10TH INTERNATIONAL COMMAND AND CONTROL RESEARCH AND TECHNOLOGY SYMPOSIUM
THE FUTURE OF C2

A COMPARISON OF AIR DEFENSE WARFARE TASK PERFOMANCE WITH AND WITHOUT AN AUTOMATED TASK MANAGER USING A GOMS MODELING TOOL

Thomas Santoro, PhD, Naval Submarine Medical Research Lab
Human Performance Department
SUBASE BOX 900, Groton, CT 06349
Phone: (860)694-2527, FAX (860)694-4096
Email: santoro@nsmrl.navy.mil

David Kieras, PhD, University of Michigan
Electrical Engineering and Computer Science Department
1101 Beal Avenue, Ann Arbor, MI 48109-2110
Phone: (734) 763-6739; Fax: (734) 763-1260
Email: kieras@eecs.umich.edu

POINT OF CONTACT:

Thomas Santoro, PhD, Naval Submarine Medical Research Lab
Human Performance Department
SUBASE BOX 900, Groton, CT 06349
Phone: (860)694-2527, FAX (860)694-4096
Email: santoro@nsmrl.navy.mil

ABSTRACT

Small teams of human performance models perform tasks in a simulated Air Defense Warfare (ADW) scenario to compare two designs for an advanced multi-modal watchstation (MMWS). In the original design, the models select and perform tasks based on their own knowledge of the on-going air warfare situation as acquired through visual search of individual tactical situation (TACSIT) displays and information windows and verbal communications among model operators. In a revised design, an intelligent agent Task Manager (TM) monitors the situation and generates a list of tasks on a separate display for the operator to monitor**.** While exercises on a prototype TM with human teams were confounded by user issues, the modeling study produced a quantitative comparison of task execution latency and total critical tasks between the two designs that showed generally improved performance for the revised design, thereby demonstrating the utility of human performance modeling for the evaluation of systems for complex team tasks.

INTRODUCTION

Modeling and simulation have been shown to be effective tools for technology evaluation that in some cases are superior to evaluation based on empirical testing with human operators. Among the various advantages to a modeling evaluation is the fact that a model can be expected to perform tasks exactly according to given instructions for every repetition of scenario events while humans are rarely so reliable. That is, because both human individuals and teams will sometimes perform tasks in highly idiosyncratic ways, the results of testing an interface design with human users can be ambiguous: is poor performance due to a poor interface design, or to the failure of the users to use the design properly? This issue can be dealt with either by a large sample of highly trained test users, or by using for the earlier stages of design evaluation a team of simulated humans who use the proposed design as it is intended to be used.

The GOMS (Goals, Operators, Methods, and Selection Rules) model developed by Card, Moran, and Newell (1983) is a well-established method for predicting usability and human performance to effectively guide the design of a user interface early in the design process. A GOMS model represents the procedural knowledge that a user must have in order to accomplish tasks given a particular system and its user interface. Static and dynamic

characteristics of this procedural knowledge representation predict the ease of learning and ease of use of the system (John & Kieras, 1996a, b; Kieras, 2003).

GLEAN (GOMS Language Evaluation And aNalysis Tool (Kieras et al., 1995; Kieras, 1998), is a computational tool for practical GOMS modeling. GLEAN is a simplified cognitive architecture that represents human perceptual, cognitive, and motor mechanisms; by "programming" this architecture with a GOMS model, it is fairly easy to construct a simulated human to perform complex computer-based tasks. The performance of the simulated human in responding to a scenario predicts actual human performance in that scenario. A model of team performance can be constructed by assembling a team of models, in which each simulated human is programmed according to a team job role. GLEAN GOMS models have been developed for watchstanders performing ADW tasks with the MMWS whose development is described in Osga, et al (2002). These models were validated against the performance of real teams in an ADW scenario (Santoro, et al., 2003, Santoro et al., 2000). In general, predicted and actual task duration times and task execution latency times have matched reasonably well, some to better than 10% absolute error.

The purpose of this paper is to demonstrate further the value of this approach in evaluating user interface designs for complex team tasks. That is, the earlier GLEAN models for ADW tasks were developed to account for performance with a first version of the system (called *Build1*), and were compared to some user testing data obtained with human operators. As part of the MMWS project (Osga, et al., 2002), a second version of the ADW support system (called *Build2*) that used a TM interface was designed and prototyped, but user testing was somewhat confounded by the tendency of experienced operator teams to impose their familiar command and communications hierarchy on the exercises. Thus, the extent to which the TM design revision actually improved performance was not clearly determined. In the work reported here, we developed models for Build2 and compared the two builds on the same task scenario strictly following the task procedures for each build. We can report not only that Build2 appears to be a genuine improvement, but thanks to the simulation modeling, we can actually quantify the extent of the improvement. The results further extend what we have reported earlier (Kieras and Santoro, 2004), to show how modeling of human performance can help guide the design and evaluation of user interfaces for complex team tasks.

