Developing User Model-Based Intelligent Agents

Alonso H. Vera and Julio K. Rosenblatt
Hughes Research Labs
3011 Malibu Canyon Road
Malibu, CA 90265
vera@hkucc.hku.hk, jkr@cmu.edu

Abstract

We describe a GOMS model of a ship-board Radar Operator's behavior while monitoring air and sea traffic. GOMS is a technique that has been successfully used in Human-Computer Interaction to generate engineering models of human performance. Based on the GOMS model developed, we identified those portions of the task where an intelligent agent would be most able to assist operators in the performance of their duties, and the nature of the knowledge that will be required for the task. We present the results of a simulated execution of the model in a sample scenario, which predicted the operator's responses with a high degree of accuracy.

Introduction

Our goal is to determine the domain knowledge required to implement an intelligent agent which assists a human user in performing a computer-based task. A central characteristic of intelligent behavior is that it is purposeful, i.e., the agent is executing a task in order to achieve a goal. Consequently, to help a user accomplish a goal, an intelligent agent must have knowledge about that goal. GOMS is a technique that has been used in the study of human-computer interaction to model user knowledge and behavior at various levels of description. We investigate here how GOMS models may be used by intelligent agents as a means of understanding the actions of other agents and users so that it may interact and cooperate with them in a consistent and helpful manner.

A GOMS model consists of a set of Goals, Operators, Methods, and Selection rules necessary to accomplish a particular task. A hierarchy of goals is created, and within that hierarchy a set of methods provide a functional description of a task. Selection rules distinguish between various operational cases and account for the idiosyncrasies of individual users. Operators provide a low level description of the actions finally performed. Thus, GOMS provides a uniform structure for representing the intentional, functional, and implementational levels of behavior. A GOMS model of a particular task might be used by an intelligent agent to understand the task at hand and the current state within that task. Acting alone, the agent can use the model to decide the next action; acting as an assistant, the agent can use the model to understand what other the user is doing and tailor its actions and recommendations appropriately.

In this paper we describe a GOMS model of Radar Operators monitoring air and sea traffic on board a ship. The task is very interactive, between the operator and other members of the crew, as well as between the operator and the radar console itself. The work presented here uses a methodology that was originally developed to address routine expert behavior on non-interactive tasks; recent work has indicated that this methodology yields excellent results when applied to interactive tasks as well (John, Vera and Newell, 1994; Gray, John and Atwood, 1993; Endestad and Meyer, 1993). Our study of the Radar Operator's task shows that it has a large amount of routine content, even when things get busy.

A Model of a Radar Operator

We have created a model of the radar operation task by decomposing the Radar Operator's actions into goals, operators, methods, and selection rules (GOMS) as first proposed by Card, Moran and Newell (1982). A GOMS model begins with the concept of a top-level goal which the user seeks to achieve, and a series of unit tasks that the user performs repeatedly until there are no tasks left. The classic example is that of a typist using a word-processor to make corrections that have been marked on a printed copy of a manuscript; the top-level goal is to edit the on-line version of the manuscript, and the unit task is simply to make the next correction (Card et al, 1982).

We have written the GOMS model using NGOMSL, or Natural GOMS Language, (Kieras, 1988; 1994). It takes the basic precepts of GOMS and defines a programming language based on those ideas, allowing creation of a model where flow of control is clearly specified and which in principle could be run on a computer; indeed, a compiler is currently being written for that purpose. The process of fully specifying the Radar Operator's decisions and actions guided us to create a model that is complete and accurate to the level of detail in which it is specified, and aided us in the task of knowledge acquisition as well; it also pointed out those places where there were gaps or inconsistencies in our knowledge of the domain.

Structure of the Model

One of the first and most basic decisions that had to be made was how to define the *unit task* for the Radar Operator; the organization of the top-level goal and the unit task would affect the entire structure of our GOMS model. In the case of text editing, the structure was easily defined by making each marked correction a unit task. The analogous decomposition in our domain would be to define the unit task as tracking each object on the radar screen. However, one obvious reason that such a scheme could neither model the Radar Operator's behavior accurately nor provide a reasonable framework for defining a system is that the task of tracking an object has no well-defined end; the task might never be completed and all other radar contacts would be ignored as a result. In addition, the Radar Operators must also receive and respond to orders which may arrive at any time, so they must be incorporated into the definition of a unit task as well.

