about future world state is high and the need for careful deliberation, either to solve complex problems or to choose optimally from a wide range of possible solutions, is low. The reactive planner allows the Apex controller agent to be interrupted by voice communications or by alerts that signal the need for immediate action.

### Templates of Elementary Human Behaviors

Representing the human agent in terms of elementary cognitive, perceptual, and motor actions may allow generalization, but it increases the complexity and difficulty of constructing human performance models. Modeling human performance at this level requires spending a great deal of time, cost, and expertise on human behaviors.

To facilitate model construction, we decompose a complex task into a set of *templates*, primitive task-level operations that represent the building blocks from which other task behavior will be constructed [13]. Templates incorporate a psychological theory of the cognitive, perceptual, and motor components of basic human activities and dependencies between these components. The choice of primitives and the method of combining basic cognitive, perceptual, and motor operations into larger behavior units are critical in constructing reusable and scaleable templates. For example, a template of move-and-click trackball action can be used to build models of other tasks.

In CPM-GOMS, the top-level goal is decomposed into subgoals down to the level of templates. Apex composes long behavioral sequences by interleaving the templates for successive behaviors. Composing behavioral sequences in this way from templates that describe fundamental task actions allows the model to easily simulate the overlapping of tasks characteristic of skilled human performance. Templates remove the burden of understanding and programming in the underlying cognitive architecture. Consequently, the use of templates could make cognitive modeling more accessible to a wider range of domain experts. Templates may allow modelers to easily incorporate existing models of generalized capabilities into models of more complex tasks.

The process in CPM-GOMS of producing a goal hierarchy by recursively applying methods to non-primitive goals (those that do not correspond to an operator) does not end at the level of templates. Instead, templates themselves are decomposed into simpler behavioral units, ending when leaf nodes of the hierarchy form a set of operator-level actions composed of elementary cognitive, perceptual, and motor activities. Thus, the task decomposition is carried out uniformly at all levels. By modeling these elementary behaviors we can develop a library of behaviors from which to construct larger behavioral sequences. In this way, we bridge the gap between the standard task analysis and a human performance model of some detail.

There are two issues that arise in this compositional approach: how templates are constructed, and how they are combined to produce extended behavioral sequences. We have previously reported a method of interleaving the elements of successive templates that provides excellent fits to observed data [12]. The details of this method are beyond the scope of this paper, but in brief, we have successfully fit data from human-computer interaction tasks including mouse-based automated teller simulations, typing, and computer aided design tasks. The method is currently being applied to shuttle cockpit procedures, airline cockpit procedures, and air traffic control operations.

Our compositional approach to simulating behavior relies on a theory of resources to predict concurrency, and on a software architecture to execute reactive plans. Concurrent operator execution occurs within and across templates. Within-template concurrency arises because cognitive, perceptual, and motor operators call on separate, independent resources. Between-template concurrency arises from the interleaving of operators from different templates. The essence of the interleaving phenomenon is that the activities specified by a template do not use all resources all of the time; idle time (slack) in the use of a resource by one template's operators represents an opportunity for operators from a later template to "slip back" and begin execution. Interleaving at the level of CPM-GOMS operator-level goals corresponds to overlap in the execution of higher-level goals – i.e. at the level of templates or classic GOMS operators – and thus accounts for the different predictions of the CPM-GOMS and classic GOMS approaches.

In order to allow Apex models to participate in large-scale simulations of the national airspace system, we integrated Apex into the ACES (Advanced Concepts Evaluation System) simulation environment developing at NASA [14].

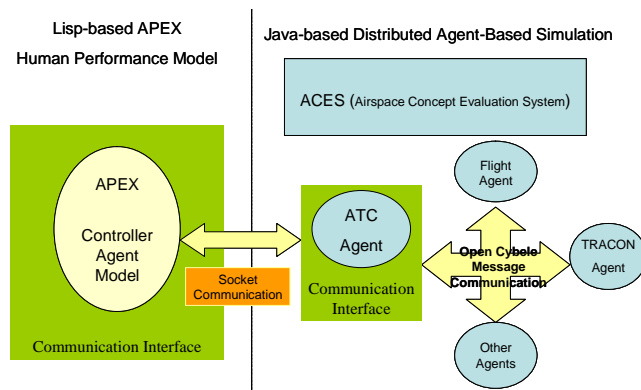
## INTEGRATION OF APEX INTO ACES

In the Virtual Airspace Modeling and Simulation (VAMS) project, a joint NASA and FAA effort, modeling and simulation methods are being applied to evaluate the effect of changes in the operation of the national airspace. New concepts and technologies are being proposed as potential ways of increasing the capacity and safety of air transportation system. Several airspace concepts involve advanced automation on the ground communicating with automated systems onboard aircraft to negotiate clearances automatically. New automation under consideration would provide individual pilots and air traffic controllers more information about their situation, and help them in predicting near-future states of the airspace. VAMS is developing the modeling and simulation infrastructure that would allow the effect of these proposed changes to be evaluated.

