ESDA2012-82892

MODEL-BASED APPROACHES TO HUMAN-AUTOMATION SYSTEMS DESIGN

Greg A. Jamieson
University of Toronto
Toronto, Ontario, Canada

Jonas Andersson
Chalmers University of Technology
Göteborg, Sweden

Ann Bisantz
University at Buffalo
Buffalo, NY, USA

Asaf Degani
General Motors R&D
Hertzeliya, Tel Aviv, Israel

Morten Lind
Technical University of Denmark
Lyngby, Denmark

ABSTRACT

Human-automation interaction in complex systems is common, yet design for this interaction is often conducted without explicit consideration of the role of the human operator. Fortunately, there are a number of modeling frameworks proposed for supporting this design activity. However, the frameworks are often adapted from other purposes, usually applied to a limited range of problems, sometimes not fully described in the open literature, and rarely critically reviewed in a manner acceptable to proponents and critics alike. The present paper introduces a panel session wherein these proponents (and reportedly one or two critics) can engage one another on several agreed questions about such frameworks. The goal is to aid non-aligned practitioners in choosing between alternative frameworks for their human-automation interaction design challenges.

INTRODUCTION

An enduring and increasingly important challenge for systems designers lies in designing automation that human operators can employ effectively in controlling complex systems.

Although researchers and practitioners have advanced several frameworks to guide human-automation interaction design, none have been widely adopted for this purpose by cognitive (systems) engineers. Rather, there are pockets of practice, largely organized according to geography or academic progeny, with no apparent trend towards consensus or even agreement on the merits of candidate frameworks. Such consensus may or may not be beneficial: e.g., it may very well be that different frameworks have different strengths and weaknesses, or be useful for different, but complementary, analysis purposes.

Previous articles have offered comparisons of several such frameworks (e.g, Burns & Vicente, 2001). Others have offered critiques of selected frameworks (Lind, 2003; Cummings, 2006). These works tend to be received uncritically within the authors' pocket of practice - and ignored or refuted (e.g., Lintern, 2006) outside of it. However, deference to thought leaders in one's in-group, and indifference or antagonism to ideas from the out-group, will not carry us toward deeper understanding of how different modeling approaches may support a variety of aspects of system analysis and design.

This panel presentation will present several frameworks advanced for modeling human-automation interaction. To motivate and structure the panel discussion, and provide a framework both for readers and attendees, this paper identifies common dimensions along which the frameworks can be compared. Panelists have described five modeling approaches according to those dimensions. A comparison matrix is then presented which highlights the potential similarities and differences among the frameworks, and which will motivate discussion during the panel session.

We will consider this effort to be a success if each author can cite this article as a fair juxtaposition of the multiple frameworks. This may seem a low threshold for success, but it is nevertheless a step forward along what has proven to be a difficult path.

BASIS FOR COMPARISON

This paper provides a description of the models in terms of five initial questions that we expect will reveal important differences between the candidate frameworks. During the panel session, panelists may extend this list as we encounter one another's initial responses.

1. What is the origin of the modeling framework?
2. What is the object of modeling?
3. What relationships are captured in the model?
4. For what problem is the framework intended or best suited?
5. What are the known limitations/development opportunities for the framework?

In the following sections, each author presents a summary description of one framework (following the order of authorship above). We then present a tabular matrix to support comparison of the frameworks.

ABSTRACTION HIERARCHY

1. What is the origin of the modeling framework?

The Abstraction Hierarchy (AH) was initially conceived to provide context for verbal protocols recorded by technicians trouble-shooting electronic equipment (Rasmussen & Jensen, 1974). The AH, coupled with an orthogonal part-whole decomposition hierarchy, served to capture the variety of ways in which the technicians spoke about, and by implication thought about, the systems that they were troubleshooting. The abstraction dimensions captured the way that technicians transitioned from concrete descriptions of physical properties of the system to abstract descriptions of the system's purpose. The decomposition dimension captured the progression of granularity, from course to fine, across which technicians spoke about the system parts, assemblies, and the whole system.

Several insights from that origin are pertinent to human – automation interaction. First, it is likely that automation and automated processes can be meaningfully described in many ways along dimensions of abstraction and decomposition. A framework that introduces an organizing principle to these varied descriptions may well prove useful. Ultimately, the design for human-automation interaction should support workers thinking about automated systems in multiple ways. Second, although the ways in which individual workers in specific situations think about the automation will vary, the object of cognition – the automated system – is comprised of relatively constant engineered components and governed by either natural or scientific laws. It is reasonable to argue that a representation of meaningful ways of thinking about an automated system would be useful for human-automation interaction design.