In what follows, we will describe the interface and the models for the Build1 interface and then the Build2 interface, and then compare them using the performance of one of the simulated human job roles. Since human user testing data was confounded by non-design task performance for Build2, we will compare both models with the expert solution specified by the Subject Matter Experts (SMEs) who developed the test scenario. This is a preliminary report of a subset of the modeling results; future reports will be more complete.

Task and Interface for Build1

The Original MMWS Interface for ADW

The basic task of the ADW team is to maintain situation awareness of the aircraft in the vicinity of a naval task force, and issue queries and warnings to aircraft, or dispatch task force aircraft to perform visual inspections of suspicious aircraft, and finally to engage aircraft if necessary. To perform this task, the team members are seated at workstations viewing a TACSIT display, communicate with each other via a voice intercom and with outside parties (ships or aircraft) via a set of radio channels, one team member per channel.

The MMWS prototype system, shown in Figure 1, was intended to be an improvement on earlier AEGIS system workstations and support the same level of ADW performance with fewer personnel than previously (see Osga, et al., 2002, for a complete presentation). The MMWS incorporates multiple displays, touch screen, keyboard, and mouse input, and speech I/O. However, the core of the ADW task modeled in this work is supported by a single display, shown in Figure 2, containing a color-coded TACSIT display with icons of aircraft and warships, and space for tables of data on a selected track to be presented as a "close control read-out" (CCRO) and other information. If the operator "hooks" (selects) the icon for a track on the TACSIT display, the CCRO shows a variety of numeric and other data on the track, such as course, speed, altitude, etc. The major difference in functionality from earlier systems is that an "auto-ID" function specified by SMEs applies a standard set of rules to automatically identify whether the track is a commercial airliner, a potentially hostile military aircraft, a friendly one, etc. The auto-ID function would set the icon for the track to a shape- and color-coded icon indicating the status of the track. An assumed part of the operator's job is to select the track at some appropriate time (e.g. after it first appeared) and verify that the automatically-assigned ID was correct.

The overall structure of the team's task was to continuously monitor the TACSIT display, watching for changes in the location or behavior of the tracks that call for issuing a query, a warning, a visual id check, or an engagement. Each team member has a role in this process, a subset of the overall task that they are primarily responsible for. In the model team used for Build1, there were four roles designated by acronyms: ADC, TAO, IQC1, and IQC2. In this report, we focus on the role of IQC1, who is responsible for issuing queries and warnings over a radio channel to aircraft whose behavior is of concern. The SMEs supplied a set of rules of engagement that specify when a query or warning should be issued for a track.

Figure 1. The MultiModal Watch Station.

Figure 2. Main display of the ADW interface on the MMWS.

Task and interface for Build2

The Task Manager MMWS Interface for ADW

The Build2 system added an additional display and TM functionality whose purpose was to assist the human operator in keeping track of which tasks needed to be done for which tracks. The underlying functionality was an agent that basically incorporated the SME rules for applying rules of engagement. When a certain set of conditions arose for a track, the TM would create the corresponding task and present it to the operator in the form of an icon in a vertical list of tasks of the same type (see Figure 3). The list of tasks was presented to the operator on the horizontal touch screen display directly in front of the main display. If the operator selects a task icon, the corresponding track on the main display is selected and the CCRO then displays all of its data relevant to the task. In addition, other task-specific functionality would be invoked; for example, a standard query message would be composed using the track data and displayed to the operator, who could then send it as a synthesized text-to-speech message over the radio channel, allowing them to attend to other tasks in the meantime. The task icons remained on the display until the

operator selected them, or until the associated track had disappeared from the TACSIT display.