To simulate novel airspace concepts VAMS has developed a new airspace simulation system, ACES. Briefly, ACES is a distributed agent-based event-driven simulation for the analysis of the NAS supporting common communications protocols. For the distributed simulation, ACES uses the High-Level Architecture (HLA) standard developed by the US Department of Defense (DOD) and the Run-Time Interface (RTI) available from the Defense Modeling and Simulation Office (DMSO) [14]. ACES provides key agent models for the NAS simulation, including aircraft, airport, Terminal Radar Approach Control (TRACON), Air Route Traffic Control Center (ARTCC), Airline Operations Center (AOC), and weather. These different types of agents are

interacting with each other through message passing during the simulation. ACES models the dynamic behavior of different aircraft types, as well as the procedures and communications agents such as aircraft and air traffic control. In its current build, ACES represents agents at a level that abstracts over individual humans and human-system interfaces. Since an important goal for us is to evaluate such interfaces, we model the entire joint human-system, and use the combined output to control an ACES agent.

In this research, a simple Apex human agent model for air traffic controller was developed to perform the functions of an individual air traffic controller in the simulation. A human-like agent must meet a set of functional requirements that derive from the role it will play in the simulation, and from our desire to closely match human behavior. One of practical issues was how to integrate these two different simulation systems to run seamlessly together. Apex is written in Lisp while ACES is a Java-based simulation system. To enable communication between these two systems written in different programming languages we developed communications interfaces and protocols using sockets. For example, the Apex controller agent communicates with ACES agents through a TCP/IP socket channel as shown Figure 2. Conceptually, ACES is responsible for all vehicles, air space entities, and the simulation environment. Apex is responsible for interpreting the simulation world and controlling the aircraft, as a human controller does.



**Figure 2. Integration of Apex into ACES**

The Apex-ACES Interface (AAI) translates simulated displays and vehicle information into the representation used by Apex and translates Apex's action representation into ACES function calls (e.g., to control the vehicle and to manipulate sensors, radios, and displays). To control the vehicle's motion Apex issues output commands to a corresponding ACES agent. AAI converts the output commands to OpenCybele message format, which can then be interpreted by other agents in the simulation. Currently, control of a vehicle occurs through setting the desired state of the vehicle such as its speed, heading, and altitude.

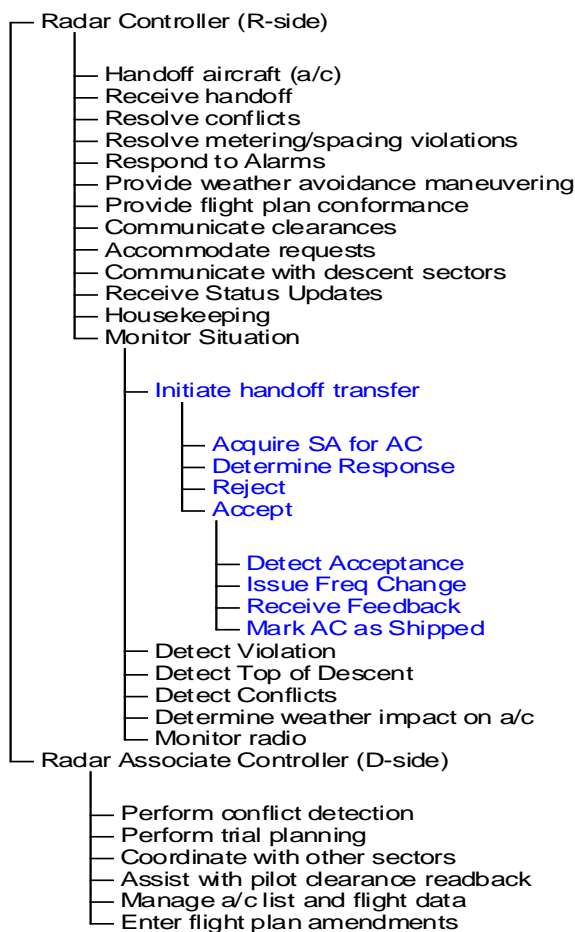
Apex is used to model both the human agent and the displays and controls the agent interacts with. In this way, timing of agent actions and delays imposed by the equipment and user interface can be simulated at high fidelity. Communication with the ACES simulation will occur at synchronized message passing times in accordance with ACES protocols. This arrangement is depicted in Figure 2. This scheme provides a natural division between the