In search of a better definition of the unit task, we turned to the training manual used by the Radar Operators (*Operations Specialist Training Manual*), where we found this statement: "Information handling comprises five major functions gathering, processing, displaying, evaluating, and disseminating information and orders." We attempted to use these five functions for our unit task structure, but our efforts were complicated by another emergent task structure; as we examined a sample Radar Operator scenario, it became apparent that the Radar Operator went through various stages of identification for each new contact: establishing tentative track, air or surface, commercial or military, friendly or hostile, and so on. The challenge was to create a goal structure that persisted in the incremental acquisition of knowledge about a given tracked object while still providing reactiveness to new information and situations and as they developed. The manual continues, "All information handling must be considered a continuous and growing process that ultimately furnishes a composite picture of a situation, enabling the commanding officer to make a final evaluation and give orders for action."

The solution we ultimately decided upon was to select a contact, determine which stage of identification should be performed next, and go through the steps of gathering, processing, displaying, evaluating, and disseminating information on this fine grained unit task. Once done with a particular stage of the identification process on a particular contact, the model returns to the top-level, where new orders may be received and acted upon or a task that has become more urgent may be selected for execution. As in John and Vera (1992), a relatively shallow goal stack and carefully designed set of selection rules was used to make the model reactive to external changes. The resulting goal and method hierarchy for the intentional and functional levels of the task is shown in Figure 1.

Knowledge Content of the Model

As can be seen in Figure 1, there are three sets of selection rules in our GOMS model. These correspond to points in the execution of a unit task where a decision must be made as to how to proceed because there are multiple methods to accomplish a goal. The first selection rule, Select Next Task, simply chooses the Execute Order method if a new order has been received, otherwise the method for Monitor Radar Contacts is selected. If an order is to be executed, Execute Ordered Task selects the method that is appropriate for carrying out that order. If no order has been received, then the Execute Unit Task selection rule must decide, for a given contact, which is the appropriate subtask; this depends on what information has already been gathered about that contact, which is reflected by the current label assigned to it. The corresponding selection rule, written in NGOMSL, is as follows:

Selection rule set for goal: Execute Unit Task
If <contact-label> is New and contact is under local control, then accomplish goal: Establish Tentative Track.
If <contact-label> is Tentative Track and contact is under local control, then accomplish goal: Establish Air or Surface.
If <contact-label> is Unknown and contact is under local control, then accomplish goal: Establish Friend or Foe.
If <contact-label> is not Tentative Track or Unknown and contact is under local control, then accomplish goal: Update Contact Information.
If contact is not under local control, then accomplish goal: Update Contact Information.
Return with goal accomplished.

This selection rule uses perceptual information that is available to the Radar Operator, in order to choose the appropriate method. This is true of the other two sets of selection rules as well. Very little beyond the ability to understand symbols and orders is encapsulated in the selection rules. The Radar Operator's knowledge is contained in the methods of the model.

Within the GOMS model of a Radar Operator, the goal hierarchy captures the structure of the decision-making process involved; its topology reflects knowledge of the nature of the task and the control knowledge needed to carry it out. The detailed knowledge which makes these decisions and the achievement of goals possible is embedded in the methods of the model. For example, the Establish Air or Surface method describes the steps that are taken to achieve that goal, including the specific steps the Radar Operator must take to make the determination:

Method for goal: Establish Air or Surface
Step 1. Accomplish goal: Gather and Process Movement Information.