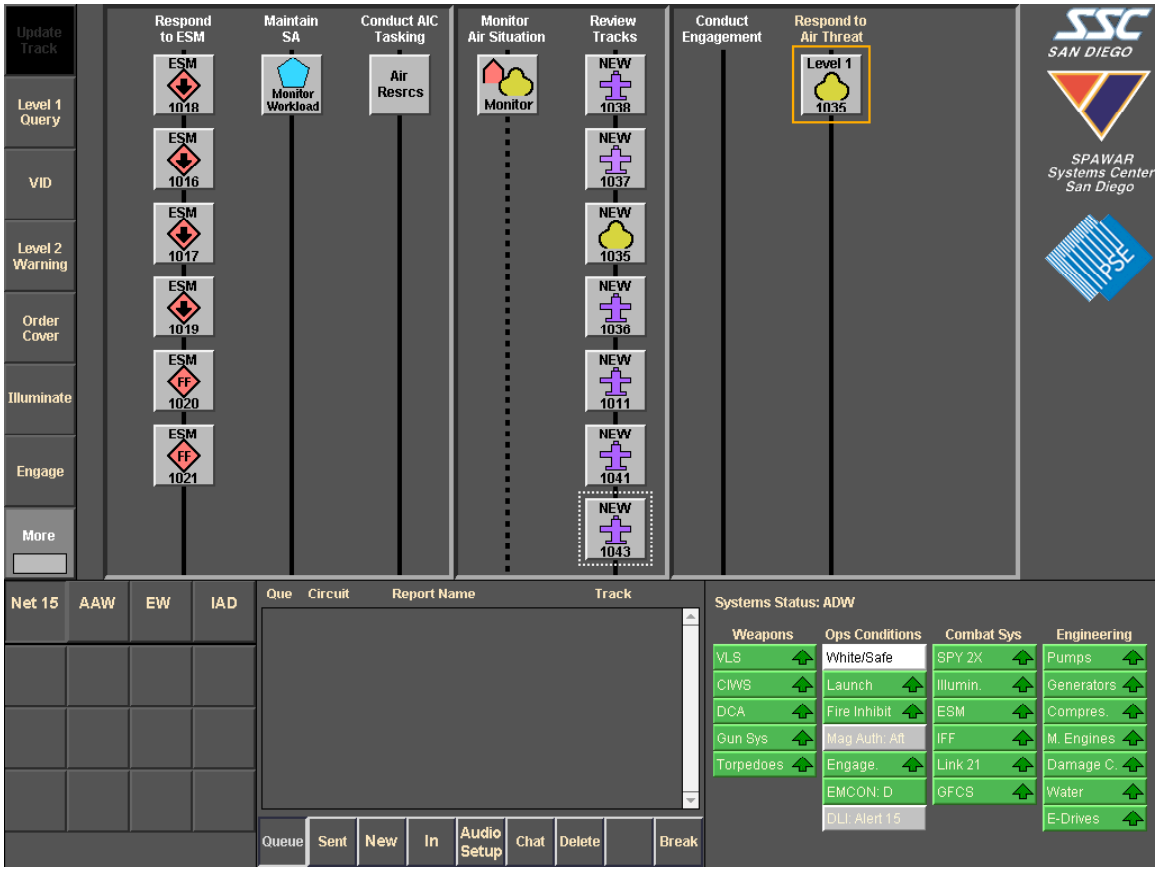

Figure 3. Sample task manager display. Only one air threat task is showing; it is currently selected.

MODELING METHODOLOGY

Watchstation components such as the mouse, keyboard, TACSIT display, CCRO, and various data windows used in ADW operations with the MMWS were simulated with a module in the GLEAN simulation system, programmed in C++ . One of the Build1 workstations was specialized to include displays of ESM (electro-magnetic sensor measures) events. This specialization was included in the simulated workstation. For Build2, the same basic simulated workstation was extended to include the TM

functionality and display. The SME rules of engagement were used to generate new tasks which were added to the simulated display, and, as in the actual interface prototype, remained until either selected or the track disappeared from the display.

The MMWS project developed a single large task scenario that guided the design effort and was used in the user testing. We used a somewhat simplified form of this scenario in the modeling work, which spanned about 1.5 hours of real time, and involved a total of 70 tracks, most of which were simultaneously present on the simulated display. The scenario was a list of about 650 track events, corresponding to appearance, disappearance, and ESM events, and events for course, speed, and altitude changes of the tracks. In the models, the state of the simulated display was updated every 1 sec of simulated time, and the GLEAN architecture itself executes on a grain size of 1 ms of simulated time. The GLEAN models themselves required only a few minutes to run the scenario.

The MMWS project had developed a body of subject-matter expert (SME) opinion on how the task should be conducted in order to conform to some stated rules of engagement. Basically, the GOMS models were simply programmed to carry out the ADW task on the specified interface according to the SME prescriptions. To illustrate the GLEAN models, Figures 4a. and 4b. show samples of the GOMS methods from these models to convey an impression of how the models were written. The first method in Figure 4a. describes how to select a track: first a mouse point, followed by a mouse button click, followed by waiting for the table of track data to appear. The terms <table> and <current_track> are working memory tags - named "slots" that hold the identity of the visual objects currently being examined.

The second method in Figure 4b illustrates some of the decision-making methods. This method examines various visual features of the current track, such as whether it is within 60 miles of ownship (O60 is YES), whether it is inbound (IOB is YES) and decides what action to perform. This action is stored in a tag, <action>, for use by the calling method.

```
Method_for_goal: Hook Track
Step 1. Point_to <current_track>.
Step 2. Click B1.
Step 3. Wait_for_visual_object_whose Label is "Track Data"
          and_store_under <table>.
Step 4. Return_with_goal_accomplished.
```

Figure 4a. Sample GOMSL Method for typical point and click interface procedure.

```
Method_for_goal: Review Track_profile
Step look_back. Store NONE under <action>; Look_at
<current_track>.
Step check_com. Decide:
     If Color of <current_track> is Purple,
          Then RGA.

Step check_tripwires. Decide:

     If O40 of <current_track> is YES,
          Then Store WARN under <action>; RGA;

     If O60 of <current_track> is YES,
          Then Store QUERY under <action>; RGA;

     If O75 of <current_track> is YES,
          Then Store VID under <action>; RGA.

     If <release_range> of <current_track> is WRR,
          Then Store WARN under <action>; RGA.

Step check_IOB. Decide:
          If IOB of <current_track> is YES,
          and <RNG> is_less_than "80",
          Then Store QUERY under <action>; RGA.

Step. Return_with_goal_accomplished.
```

Figure 4b. Sample GOMSL Method for decision-making procedure. RGA is an abbreviation for Return_with_goal_accomplished.

