CONTEXT IN MODELS OF HUMAN-MACHINE SYSTEMS

Todd J. Callantine

San Jose State University/NASA Ames Research Center

Abstract: All human-machine systems models represent context. This paper proposes a theory of context through which models may be usefully related and integrated for design. The paper presents examples of context representation in various models, describes an application to developing models for the Crew Activity Tracking System (CATS), and advances context as a foundation for integrated design of complex dynamic systems.

Keywords: Models, Modeling, Man-Machine Systems, Design, States, Events, Constraints

1. INTRODUCTION

Models are powerful tools for the analysis, design, and evaluation of human-machine systems. However, their utility depends on the fidelity with which they represent interactions among humans, machines, tasks, and the environment, and the effort required to develop and apply them effectively. These factors have led to the development of a variety of models that address one or more facets of analysis, design, and evaluation of a particular human-machine system, or subsystem. Some models focus on machine behavior to better understand machine responses to human and environmental inputs and design displays, and ensure that a system will operate safely (e.g., Degani, and Heymann, 1998; Feary, *et al.*, 1997; Leveson *et al.*, 1997; Sherry, 1995). Others focus on operator activities to design and analyze interactions, and provide the foundation for analysis tools, training systems, and interface designs that incorporate knowledge-based displays and aiding functionality (e.g., Callantine, 1996; Callantine *et al.*, 1997, 1998; Funk and Lind, 1992; Mitchell, 1996, 1998).

These and other successful models prescribe and/or describe, with sufficient coverage and at appropriate levels of abstraction, the salient behaviors of relevant system elements in a form that supports both computational application(s) *and* communication between designers and practitioners in the domain of interest (cf. Heimdahl, *et al.*, 1997; Mitchell, 1996). As the complexity of systems and associated design efforts grows, it has become increasingly important to develop multi-purpose, computational models that meet these requirements and effectively support the design process. A new air traffic management system, for example, involves numerous designers and practitioners collaborating to concurrently develop new cockpit interfaces, air traffic control automation, and procedures—to name a few—all of which have extensive organizational impacts.

Integrated design on a large scale necessitates ways to effectively *combine* a variety of models with different focuses and forms, so that designers and stakeholders can use models suited to their areas of expertise to facilitate the design process and operation of the resulting system (cf. Vakil and Hansman, 1998). Models 'reverse-engineered' from available technical information about a new piece of technology can foster discrepancies in the resulting operator procedures and training (Smith and Moses, 1998). Instead, a collection of models should evolve together, creating a record of the design process that 'lives' in the design.

One possible solution lies in a feature common to all successful models, regardless of their focus or form: a means of representing context. Context is a central tenet of human-machine systems engineering (Mitchell, 1996). From the perspective of a given element in a human-machine system, context can be thought of as the relationship of the state of the given element to the dynamic collection of constraints imposed by other elements. Models of human interaction with complex systems, for example,

typically represent operator activities together with conditions, or rules, that indicate the context in which the representation prescribes and/or describes operator task performance. Similarly, machine models represent the conditions under which an operator intervention results in a specific machine behavior. However, modeling efforts have attended more to the preferred units of analysis, coverage, and level(s) of abstraction for operator or machine behaviors than to the associated context. Context representations, while 'explicit,' are seldom hierarchical and lucid from various perspectives.

The thesis of this research is that, within a domain, models with different focuses and forms may be usefully related through context, and that context can provide the foundation from which different models can evolve along with a design while maintaining consistency. This paper proposes a theory of context to investigate these assertions. The theory posits classes of domain-specific context information can be combined to explicitly represent context in a general form, at multiple levels of abstraction. The theory arose from research on the Crew Activity Tracking System (CATS), which uses a model of correct task performance to predict operator activities and interpret operator actions in real-time (Callantine, 1996). A CATS model uses logical equations of Boolean-valued "context specifiers" to represent when an operator activity should be performed, as shown in Fig. 1. Similar expressions are used for other representing other conditions, such as when an activity is no longer applicable.

The paper gives examples of context representation in human-machine systems models, then describes the theory. Next, the paper presents a prototype modeling tool, called the CATS Modeler, that embodies the proposed theory. The CATS modeler enables domain-specific knowledge required by a CATS model to be specified and visualized graphically. The paper concludes with remarks on potential benefits of the theory.