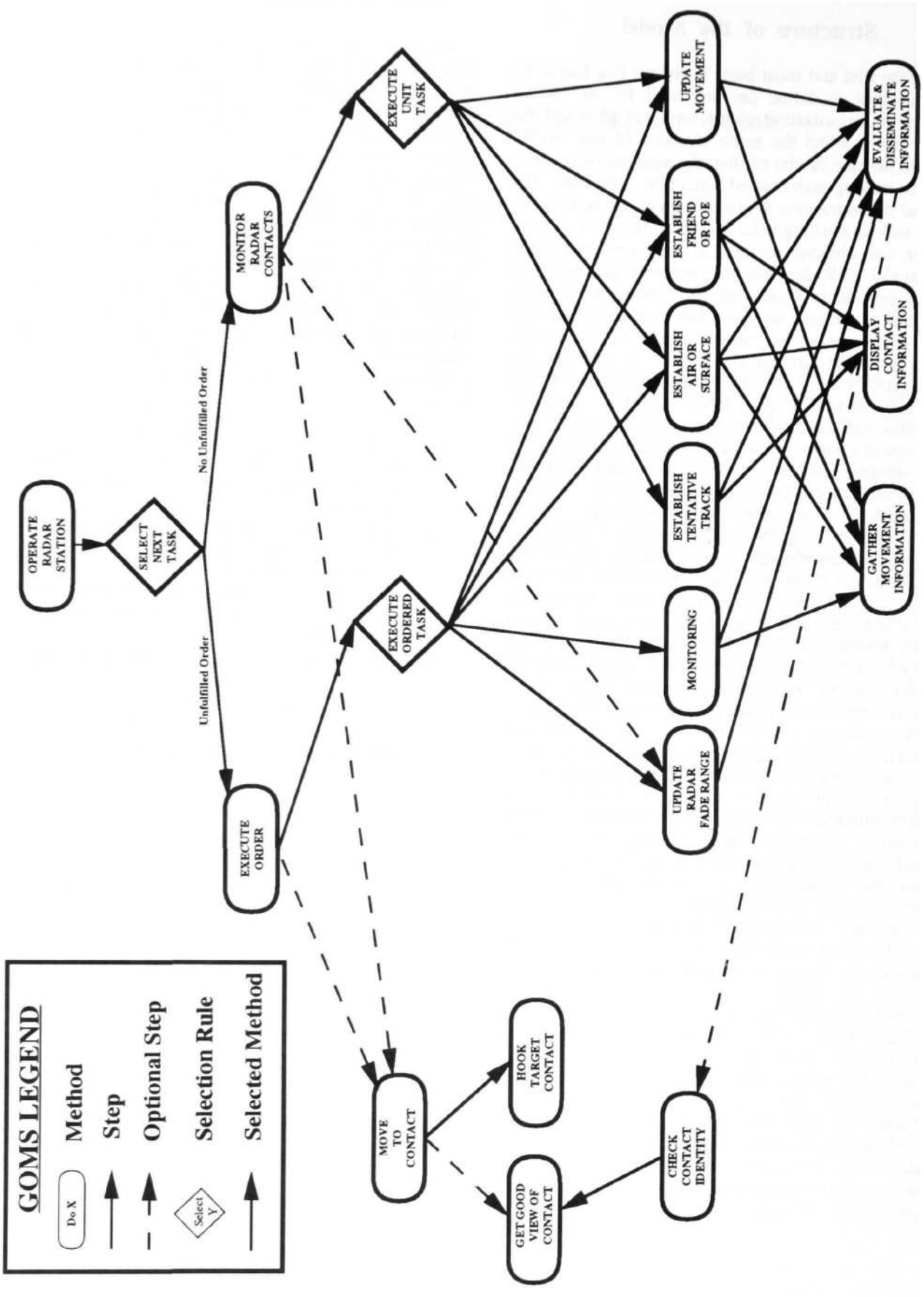

Figure 1. Radar GOMS Goal Hierarchy

Step 2. Decide: If contact is not a track,
then remove tentative track and return with goal accomplished.
Step 3. Decide: If contact type is determined to be Air,
then Accomplish goal of: Display Contact Information Unknown Air.
If contact type is determined to be Surface,
then Accomplish goal of: Display Contact Information Unknown Surface.
If contact type is not determined,
then return with goal accomplished.
Step 4. Accomplish goal: Evaluate and Disseminate Information.
Step 5. Return with goal accomplished.

The implementational (or keystroke) level is created by further decomposing the structure of the task into primitive operators, such as reading text, pointing and clicking the mouse, etc. For example, the first step in the Establish Air or Surface method is to invoke the subgoal Gather and Process Movement Information, implemented by the following method whose steps consist of operators that are not analyzed further:

Method for goal: Gather and Process Movement Information
Step 1. Decide: If contact is under local control,
then perform position correction.
Step 2. Read contact information.
Step 3. Decide: If CPA needs to be evaluated,
then compute contact CPA.
Step 4. Process contact information.
Step 5. Return with goal accomplished.

Reactivity of the GOMS Model

When developing a model of a dynamic, real-world task, it is of critical importance to capture and retain the qualities that allow humans to perform the task as well as they do. One of the key characteristics of human performance in this task is reactivity. The operators in the scenario are able to quickly react to new contacts and respond to new orders. They can stop whatever task they are doing, begin a new one, execute it, and return to the original task. The GOMS model presented here must be able to reproduce this reactivity in a natural way that results in sequences of behaviors like those of the operators in the scenario.

The GOMS model of the operator derives its reactivity from the organization of its methods and selection rules. At any given moment in the model's behavior, the goal stack is relatively shallow because there are very few chained methods that get called as a sequence. Reactivity is not achieved by forcing the model to check for changes in the world (e.g., new orders or contacts) within each method, but instead by returning to the top-level goal after completing a portion of prioritized sub-tasks. The top-level method can then check for changes in the world.