In this project we demonstrated a rather straightforward approach to analyzing team performance: if one can simulate an individual user acceptably well, then one can model a team of such users by setting up a model of each user and having the models interact with each other according to specified team procedures or team strategies. These are simply part of each individual's methods. For example, the GOMS model for the ESM operator specifies that when the operator notices a new ESM event on the workstation display, the operator will announce it by speech over the intercom. The methods for another team member would specify that upon hearing this announcement, the track should be re-evaluated. Like GOMS in general, this team modeling approach would be expected to work well only in highly proceduralized tasks.

Figure 5 shows the overall structure of the team model for Build1. There are four simulated humans, each performing a specific role; three of them are using the basic simulated workstations, while one is using a workstation that includes specialized displays for ESM information. The scenario events are generated by a master device which takes a scenario file as input, and broadcasts the corresponding event information to each simulated device, insuring that the track information is in synchrony, even though the devices will all be in different states as their simulated humans interact with them. The four simulated humans communicate with each other via speech over an intercom channel; a vocal output from one of the operators is broadcast to the other operators as auditory input. Each simulated human also communicates by speech with outsiders over radio channels through their simulated workstations. The key feature is that all of the team interaction takes place via speech interactions over the intercom, while the individual activities of team members take place in interaction with their individual

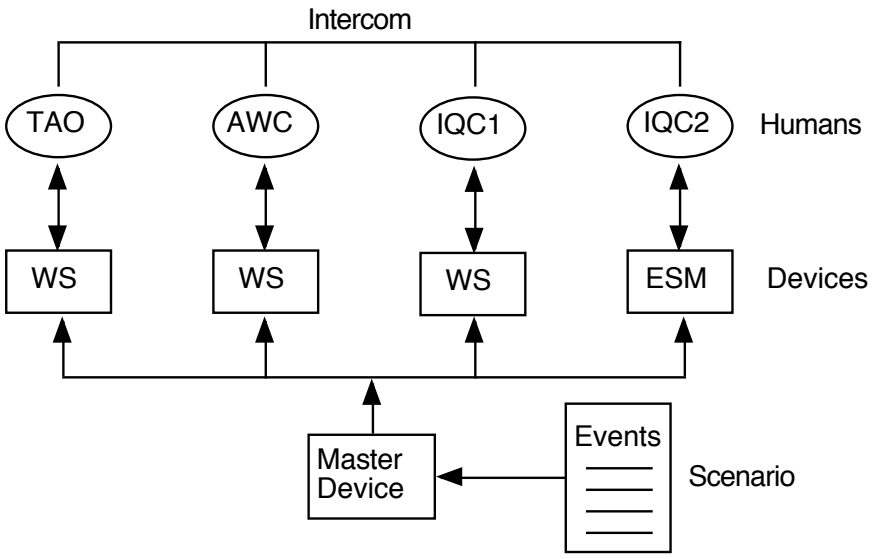

Figure 5. Structure of a team model.

workstations.

With this basic framework, we explored different team designs in terms of whether and how the team members cooperated on the task. Different team organizations were simply represented in the individual GOMS methods that specified when announcements would be made over the intercom, and what actions would be taken in response. Different team procedures produced clear differences in predicted overall team performance. The relative ease of this team modeling approach suggests that it will be very useful; it should be applicable to other types of cognitive-architectural models of human cognition and performance.

Bracketing Models

A major problem in human performance modeling methodology is to choose the task strategy that the simulated humans will follow. Much of this strategy can be based on the task analysis for the task as a whole, but some of the fine details of how human capabilities will manifest in the task are indeterminate given the current state of the psychological theory. One solution to this problem is the *bracketing heuristic* (Kieras & Meyer, 2000). Two models are constructed, one of which represents *best-possible* performance that the cognitive architecture will permit, or perhaps even beyond; the other represents *worst-reasonable* performance, in which the simulated human is assumed to perform the task in a methodical way that meets task requirements but without attempting to be any faster than necessary.

One aspect of performance is whether the operator would notice critical display events immediately or not. In the best-possible models, interrupt procedures triggered by salient screen events would save the events in a list of critical events in memory. Then the model would access this memory store at logical break points in the decision-making process and evaluate the new events to decide if they were worth further monitoring. If so, they would be saved in a list of suspicious tracks with a slow operation that represented either writing down the list item, or committing it to long-term memory.

The worst-reasonable visual search was one in which display events could be noticed and acted upon only part of the time when, for example, the operator was not visually occupied reading a message text or finding a button or icon to click on. The model would make a deliberate decision to search for new and changed air tracks at appropriate points during the overall task process and save them, if necessary, in the same suspicious track list. Of course, a model following this strategy might miss critical events, or not notice what order the events had appeared in.