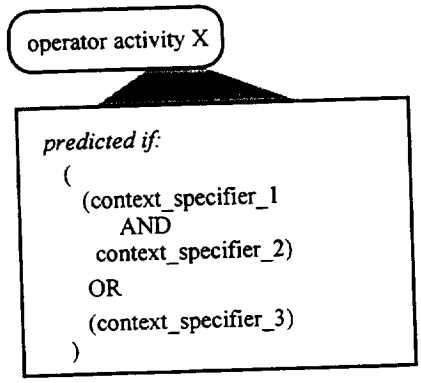

Fig. 1. Generic depiction of a logical equation of "context specifiers" as conditions for predicting an activity in a CATS model.

2. CONTEXT REPRESENTATION IN HUMAN-MACHINE SYSTEMS MODELS

Before describing the theory of context, the paper first provides examples of context representation in human-machine systems models. All human-machine systems models represent context; indeed, it is context that makes modeling human-machine interactions tractable (Mitchell, 1996). How context is represented, however, varies with the form, unit of analysis, level of abstraction, and intended application of a particular model. Context is typically represented at a level of abstraction congruent with the model hierarchy and intended application. Moreover, modelers commonly make inexplicit assumptions when specifying context.

First consider CATS models, which use operator activities as the unit of analysis. Table 1 lists examples of context specifiers defined to represent context in

Table 1. Examples of context specifiers that have been used as conditions in CATS models. Italics denote context specifiers expressed at an elemental level; others are expressed conceptually.

Name	Description	Name	Description
aircraft_heading_within_limits	Heading within +/- 0.5 degrees of cleared heading issued by ATC	*afs_engaged_roll_mode_LNAV*	Engaged autopilot roll mode is LNAV
mcp_altitude_outside_limits	MCP altitude does not match cleared altitude from ATC	aircraft_speed_outside_XR_limit	Aircraft cannot attain the speed required to meet the next waypoint crossing restriction
fms_target_speed_outside_limits	FMS-computed target speed more than 10 knots from speed issued by ATC or published procedure	cdu_descent_speed_built	A descent speed (*e.g.*, 320/) that matches the ATC-cleared descent speed has been "built" in the CDU scratchpad
fms_target_altitude_within_limits	FMS-computed target speed within +/- 100 feet of altitude directed by ATC or published procedure	cdu_descent_mach_entered	A descent mach (*e.g.*, .82/) built in the CDU scratchpad that matches the ATC-cleared descent mach has been selected to the appropriate line on the correct CDU page
afs_engaged_pitch_mode_FL_CH	Engaged autopilot pitch mode is Flight Level Change	cdu_execute_complete	FMS has computed a valid route after a short time delay, and the previous modified route is now the active route

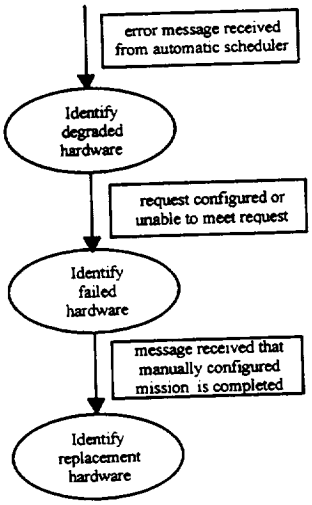

Fig. 2. Portion of an Operator Function Model for satellite ground controllers showing conceptual context information (Rubin, *et al.*, 1998).

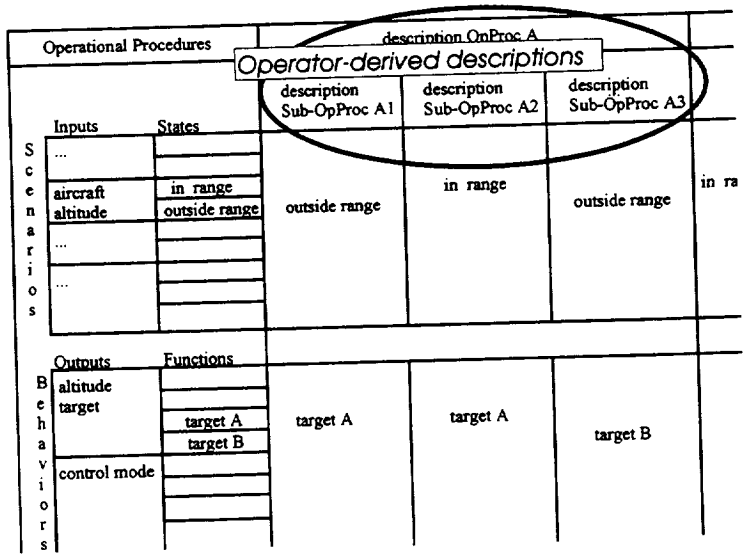

Fig. 3. Portion of an Operational Procedure (OpProc) table for flight management system design and training depicting some typical contextual information (e.g., aircraft altitude in range) and descriptions used by operators (Sherry, 1995; Feary et al., 1997).