human performance model and the ACES-level agent. The ACES incorporates objects that represent high-level agents. The Apex agent need only transmit to its ACES counterpart those messages of significance in the larger context. Thus, while each keystroke of data entry into flight computers must be simulated to predict the time, the ACES-level agent need only be informed when the keystrokes produce some change in its state, such as activating a mode, or changing a control setting.

## AIR TRAFFIC CONTROL TASK ANALYSIS

Our Apex agent functions as an air traffic controller who controls traffic in a single enroute sector. At present, ground based air traffic control installations have responsibility for routing of aircraft and maintenance of safe separation between aircraft. There are three principle types of installation: tower, TRACON, and enroute. Tower control handles surface operations including take-off and landing. TRACON handles arrival and departure sectors for airports. Enroute handles the remainder of the flight between departure and arrival sectors. A controller in the ATC environment, for instance, monitors the ongoing flights, responds to various pilot requests, and adjusts to weather conditions by instructing pilots to alter their aircraft speed, flight levels, and often headings in order to maintain safe and efficient flow of air traffic throughout the airspace control sectors. Other entities modeled in the ACES simulation include the airline operating centers, which are run by each airline and provide schedule and gate information to their own aircraft, and the aircraft themselves. Previous work has attempted to provide a more or less complete functional analysis of enroute [15]. Those have not yet been turned into workable computational agents. Some progress toward a complete controller agent has begun, but the controller's task is a complex mixture of spatial, geometric, temporal, and procedural reasoning. No software agent has yet been able to handle more than a portion of the entire task though progress in applying rule-based systems has yielded agents with some capability.

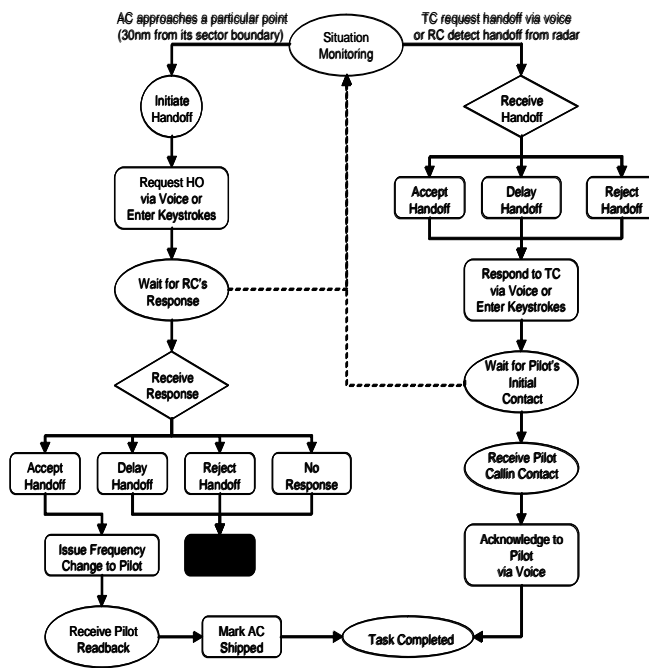
We have begun our agent development by focusing on the task of enroute controllers, following existing task analyses [16, 17]. A high level functional analysis of a portion of current enroute control is shown in Figure 3. Each sector is managed by two controllers, the radar controller (R-side) and the radar associate controller (D-side), whose duties complement and overlap. The R-side controller is the primary controller in contact with aircraft. The D-side controller assists, sharing responsibility for conflict detection and planning, as well as performing routine housekeeping. It is apparent from Figure 3 and the earlier discussion that an individual controller is an agent among airspace entities. Each individual agent shares responsibility for ensuring safe operations, but those responsibilities are themselves divided. For example, the aircrew is primarily responsible for the safety of their aircraft, the controller for the safety of all aircraft in his/her sector. In addition to the formal rules for controllers (as well as aircrew) there exist informal practices. These informal rules have not been exhaustively studied, but are apparent in the behavior of controllers. They represent social contracts between entities that have emerged with time. Because of reciprocity, many informal rules act for the general good of all players. For example, controllers will try to manage the traffic in their sector so that they do not overburden the downstream controller to whom they hand the traffic off. This cooperative behavior may not apply if they are busy.