In more detail, a typical GOMS Operator for visual search would be the following:

> Look_for_object_whose Type is Blip, Color is Red, Status is Changed,
> and_store_under <any_red>.

The GLEAN visual processor would respond to this command by randomly selecting a visual object satisfying the specifications and creating a working memory location containing the label of that object. If no such object existed, the memory location would contain the Absent object. The average total time for this search operation is 500 ms in the GLEAN implementation and 500 ms would be charged to the task execution time when this operator was used.

The Status attribute for visual objects in GLEAN is volatile: it assumes values of New or Changed when appropriate events occur but maintains those values for only 500 ms. Hence, the model operator must interrogate the visual object within 500 ms of an event to recognize that the object had changed Status. This mechanism is used to imitate the visual effect known as change blindness. If an observer monitoring a display of a number of objects looks away, when the display is again inspected, the addition, subtraction, or change of an object may not be noticed. In GLEAN, this behavior is represented by the requirement that object changes be observed within 500 ms of their occurrence.

A second aspect of performance concerned team structure and how it could help ensure that critical events were not missed. In the worst-reasonable model, no such team structure support was present; rather each simulated team member did its own task role independently of the other team members, meaning that whether it missed an event was completely a function of its own individual strategy for noticing events. In the best-possible model, the team members announce critical events over the intercom, and each one adds what it hears to its list of events to be processed. In this way, if one team member is too visually busy to notice the display event, it can still pick it up by audio from another team member who does happen to notice it. Of course, the event might go unnoticed by all team members if they are too busy.

These two performance aspects were combined to produce two interesting models for each version of the interface; specifics of these models are presented next.

Models for Build1

The top-level decision-making process for models not using a TM was built around the 'OODA Loop' (Observe, Orient, Decide, Act; Boyd, 1984) concept. The Observation stage of this process, as captured by the GOMS models, is shown in figure 6.