Avoiding long linked sequence of methods allows the model to check for important changes in its environment in a way that does not overload working memory nor unnecessarily interrupt routine behaviors. The model was designed in such a way as to simulate the operator's sequence of contact-identifying behaviors while remaining sensitive to changes that affect its goal prioritization. It is thus able to combine the routine collection of information about contacts, detecting new targets, and executing orders in a cognitively plausible way.

Using Predictive Models for Agent-Assisted Decision-Making

The goal hierarchy of the GOMS model captures the structure of the decision-making process involved in performing the task; its topology reflects knowledge of the nature of the task and the control knowledge needed to carry it out. Additional knowledge which makes these decisions and the achievement of goals possible is embedded in the methods of the model. We propose that an intelligent agent must embody, at least in part, a model of the user, and that GOMS models provide a suitable structure for this purpose. The GOMS model described here indicates that there are at least three places where the user performs complex cognitive operations that might be assisted by an intelligent agent: evaluation of contact information, selection of most relevant contact, and checking of contact identity. These are parts of the operator's task that require decision-making based on both the current situation and experience-based knowledge.

These cases represent those aspects of the task where an intelligent agent, armed with a model of the operator's knowledge and behavior, can make significant contributions by knowing the current goal of the operator. A GOMS model provides a dynamic representation of users' goals, knowledge, and priorities, as well as their behavior. At any given point in the problem-solving process, there exists the current goal stack; associated with each goal is a method for achieving that goal, consisting of a series of primitive perceptual, cognitive, and motor operators that are to be executed. Using this knowledge, the intelligent agent can anticipate the information the operator will require to accomplish his goals, present that information in a useful form, and recommend actions based the information. Furthermore, the agent may use its model of the operator's knowledge to determine what task the operator is currently performing. If the operator diverges from the behavior predicted by the model, the agent will try to map the new behavior onto the model. The agent can then assess whether the operator's divergent behavior is warranted by the current situation. If it is not, the agent can recommend alternative courses of action that the operator should follow. If the operator's behavior is warranted by the situation, then the agent can update its location on the model to continue assisting the operator on the appropriate task.

When selecting the next task to be performed, the agent can estimate the priority of processed information in order to focus the radar operator's attention on the most significant and urgent developments and to adapt to the operator's priorities when they diverge from those predicted by the model. As the operator evaluates new information, the agent must update its model of the operator's beliefs. The agent can anticipate the information the operator will require to accomplish her goals, present that information in a useful form, and recommend actions based the information.

Monitoring Working Memory Load

Due to their procedural nature, GOMS models also provide the means to predict a user's working memory load. At any given point in the problem-solving process, there exists the current goal stack; associated with each goal is a method for achieving that goal, and each method explicitly states what information it accesses and must therefore be stored in working memory for the duration of that method's execution. For example, referring to the model shown in Figure 1, the operator begins execution of the Operate Radar Station method, where she must retain whether or not a new order has been received, and if so what that order is. Within the selection rule Select Next Task, the operator must then choose which task she will perform next, and remember that information, and so on. Table 1 shows a trace of which variables are retained within each method, and the total working memory load at that point in the goal stack.

Table 1: Working Memory Load during task execution.

Method/Rule	Variables Retained	WM
Operate Radar Station	(order = nil)	0
Select Next Task	none	0
Monitor Radar Contacts	target_contact = Casper	1
Move to Target Contact	contact_label = Unknown	2
Execute Unit Task	none	2
Establish Friend or Foe	none	2
Gather and Process Movement Information	bearing = 24.5 range = 32 altitude = 30,000 speed = 500 heading = 69.3	7
Display Contact Information <new_label>	<new_label = Hostile> (contact_label = Hostile)	8
Evaluate and Disseminate	none	7

In the present scenario, the operator may often be in a situation where her working memory is overloaded. For example, if there are many new contacts as well as orders to be executed, the operator may not be able to retain the necessary sequence of behaviors. Using the model to evaluate situations where the operator's working memory capacity might be overwhelmed, it is possible to predict when behavior may deviate from what is expected, and to determine when an agent can be most effectively assisted.

Comparison of Predicted versus Actual Radar Operator Behavior

A simulated execution of the GOMS model was conducted on a test scenario; this resulted in a sequence of behaviors that the model would perform if it were operating the radar station. We then compared this sequence of behaviors with that of the operators in the scenario. Table 2 summarizes the results of this analysis.