CATS models (Callantine, 1996; Callantine, *et al.*, 1997, 1998), illustrating how they attempt to capture the manner in which supervisory controllers express when interventions with automated systems are needed. Some are expressed in terms of elemental state variables, while others are conceptual—they use concepts (e.g., target speed) that can have different meanings to different people or under different circumstances. CATS models are intended to be *memoryless*; evaluating the conditions in the model using the current state information should yield the currently preferred set of operator activities. A memoryless model can therefore readily answer "what if...?" queries.

However, memoryless context representations can be difficult to specify. When representing cognitive or perceptual activities along with overt operator actions, which is important (e.g., Mitchell, 1998), *exogenous* state information on which to base context may not be readily available; for example, it may hinge on the 'state' of verbal discourse between humans. In other cases, the information may be *endogenous*, for example, a constraint on available cognitive resources. For these reasons models often express context partly in terms of events, or as a script of behaviors. In Operator Function Models (OFMs), for example, operator-relevant concepts accompany elemental context information (Fig. 2) (Rubin, *et al.*, 1988). The proposed theory allows 'context models' to be used to explicate assumptions and support queries to the model in these cases, an idea that is further described below.

Models that represent machine behavior are typically state-based; Mitchell (1996, 1998) describes how operator activity models originated from state-based models. Sherry's (1995) Operational Procedure model (Fig. 3), Leveson *et al.*'s (1997) SpecTRM-RL (Fig. 4), and the Statechart models developed by Degani and Heymann (1998) (Fig. 5) all use *states* as the unit

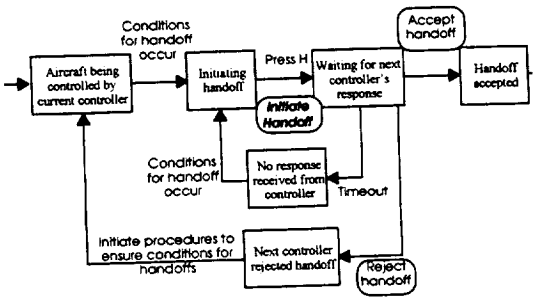

Fig 4. Portion of a SpecTRM-RL model of an air traffic controller procedure for "handing off" an aircraft to a different controller. Italics denote an activity performed by the 'handing off' controller (Leveson, *et al.*, 1997).

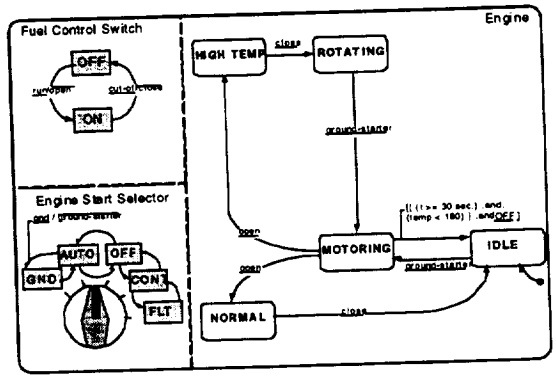

Fig. 5. Statechart model of a pilot procedure for "hot" engine starts uses both elemental and conceptual context information (Degani and Heymann, 1998).

of analysis, and include operator interventions as part of the contextual representation. State-based models are receptive to formal methods, making them advantageous for code generation (Sherry 1995), safety analyses (e.g., Leveson, *et al.*, 1997) and procedure design (e.g., Degani and Heymann, 1998).

Human-centered design efforts have included attempts to bridge the representational gap between state-based and event-based models. For example, Heimdahl *et al.* (1997) and Sherry (1995) both draw state-based models of machines toward event-based descriptions of operator activities, while Mitchell (1998) stresses the importance of representing system state in an OFM-based design methodology. Because research has attended more to extending models than relating them via context, some interesting crossover has occurred; for example, Fig. 5 depicts a state-based model developed to apply safety analyses to a new operator procedure. Because the procedure is part of a multi-agent system, this model also includes conceptual information about the actions of *another* air traffic controller, as well as the controller performing the procedure. This example suggests that even when context is expressed at a level likely to match that of an activity model, context information still needs to be decomposed to determine how the models may relate.