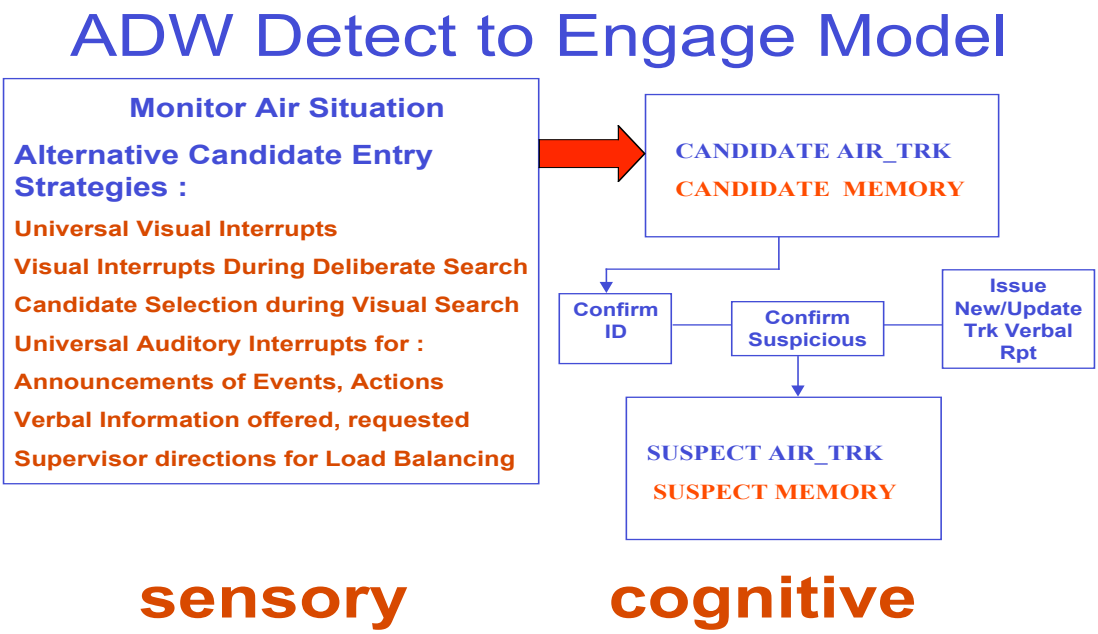

Figure 6. Air Defense Warfare Observation Stage

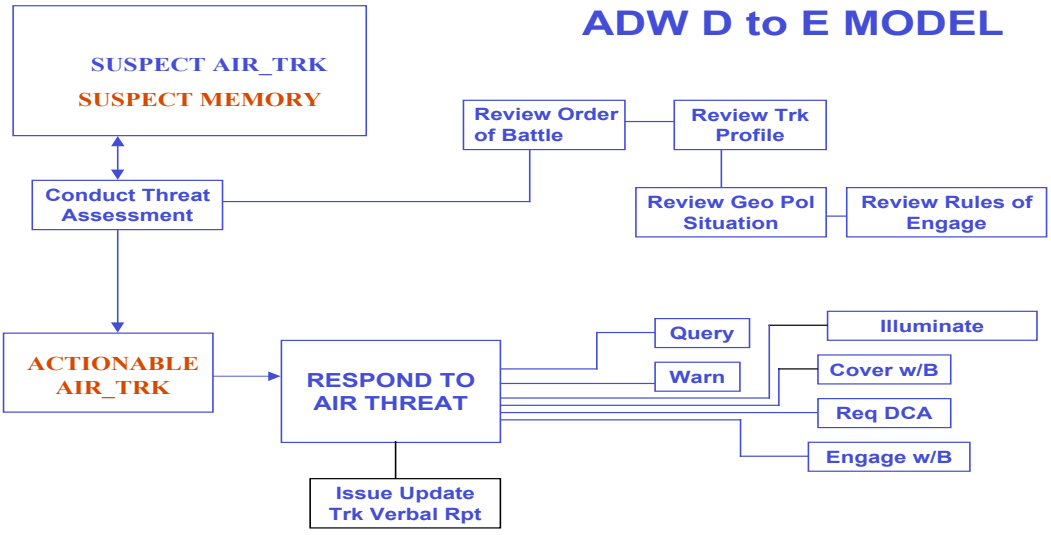

Figure 7. Orientation, Decision, and Action stages of the ADW Model.

According to this sequence, 'Observations' are made by sensory mechanisms interacting with the HCI. Next, in the critical 'Orientation' stage, the meaning of those observations

is interpreted in the context of the on-going tactical situation through assessment processes (Santoro and Amerson, 1998) as shown in Figure 7. Depending on the results of that 'Orientation' stage, a threat assessment is developed that then leads to a 'Decision' being taken on possible 'Actions.' This sequence of processes is iterative with each Decision and Action stage followed by Observations and Orientations which serve to correct and guide the sequence to an acceptable end result.

The success of such a model in meeting the requirements of the 'Expert Solution' for the given ADW scenario depends first on capturing the identities and parameter change events of all the critical air tracks in its Observation Stage and then reviewing the status of those tracks in the Orientation stage at timely intervals, making correct inferences in those reviews that lead to taking the appropriate actions in accordance with the predictions of the 'Expert Solution.' Assuming that correct inferences are always made on available information, failure of an OODA loop-style process would be due to (1) missed or late observations of critical air tracks when they initially become available for entry in the candidate air-track memory and (2) late status review of known hostile air-tracks retained in the suspect air-track memory.

Thus, the GOMS models for the two Build1 teams involve complex sensory and cognitive behaviors.; the example evaluation method in Figure 4 illustrates the complexity of the decision-making process. The OODA Loop decision-making model as well as the visual search mechanisms are attempts to represent such behaviors with the available human performance elements represented in the GOMS model.
It is therefore important to carefully qualify the predictions of these model teams. When the model teams fail to act quickly enough on a critical event, it could be due to other factors besides the limitations of visual search. For example, missed or late actions could be due to the serial nature of the threat evaluation process in examining the contents of the suspicious track memory, which can be quite lengthy at certain times. There are no doubt better ways of selecting a suspicious track from memory than just picking the oldest resident. Real operators may chose to evaluate the closest inbound air track first, or the most dangerous known threat.

Over the several team structure and bracketing levels we explored, two Build1 models are of particular interest and are presented here. The first is a three-member autonomous team (no intercom sharing of events) with a best-possible visual search. The second is a four-member team with the worst-reasonable, deliberate, search and intercom communication of critical events to support the IQC1 query and warning tasks.

Models for Build2

The major contribution of the TM is to supplement the visual and cognitive activities involved with searching for new and changed air tracks on the TACSIT display and determining when scenario events have converged to the point that an actionable task needs to be done. The TM automatically monitors all parameters for air tracks and continuously makes an assessment of possible actionable tasks for each. Hence the

operator need only search the TM display where tasks are listed rather than doing the more complicated visual search and cognitive decision making required to determine when events have given rise to actionable tasks. However, because the TM display also involves visual search, it is certainly possible that the Build2 model teams might fail to notice and act on a task in a timely way.

The two Build2 teams both had only three members and used deliberate search only of the TM display, and differed in whether the team members worked autonomously or involved a team command structure.

The Build2 ADW model teams consisted of an Air Warfare Coordinator, the AWC, and two Information Quality Coordinators, IQC1 and IQC2. These are the stations directly responsible for the majority of all ADW actions. An actual ADW team would have at least two more members; an Air Intercept Coordinator, the AIC, who manages communications with friendly air assets, and a general-purpose supervisor, the Tactical Action Officer or TAO. Only the first three operators identified were modeled in order to test hypotheses concerning their ability to perform all assigned tasks when strictly following the design operating procedures for the TM. GOMS models for the three-member ADW team were programmed to search the TM display for assigned task items, confirm the correctness of the task in question given the current tactical situation, and perform the actions directed in those tasks.