**Figure 3. High-Level Tasks of Enroute Controllers**

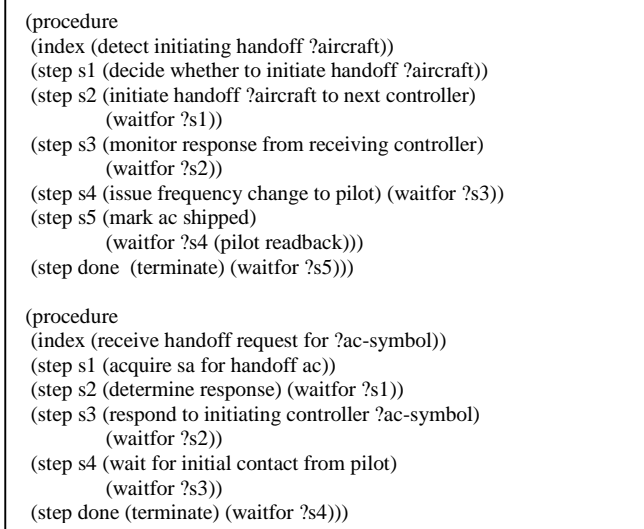
We developed a simple model of this procedure in Apex using observed times for the interaction of the various agents involved. As shown in Figure 3, the tasks of air traffic controllers require multi-tasking capability. For example, situation monitoring task is performed in parallel with other tasks, as are handoffs. However, in our Agent model, tasks that require the same resources such as radio, trackball, or keyboard, can temporarily prevent parallel execution. For example, a controller can visually detect two handoffs simultaneously, but the controller can only accept one handoff at a time because the task requires the same resource, the use of trackball. This illustrates how the resources in our Apex controller agent capture capabilities and constraints on human behavior, which have consequences for overall performance.

Typically R-Side controllers initiate handoff when an aircraft reaches a trigger location (e.g. 30nm from boundary of next sector) and the receiving controller accepts a handoff before an aircraft enters his sector. Once a handoff is accepted, the transferring controller issues a frequency change clearance before the aircraft leaves his control sector. The details of this Transfer of Communications (TOC) differ for voice or datalink, and this difference is of interest in assessing the impact of datalink. Then flight crew will tune new frequency and make an initial contact to receiving controller in next sector. Figure 4 shows the high-level tasks of controllers for initiating and receiving handoffs.



**Figure 4. Task Analysis for a Handoff Task**

The hierarchical goal structure of a GOMS model is expressed in Apex using its Procedure Description Language (PDL). PDL steps are decomposed hierarchically into procedures of simpler steps until those steps bottom out in primitive actions that occupy human resources. Figure 5 shows high-level PDL procedures for the handoff task that decompose into low-level procedures as shown in Figure 6.



**Figure 5. High-Level Task Procedures for Enroute Controller**

```

(procedure
(index (initiate handoff ?ac-symbol to receiving controller))
(step s1 (move-cursor-to ?ac-symbol))
(step s2 (click trackball on ?ac-symbol) (waitfor ?s1))
(step done (terminate) (waitfor ?s2)))

(procedure
(index (move-cursor-to ?target))
(profile right-hand)
(step s1 (trackball-time ?target => ?time))
(step s2 (start-activity right-hand moving-ic-trackball
:object ?target :duration ?time => ?a)
(waitfor ?s1))
(step s3 (terminate) (waitfor (completed ?a))))

(procedure
(index (click trackball on ?target))
(profile right-hand)
(step s1 (trackball-object => ?object))
(step s2 (start-activity right-hand clicking-trackball
:object ?object :ac-symbol ?target :duration 200 => ?a)
(waitfor ?s1))
(step s3 (terminate) (waitfor (completed ?a))))

```

**Figure 6. Template-Level Procedures for Enroute Controller**

## SUMMARY

We have described the use of human agent models in agent-based simulation as a mechanism for analyzing large-scale complex systems, such as the advanced air transportation system. We illustrated this by showing a simple agent model of air traffic controller for handoff using a CPM-GOMS task analysis integrated into the Apex computational architecture. The Apex agent interacts with a distributed agent-based simulation (ACES), making it possible to analyze the impact of changes of human behavior on the overall system performance. We are exploring the use of the model to evaluate new technologies for air traffic control.