3. A THEORY OF CONTEXT

This section describes how context representations described above are theoretically related. *Context* is the relationship between the state of a system element—some partition of the system—and the dynamic collection of constraints imposed on it by other elements. A *constraint* is a set of bounds on a *state*—a property that describes some aspect of the system element's overall condition at a given time. Context may be represented by relating five classes of information: state variables, limit states, concepts, values, and modifiers. A *limit state* is a *state variable* that is part of a constraint. In aviation, for example, air traffic controllers can impose a constraint on aircraft heading; "cleared_heading" denotes a limit state based on "aircraft_heading," which is a state variable that represents an aspect of the aircraft's overall state (i.e., its heading).

Concepts are contextual information that is expressed at a level of abstraction higher than the elemental level at which a state variable represents an invariant with respect to the world. For example, in a context specifier such as "high_on_the_path," "path" is a concept used to represent a high-level constraint generated by an aircraft's Flight Management System. By contrast, "aircraft_ heading_within_limits" is a context specifier expressed in terms of an elemental state variable (i.e., "aircraft_heading") whose *meaning* is invariant—while there are different ways of expressing heading, they can be reliably converted. Concepts may also represent things that are temporally removed from the present. For example, predicted future states or *events* used to describe the attainment of some state are concepts. The remaining class of context infromation, *values*, may be Boolean, numeric, fuzzy, etc., as long as they have the same units as other context information they appear in a relation with.

The theory provides for the representation of concepts in one of two ways. First, they may be decomposed via logical equations of context specifiers representing lower-level concepts until the level of elemental state information is reached. Alternatively, a concept can be evaluated through the use of a "context model"—a model that takes state variables and limit states as input and produces output that matches the units of the relationship in which the concept appears. The model output might be a Boolean value, a predicted future state value, or a fuzzy-valued determination about progress toward meeting some constraint. Thus, through the notion of concepts, the theory affords both flexibility and the capability to represent context in terms salient to various designers and practitioners.

4. REPRESENTING CONTEXT USING THE CATS MODELER

A model's utility depends increasingly on a supporting framework that includes an application methodology and computer-based tools for developing and visualizing the model, so that it can be easily modified and understood by people with different areas of expertise. A model of a new procedure for example, may change frequently in response to decisions made by automation designers, and vice versa. The process of negotiating such changes requires conventions for communicating about them and thus benefits from a way to visualize the relevant models (e.g., Mitchell, 1998; Leveson, *et al.*, 1997). A prototype tool, called the CATS Modeler, is under development for the purpose of specifying and visualizing CATS models. The CATS Modeler is implemented in Java™, with an eye toward permitting collaborative model development via the World Wide Web.

CATS models are represented in computer-readable files that afford easy editing of the activity hierarchy and conditions (Callantine, 1996). In addition to making the specification process graphical, the CATS Modeler is designed to allow the CATS context specifiers to be defined graphically. Because CATS is designed to take system state information and constraints on operation as inputs for predicting operator activities according to the conditions in a model of operator activities, defining the data required and mechanism for evaluating a given context specifier at runtime is especially important.