2. What is modeled in the framework?

The AH models the physical and functional elements of a work domain. Importantly, the AH provides multiple descriptions of the same object, distinguished along several levels of abstraction.

3. What relationships are captured in the model?

The AH captures the goal-relevant structural *means-ends* relations between the objects, functions and purposes that comprise the work domain.

4. For what problem is the framework intended or best suited?

The AH has proven to be well suited to several uses. Generally speaking, it is useful as a knowledge representation framework for domains that are sufficient complexity that most humans will use coping mechanisms when reasoning about the domain. Abstraction is a means of coping with complexity and means-ends relations span levels of abstraction.

In systems engineering practice, the AH serves as a vehicle for cognitive (systems) engineers to capture, organize, and develop their understanding of the functional relationships between system elements. This knowledge representation can then be used in the conceptualization, exploration, and synthesis of decision support systems, human-system interfaces, etc.

5. What are the known limitations/development opportunities for the framework?

Use of the AH in the representation of automated systems and the design for human-automation interaction is limited by an unfortunate precedent. Vicente (1999) explicitly excludes automation from work domain descriptions by defining a work domain as "the system being controlled, independent of any particular Worker, Automation, Event, Task, Goal, or Interface" (p. 10). In other work, however, the AH is said to "represent the set of goal-relevant constraints governing the operation of the controlled system." (Bisantz & Vicente, 1994, p. 84). If the analyst adopts the perspective that the automation is controlled by the worker, then its goal-relevant constraints can be represented through an AH. This latter interpretation is adopted by many contemporary AH practitioners. For example, Burns et al. (2004) and Naikar, Hopcroft, and Moylan (2005) argue that, in cases where workers must monitor, configure, manipulate or reason about automation, it should be represented in the AH.

INTEGRATED ABSTRACTION HIERARCHY AND PERCEPTION-ACTION CYCLE

1. What is the origin of the modeling framework?

The origin of the tentative model is an attempt to describe and explain a range of human-automation related problems (e.g. out-of the loop problems, miscalibrated trust, loss of skills) in a unified model (Andersson, 2010).

2. What is the object of modeling?

The object of modeling is human-automation interaction, e.g. an operator's interaction with a control system in a power plant. The model template is used to create a knowledge representation and a system representation in the same format. By assessing discrepancies between the available knowledge (residing either 'in the head' or 'in the world' (Norman, 2002)) and the knowledge needed in different situations, automation related problems could be identified.

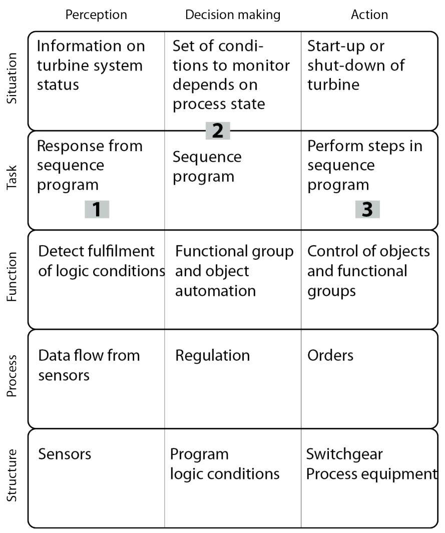
3. What relationships are captured in the model?

The model has its theoretical basis in cognitive systems engineering (CSE) (Woods & Hollnagel, 1983; Rasmussen et al., 1994; Hollnagel & Woods, 2005). The model consists of an assembly of two fundamental concepts in CSE, the abstraction hierarchy (Rasmussen, 1979), and the perception-action cycle (Neisser, 1976).

The abstraction hierarchy is used to describe the work domain. The Structure, Process and Function levels (Fig. 1) are recognized from Rasmussen's means-ends hierarchy (Rasmussen, 1994). These three levels provide a task-independent representation of the work system. An important difference compared to Rasmussen's hierarchy is the two top levels. The Task and Situation levels are included to bridge the gap between the work- and task domains to enable description of work in a single representation.

The perception-action cycle is a common denominator for human and technical cognitive systems. It provides a basis to describe both humans and automation in the same model with perception, decision-making and action as the common elements. The common basis for description facilitates mapping between the work domain, the operator's mental model and the means to support cognitive work. The assessment of mismatch between the operator model and the system model can then be used as a problem indicator.

Combined, the adapted abstraction hierarchy and the perception-action cycle can serve as a guide for analysis.