In the first TM model team, each operator was given free rein to select and execute assigned tasks without stating intentions or requesting permission. The AWC made verbal reports on all new or changed air tracks. He directed the AIC to bring in friendly air assets, the defensive counter air, or DCA, to make a visual identification, to intercept and escort, or to engage with weapons, as appropriate, suspect tactical aircraft. Finally, he had to directly engage suspect aircraft or missiles with ownship weapons in the event of an attack. The IQC1 had responsibility for issuing queries and warnings to suspect aircraft. Just as the AWC verbal reports, these off-board radio messages were assumed to be automatically composed text that was transmitted by a text-to-speech mechanism. The IQC1 had to wait for responses from the suspect aircraft and report the results to the other team members over an internal communications circuit. The IQC2 was responsible for reporting all ESM emissions detected from air tracks in the scenario. This information was passed both by text-to-speech on a battle group network and locally over the internal ADW team circuit.

The second model TM team consisted of the same three members with the same task responsibilities but with an additional command structure imposed on the IQC1 by the AWC. This was devised to replicate observations of human teams using the MMWS with (or without) the TM. The human teams typically adapted some form of command hierarchy resembling their usual procedures in actual operations. To test the effect of an additional operator restriction, it was decided to require the IQC1 to only perform queries and warnings when instructed to do so by the AWC. This removed the decision-making responsibility from the IQC1 and imposed it on the AWC. He selected and confirmed

query and warning tasks and then directed the IQC1 to perform them and report back on the responses from the suspect aircraft.

The Build2 teams have relatively straight-forward task procedures to follow, thanks to the presence of the TM, the team members can work with little or no interaction.

The top level task of the IQC1, operating independently, is to monitor the TM display for threat response tasks. The TM display column for threat tasks includes queries and warnings, together with other tasks. The obvious strategy for IQC1 would be to frequently check for a query or warning task in the list, click its icon and proceed as directed. The TM will then open windows for all the pertinent information on the air track in question including color-coded track parameters that support the action and others that are counter-indicators for the action. The model operator is programmed to do a complete threat evaluation of this information for the purpose of confirming that indeed the indicated action is correct. This includes a review of the current rules of engagement and the available order of battle information on the suspect aircraft.

For queries and warnings, the TM will present a machine-generated text containing the verbiage of the message to be sent to the air track. Again, the IQC1 has to simply inspect the text for correctness and then click on the send button to initiate a text-to-speech processor that handles the actual transmission. He is free to inspect the TM or other visual displays as desired while the message is being sent. Next, the IQC1 has to wait some appropriate time for the aircraft to respond. A number of the critical air tracks have preprogrammed response that will be returned to the operator for his evaluation. Other air tracks make no response. The operator will make an evaluation of the response or lack of response and announce this finding to the rest of the ADW team. He also enters an update report of his actions and results into the system that is available for examination by the team at any time.

RESULTS

The data for Query actions and Warning actions are presented in Table 1. The table shows the percentage of actions called for by the Expert Solution that were taken by each model, taken within the time period allowed by the Expert Solution, and the time delay in taking the action, measured from the earlier time when the Expert Solution calls for it - the assumption is that the sooner the action is taken once called for, the better.
These actions were chosen for discussion because they require prompt threat evaluations on new and changed tracks and periodic monitoring of all suspect tracks throughout the time course of the scenario. While they are the primary responsibility of one team member, the IQC1, observations from exercises and video indicate that these actions are the subject of considerable discussion and (probably) workload sharing among team members in exercises with human teams.

	Performed	Within time	Avg. delay (s)
Queries - 12 tracks			
Build 1			
Autonomous team, best search	75%	58%	423
Cooperating team, deliberate search	67%	58%	367
Build 2			
Autonomous team	100%	75%	237
Commanded IQC1	100%	75%	284
Warnings - 10 tracks			
Build 1			
Autonomous team, best search	100%	100%	164
Cooperating team, deliberate search	90%	100%	189
Build 2			
Autonomous team	100%	90%	117
Commanded IQC1	100%	50%	185

Table 1. Results for the models of IQC1 actions for the scenario. Shown are the percentage of actions called for by the Expert Solution that were performed by the models, performed within the time window called for by the Expert Solution, and the average time delay after the beginning of the Expert Solution time window.

The results show that overall, the Build2 teams are generally much better at performing the required actions, doing so both more quickly and staying within the called-for time limits more often. Moreover, the Autonomous teams, with one exception, were faster than those that had inter-operator communications. While the Build1 autonomous team was slower than the collaborative team for query actions, it did execute more of the required actions than the collaborative team. The autonomous IQC1 model in Build2, in particular, performed very well on the whole. However, when a command structure with the AWC was introduced, as occurred in real exercises with Build2, performance for this model operator dropped back to levels equivalent to that in Build1.