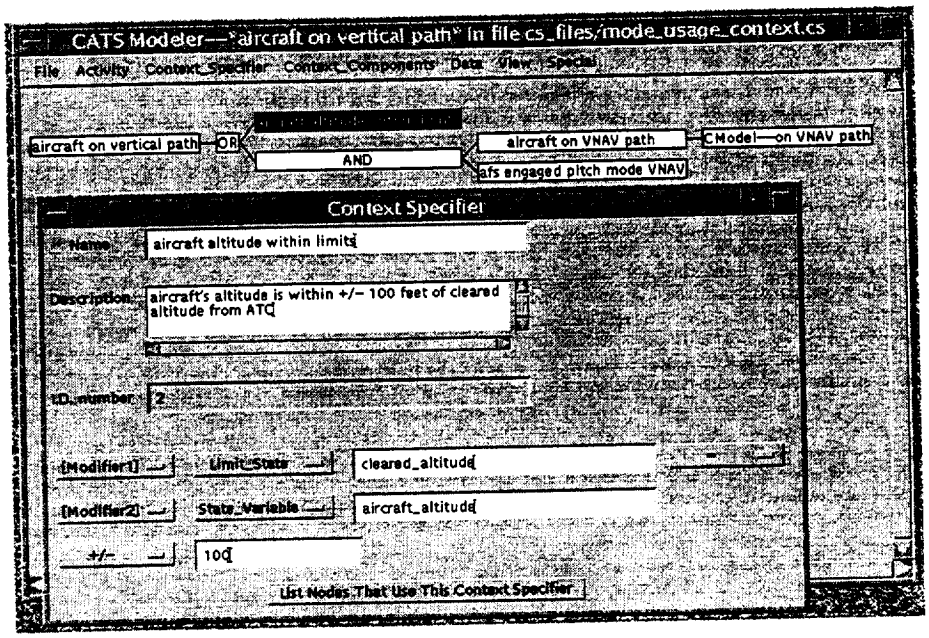

Fig. 6. Screen snapshot from the prototype CATS Modeler, showing the dialog used to define an elemental context specifier ("aircraft altitude within limits"), and a logical equation of context specifiers used to define a context specifier that is a *concept* ("aircraft on vertical path").

By implementing the proposed theory, the CATS Modeler offers a great deal of flexibility for specifying contextual information. Elemental context specifiers are defined using a simple dialog; context specifiers that are concepts can be defined as logical equations of other context specifiers. Fig. 6 shows an elemental context specifier ("aircraft altitude within limits"), and a context specifier that is a *concept* being defined for use in a CATS model. The AND/OR tree used to express the logical equation supports additions, deletions, and adjustments via graphical manipulation, while the elemental context specifier is defined by relating a state variable and limit state using a dialog to create a definition "sentence." Other choices offered in the dialog are enumerated in Fig. 7. One result of the context specification process is knowing that CATS must have access to a limit state called "cleared_altitude" and a state variable called "aircraft_altitude" in order to evaluate conditions, and concept-level context specifiers, that include the context specifier "aircraft altitude within limits." The second is that context specifier definitions created through the graphical specification process are explicit, hierarchical representations that eliminate the need to write code to evaluate context specifiers that appear in a CATS model—all of the knowledge that a CATS model represents can be specified and visualized graphically.

The 'context models' allowed for by the theory of context implemented in the CATS Modeler are exemplified by the "CModel—on VNAV path" node that appears in Fig 6. 'Context models' are a way of expressing, first, that it is too tedious to express the concept in question using logical equations of lower-level context specifiers—and that the model *is* making some assumptions here that have yet to be explicated. More importantly, however, they may serve as links through which different types of models can be integrated for various purposes. In this case, for example, a model certainly exists for how the Vertical Navigation (VNAV) mode of an aircraft 'knows' whether it is on the computed vertical path; the model could be 'attached' here to accurately provide this context information. Simulators or other sources are often not as accurate, or the information is distributed, making it difficult to be certain of its validity at any given time. In addition, context may depend on predicted future states that require a 'context model' for evaluation. The context specifier "aircraft_speed_outside_XR_ limit" in Table 1 exemplifies a concept that requires a predictive model.

Similarly, models attempting to predict what an operator will *actually* do in a given situation might use a 'context model' based on a theoretical model of cognition or perception to explicate the role of a particular decision making process or interface in determining context. Whether or not the models employed for this purpose are accurate, they nonetheless explicate assumptions about context that impact the utility of a human-machine systems model.

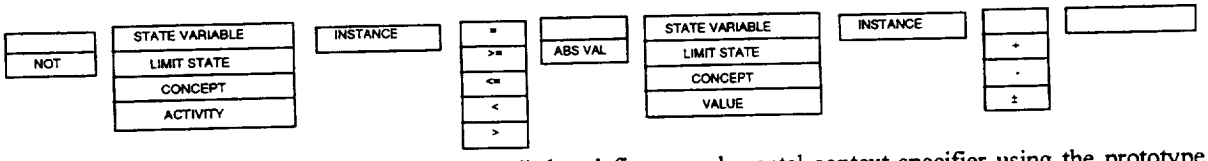

Fig. 7. Choices offered for creating a "sentence" that defines an elemental context specifier using the prototype CATS Modeler.