As agent technologies and distributed simulation techniques advance, the use of simulated human agent models is expected to provide increased benefits for system design and analysis. However, several practical issues in integrating high-fidelity human performance models into a large-scale simulation must first be resolved. In particular, methods of synchronizing each agent's time advance with other agents, and the computational demands of the human agent must be more fully worked out.

## ACKNOWLEDGEMENTS

This work was supported by the Virtual Airspace Modeling and Simulation project of the NASA Airspace Systems program. The authors wish to thank Lisa Bjarke, Karlin Roth, and Larry Meyn for programmatic assistance, and Parimal Kopardekar and Ken Leiden for making available task analysis and task modeling data.

## REFERENCES

[1] Wickens, C. D.; A. S. Mavor; J. McGee; National Research Council (U.S.). Panel on Human Factors in Air Traffic Control Automation. 1997. *Flight to the future : human factors in air traffic control*. Washington, D.C.: National Academy Press.

[2] Card, S. K.; T. P. Moran; A. Newell. 1983. *The psychology of human-computer interaction*. Hillsdale, N.J.: L. Erlbaum Associates.

[3] Lee, S. M. 2002. "Agent-Based Simulation of Socio-Technical Systems: Software Architecture and Timing Mechanisms," Doctoral dissertation, Georgia Institute of Technology.

[4] Pritchett, A. R.; S. M. Lee; K. M. Corker; M. A. Abkin; T. G. Reynolds; G. Gosling; A. Z. Gilgur. 2002. "Examining air transportation safety issues through agent-based simulation incorporating human performance models," in *Proceedings of the IEEE/AIAA 21st Digital Avionics Systems Conference*.

[5] Jones, R. M.; J. E. Laird; P. E. Nielsen; K. J. Coulter; P. Kenny; F. V. Koss. Spring 1999. "Automated Intelligent Pilots for Combat Flight Simulation," in *AI magazine*, pp. 27-42.

[6] Callantine, T. 2001. "Agents for Analysis and Design of Complex Systems," In *Proceedings of the 2001 International Conference on Systems, Man, and Cybernetics*, 567-573.

[7] Laughery, K. R. and K. M. Corker. 1997. "Computer Modeling and Simulation," in *Handbook of Human Factors and Ergonomics*, G. Salvendy, Ed. New York: John Wiley and Sons, Inc., pp. 1375-1408.

[8] Freed, M. 1998. "Simulating Human Performance in Complex, Dynamic Environments," Doctoral dissertation, Northwestern University.

[9] John, B. E. 1990. "Extensions of GOMS analyses to expert performance requiring perception of dynamic visual and auditory information," In *Proceedings of CHI*, Seattle, Washington, 107-115.

[10] John, B. E. and W. D. Gray. 1995. "CPM-GOMS: An Analysis Method for Tasks with Parallel Activities," In *CHI'95 Conference companion on Human factors in computing systems*, Denver, Colorado, 393-394.

[11] Gray, W. D.; B. E. John; M. E. Atwood. 1993. "Project Ernestine: Validating a GOMS Analysis for Predicting and Explaining Real-World Task Performance," *Human-Computer Interaction*, vol. 9, pp. 237-309.

[12] John, B. E., Vera, A. H., Matessa, M., Freed, M., and Remington, R. 2002. "Automating CPM-GOMS," in *Proceedings of CHI'02: Conference on Human Factors in Computing Systems*: New York, ACM Press, pp. 147-154.

[13] Matessa, M.; A. Vera; B. E. John; R. Remington; M. Freed. 2002. "Reusable Templates in Human Performance Modeling," In *Proceedings of the Twenty-fourth Annual Conference of the Cognitive Science Society*.

[14] Raytheon ATMSDI Team. 2002. "Creation of a Modeling and Simulation Capability to Support NAS-Wide Analyses of Advanced ATM Tools and Concepts," NASA Contractor Report CTO-07 Documents, Contract Number NAS2-00015.

[15] Seamster, T. L.; R. E. Redding; J. R. Cannon; J. M. Ryder; J. A. Purcell. 1993. "Cognitive task analysis of expertise in air traffic control," *International Journal of Aviation Psychology*, vol. 3, pp. 257-283.

[16] Leiden, K. 2000. "Human Performance Modeling of En Route Controllers," Micro Analysis & Design, Inc., Boulder, CO RTO-55 Final Report, Prepared for NASA Ames Research Center, December.

[17] Niessen, C.; S. Leuchter; K. Eyferth. 1998. "A psychological model of air traffic control and its implementation," In *Proceedings of the Second European Conference on Cognitive Modeling*, Nottingham, U.K., 104-111.