Cases where the Build2 model slightly exceeded the allowable times for warnings probably result from the fact that warnings and queries were posted in chronological order on the TM display and were thus selected in order by the IQC1. The Build1 IQC1, on the other hand, searched for close-in air tracks first and might thus pick up a warning task a bit earlier, if it was observed at all. Averaging the two types of actions and the two teams for each Build together, the Build2 design results in a 17 percentage point improvement in actions performed, and a 24% improvement in action delay. Explaining these results more thoroughly will require further work, along with other aspects of the

model team performance, but the conclusion is that the TM interface, used as designed with autonomous operators, makes a significant improvement to operator performance.

CONCLUSION

The results provide reasonable support for the success of a TM interface. Of course, it is possible that there is some model or human team that might do as well without a TM, by relying on better visual search or more sophisticated coordination via speech interaction than our rather simple model represented. However, one thing to note is that such interaction does not seem to be required for the TM design to obtain superior levels of overall performance to the Build1 interface for some cases and, in fact, seems to reduce performance in other cases. Future work with such team models should focus on trying to arrive at a better understanding of the collaborative behaviors involved in execution of a hierarchy of tasks in parallel by several operators. The TM interface, in the models examined thus far, is seen to be an effective replacement for such collaboration when it involves continuous monitoring of multiple streams of constantly-changing information, a task well-suited to automation.

ACKNOWLEDGEMENTS

This work was supported under Work Unit number 62233N-R3322-50214 from the Naval Air Warfare Center, Orlando, FL and by the Space and Naval Warfare System Center (SPAWAR), San Diego, CA, Code 246210 under funding from the Office of Naval Research, Cognitive, Neural, and Social Science & Technology Division. Research Area: Decision Support Systems and Models for Intelligent Mission Management.

REFERENCES

Boyd, J.R. (1984). Organic design for command and control. Unpublished briefing paper, pp. 5, 32-35.

Card, S., T. Moran, et al. (1983). The psychology of human-computer interaction. Hillsdale, NJ, Lawrence Erlbaum Associates, Inc.

John, B. E., & Kieras, D. E. (1996a). Using GOMS for user interface design and evaluation: Which technique? *ACM Transactions on Computer-Human Interaction*, 3, 287-319.

John, B. E., & Kieras, D. E. (1996b). The GOMS family of user interface analysis techniques: Comparison and contrast. *ACM Transactions on Computer-Human Interaction*, 3, 320-351.

Kieras, D.E. (1998). *A guide to GOMS model usability evaluation using GOMSL and GLEAN3*. (Technical Report No. 38, TR-98/ARPA-2). Ann Arbor, University of Michigan, Electrical Engineering and Computer Science Department. January 2, 1998. Current version available online at http://www.eecs.umich.edu/people /kieras /GOMS/GOMSL_Guide.pdf.

Kieras, D.E. (2003). Model-based evaluation. In Jacko, J.A. & Sears, A. (Eds) *The human-computer interaction handbook*. Mahwah, New Jersey: Lawrence Erlbaum Associates, pp. 1139-1151.

Kieras, D. E., & Meyer, D. E. (2000). The role of cognitive task analysis in the application of predictive models of human performance. In J. M. C. Schraagen, S. E. Chipman, & V. L. Shalin (Eds.), *Cognitive task analysis*. Mahwah, NJ: Lawrence Erlbaum, 2000.

Kieras. D.E. & Santoro, T.P. (2004). Computational GOMS Modeling of a Complex Team Task: Lessons Learned. In *Proceedings of CHI 2004: Human Factors in Computing Systems*. New York: ACM, Inc.

Kieras, D.E., Wood, S.D., Abotel, K., & Hornof, A. (1995). GLEAN: A Computer-Based Tool for Rapid GOMS Model Usability Evaluation of User Interface Designs. In Proceeding of UIST, 1995, Pittsburg, PA, USA. November 14-17, 1995. New York: ACM. pp. 91-100.

Osga, G., Van Orden K., Campbell, N., Kellmeyer, D., and Lulue D. (2002). Design and Evaluation of Warfighter Task Support Methods in a Multi-Modal Watchstation. Space & Naval Warfare Center, San Diego, Tech Report 1874, 2002.

Santoro, T.P., and Amerson, T.L. Definition and measurement of situation awareness in the submarine attack center. *Proceedings of the Command & Control Research & Technology Symposium*, pp. 379-388, Monterey, June, 1998.

Santoro, T.P., Kieras, D.E., and Campbell, G.E. (2000). GOMS modeling application to watchstation design using the GLEAN tool. *Proceedings of the Interservice/Industry Training, Simulation, and Education Conference*, pp. 964-973, Orlando, FL. Nov, 2000.

Santoro, T.P, Kieras, D., & Pharmer, J. (2003). Verification and validation of latency and workload predictions for a team of humans by a team of computational models. (Tech. Rep. No. TR1227). Groton, CT, Naval Submarine Medical Research Laboratory. May 1, 2003. Also available at:
`ftp://www.eecs.umich.edu/people/kieras/GOMS/Santoro_et_al.pdf`