Table 2: Summary of Comparison Between Model and Operator Behavior

SCENARIO \ MODEL	Match	Miss	
Match	54	4	T=58
Miss	107	NA	
	T=161		

There were 58 distinct operator behaviors described in the original scenario. The model generated a total of 161 behaviors in total and matched 54 of the 58 operator behaviors. In order to be considered matching, the model must generate not only the same behaviors as the operator, but it must also do so in the same order. Model behaviors that occurred out of order were counted as mismatches. Of the 4 operator behaviors that were not matched by the model, 2 of them were behaviors that the model performed implicitly. That is, the model, as we built it, did not explicitly perform these actions as independent methods, but instead had them built in to other methods. This was not an important design decision on out part but simply the consequence of the granularity level chosen for these methods. Of the other two actions that were not accounted for, one involved changing the type of radar used and simply was not included in our model, and the other involved a non-routine reporting of information. Discounting the behaviors performed implicitly by the model, only 2 (3.5%) of the operator's behaviors were not matched by the model.

The model generated 107 behaviors in addition to the 54 that matched those of the operator. Although this is a large number of extra behaviors, a case by case analysis shows that 104 of them are behaviors that are implicit in the scenario. That is, they are behaviors that were necessarily performed by the operator, but that were not explicitly described in the scenario. For example, when the model selects a new contact to establish a Tentative

Track, it must necessarily execute an intermediate method of moving to the contact and then hooking it. The operator must also perform this sequence of behaviors, but the full set of steps is not explicitly described in the scenario.

The remaining 3 behaviors produced by the model that were neither matches nor implicit in the scenario were behaviors that probably should have been done by the operator but were left out because of time or memory constraints. As discussed, the GOMS model provides a measure of working memory usage that can be used to better predict the mental states of an agent by taking these constraints into account. Steps within methods that are not strictly necessary can be noted as optional, so that non-deterministic behavior may be accounted for. Overall, only 3 out of 161 (less than 2%) behaviors generated by the model were neither matches nor implicit when compared to the operator's behavior in the scenario.

The model predicted 96.5% of the operator's behaviors. Furthermore 98% of the behaviors generated by the model were either explicitly or implicitly present in the scenario. These results indicate that the model is successfully simulating the operator's behavior and therefore capturing the knowledge required to perform the task.

Conclusion

Using the GOMS methodology for analysis of human-computer interaction, we have developed a model of a Radar Operator's goals, and the methods that used to accomplish them. A simulated execution of the model in a test scenario predicted the operator's responses with a high degree of accuracy, and furthermore provided details of those actions that were not explicitly stated in the scenario description. Based on this model, we were able to identify those portions of the task where an intelligent agent would be most able to assist the operator and to describe the nature of the knowledge required for the task.

By creating a knowledge-level description of a task (Newell, 1982), GOMS models provide the means to predict users' goals, beliefs, priorities, and, therefore, their actions. When the user's actual behavior departs from what the model has predicted, the agent may inform the user of the unexpected actions and recommend an alternate course of action. If the operator's behavior is warranted by the situation, then the agent can update its model or its parameters so that the agent may continue assisting the operator on the appropriate task.

Acknowledgements

This work was conducted while both authors were at Hughes Research Labs, sponsored by the Information Sciences Laboratory at HRL under the guidance of Mike Daily and David Payton. A. Vera is currently at The University of Hong Kong and J. Rosenblatt is at Carnegie Mellon University. The opinions expressed in this paper are those of the authors and not necessarily those of Hughes.

References

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

Endestad, T. & Meyer, P. (1993). GOMS analysis as an evaluation tool in process control: An evaluation of the ISACS-1 prototype and the COPMA system. Technical Report HWR-349, OECD Halden Reactor Project, Instituut for Energiteknikk, Halden, Norway.

Gray, W. D., John, B. E. & Atwood, M. E. (1993). Project Ernestine: A validation of GOMS for prediction and explanation of real-world task performance. Human Computer Interaction, 8, 3, pp. 209-237.

John, B. E. & Vera, A. H. (1992). A GOMS analysis for a graphic, machine-paced, highly interactive task. In Proceedings of CHI (Monterey, May 3-7, 1992) ACM, New York, pp. 251-258.

John, B. E., Vera, A. H. and Newell, A. (1994). Toward real time GOMS: A model of expert behavior in a highly interactive task. Behaviour and Information Technology, 13, 4, pp. 255-267.

Kieras, D. E. (1988). Towards a practical GOMS model methodology for user interface design. In M. Helander (Ed.), Handbook of Human-Computer Interaction (pp. 135-158). Amsterdam: North-Holland Elsevier.

Kieras, D. (1994). GOMS Modeling of User Interfaces Using NGOMSL. Tutorial Notes, CHI Conference on Human Factors in Computing Systems, Boston, MA, April 24-28.

Newell, A. (1982). The knowledge level. Artificial Intelligence, 18, 87-127.

Operations Specialist 3 & 2, Vol. 1, Navy Training Manual.