One additional noteworthy item is the appearance of the choice "ACTIVITY" in Fig. 7. Because CATS assigns 'statuses' to activities in a model as part of its processing methodology (Callantine, 1996), the CATS Modeler offers the opportunity to 'chain' contextual information as a modeling convenience, similar to scripting activities in other models. If the conditions under which an activity attains some status is well-defined, using the fact that the activity *has* that status at the current time is nearly as good as explicating the context. However, interactions between the predictive and interpretive portions of the CATS processing scheme (Callantine *et al.*, 1998) must be taken into account when exercising this option.

5. CONCLUDING REMARKS

The notion of 'context as constraints' is very powerful; in fact, any human-machine systems research or design endeavor can be cast in terms of context. For example, any interaction in a human-machine system can be viewed as conveying or describing progress toward some constraint. Designing complex systems essentially entails determining how to generate, communicate, assess, amend, and achieve dynamic constraints. Notions such as "cognitive complexity" and "mode awareness" describe situations when operators are tasked with converting constraints or assessing them in terms relevant to the machine, instead of the goals they are trying to achieve.

This paper proposed a theory of context as a means of relating human-machine systems models in one of two ways: first, decomposing context information into its elemental form, using state descriptors with invariant meaning or, second, using one model to support context representation in another. Application of the theory could make models more representative of the system element(s) they are abstractions of, easier to specify, and better suited for a particular application. The perspective taken establishes a starting point for integrated design, viz., the constraints that are the medium through which the system elements relate.

6. ACKNOWLEDGEMENTS

This work benefited from discussions with a number of insightful individuals, including Asaf Degani, Christine Mitchell, Everett Palmer, Nancy Leveson, Thomas Prevot, Micheal Shafto, Lance Sherry, and Charles Hynes.

7. REFERENCES

Callantine, T. J. (1996). Tracking operator activities in complex systems. Unpublished Doctoral Dissertation, Atlanta, GA: Georgia Institute of Technology.

Callantine, T. J., Palmer, E. A., and Smith, N. (1997). Model-based crew activity tracking for precision descent procedure refinement. *Proceedings of the 1997 IEEE Conference on Systems, Man, and Cybernetics*, Orlando, FL.

Callantine, T. J., Mitchell, C. M, and Palmer, E. A. (1998). GT-CATS: Tracking pilot mode usage activities in the glass cockpit. *International Journal of Aviation Psychology*, submitted for publication.

Degani, A. and Heymann, M. (1998). Formal aspects of procedures: The problem of sequential correctness, submitted for publication.

Feary, M., Alkin, M., Palmer, P., Sherry, L., McCrobie, D., & Polson, P. (1997). Behavior-based vs. system-based training and displays for automated vertical guidance. In *Proceedings of the Ninth International Symposium on Aviation Psychology*. Columbus, OH.

Funk, K. H., and Lind, J. H. (1992). Agent-based pilot-vehicle interfaces. *IEEE Transactions on Systems, Man, and Cybernetics, 22*(6), 1309-1322.

Heimdahl, M., Leveson, N. G., Reese, J. (1997). Intent specifications: An approach to building human-centered specifications. Everett, WA: Safeware Engineering Corporation.

Leveson, N. G., Sandys, S., Pinnel, D., Brown, M., Joslyn, S., Alfaro, L., Zabinsky, Z., and Shaw, A. (1997). Final report: Safety analysis of air traffic control upgrades. NASA Contractor Report: Ames Research Center.

Mitchell, C. M. (1996). Models for the design of human interaction with complex systems. In *Proceedings of the Conference on Cognitive Engineering Systems in Process Control*, Kyoto, Japan, November.

Mitchell, C. M. (1998). Model-based design of human interaction with complex systems. In W. B. Rouse and A. P. Sage (Eds.), *Systems Engineering & Management Handbook*. New York: Wiley (to appear).

Rubin, K.S., Jones P. M., and Mitchell, C. M. (1988). OFMspert: Inference of operator intentions in supervisory control using a blackboard architecture. *IEEE Transactions on Systems, Man, and Cybernetics, 18(4)*, 618-637.

Sherry, L. (1995). A formalism for the specification of operationally embedded reactive systems. In *Proceedings of the International Council of System Engineering*, St. Louis, MO.

Smith, N.and Moses, J. (1998). Personal Communication, NASA Ames Research Center.

Vakil, S. S. and Hansman, R. J. (1998). Functional models of flight automation systems to support design, certification, and operation. AIAA Report 98-1035. Reston, VA: American Institute of Aeronautics and Astronautics.