Perception
Decision making
Action
Situation
Information on turbine system status
Set of conditions to monitor depends on process state
Start-up or shut-down of turbine
2
Task
Response from sequence program
1
Sequence program
Perform steps in sequence program
3
Function
Detect fulfilment of logic conditions
Functional group and object automation
Control of objects and functional groups
Process
Data flow from sensors
Regulation
Orders
Structure
Sensors
Program logic conditions
Switchgear Process equipment
Figure 1. The tentative model template with a simplified description of an automatic turbine system

4. For what problem is the framework intended or best suited?

The aim of the present modeling approach is to develop a template that can be used in projects where a CSE approach is motivated but the available resources do not permit in-depth analysis. The intention is that the model template should support high-level analysis by functioning as a mediating object to spark discussion, facilitate new perspectives and create a common understanding of the problem scope in a multidisciplinary development/design team.

To illustrate the intended use a simplified example is given in Figure 1. The data in the example comes from an empirical study made in the nuclear power domain (Andersson, 2008). Seven turbine operators were interviewed regarding their use of an automatic turbine system (ATS) and the associated hard wired panel interface. The template was populated using interview results and technical documentation. The list below refers to the notations in Figure 1.

1. The ATS indicates the state of each program step. Information from the ATS is provided in aggregated form as sequence steps. The program logic that controls the sequence program are not accessible through the ATS user interface which complicates troubleshooting.
2. The ATS monitors different conditions depending on the process state. If a sequence is initiated without having all conditions fulfilled, the ATS will drop back to a stable state. The viewing of fulfilled conditions are however inconsistent to other indications in the control room which leads to uncertainty.
3. Operators can perform on three distinct levels of automation. The preferred step-wise semiautomatic mode provides time to think combined with the speed of the automation's execution of actions. However, it is difficult for the operators to know exactly which objects in the process that are affected by an action due to the poor interface transparency of lower system levels.

The simplified example shows how the model can be used to analyze human-automation interaction. The analysis shows that ATS behavior is obscured by the design and the functionality of the task-based ATS user interface. The technology available in the control room does not provide the means to move efficiently between different levels of detail and abstraction. In this example, it leads to perceived out-of-the-loop problems which increase the time needed to assess, locate and deal with problems in case of an ATS failure.

5. What are the known limitations/development opportunities for the framework?

While intended for high-level modeling, the tentative model deliberately trades depth in analysis for a rapid system overview. This is a limitation since it comes at the cost of representing details in the mapping between work domain and operator mental model. The information provided in the model template can however be adapted to the scope of the analysis by iteratively arriving at a sufficient level of detail.

The model is new and has so far only been tested in small scale. Further testing is needed to determine its strengths and weaknesses. The ambition for future development is to make it into an analysis technique that fits within industrial business project constraints in an iterative product development process.

LENS MODEL

1. What is the origin of the framework?

The Lens model was developed by Egon Brunswik (Brunswik, 1955) in his psychological theory of probabilistic functionalism. Essentially, Brunswik described a process by which organisms (people) are able to function successfully in their environment by relying on available information (proximal cues) which provide information about relevant environmental aspects of variables (distal criteria). The proximal information does not uniquely specify the environmental state, and organisms are able to act "vicariously" by relying more or less on different cues, depending on the circumstance (Goldstein, 2006; Hammond & Stewart, 2001). While Brunswik's original Lens Model served as an explication of his theoretical concepts, mathematical formulations of the model (based initially on multiple regression) were developed (Hammond, Stewart, Brehmer, & Steinmann, 1975; Cooksey,

1996) and have been widely applied in the analysis of human judgment (Cooksey, 1996). A key insight of the Lens Model, which is expressed formally in the mathematical formulation, is that judgment success is a function not only of an organism's ability to use environmental information correctly and consistently, but on the degree to which the environment is predictable given the available cues – that is, the structure of environment itself is a limiting factor in performance. More recently, the Lens Model has been adapted for use in cognitive engineering oriented human-system modeling, including studies of dynamic decision making (Bisantz et al., 2000), reliance on automated aids (Bass & Pritchett, 2008; Seong & Bisantz, 2002, 2008; Seong, Bisantz, & Gattie, 2006), display interventions (Strauss & Kirlik, 2006) and team performance (Adelman, Yeo, & Miller, 2006).

2. What is modeled in the framework?

As typically applied in human judgment research, the Lens Model models the relationships between a human judgment, the unknown environmental criterion (or variable) to be judged, and the informational cues available on which the judgment can be based. The relationships between the cues and either the criterion, or the judgment, are assumed to be probabilistic. Quantitative model parameters used to assess judgment performance are developed empirically, by modeling multiple instances of judgments, criteria, and cue values using multiple regression techniques (although other mathematical frameworks have been utilized - see Jha & Bisantz, 2006; Rothrock & Kirlik, 2003). These parameters include measurements of judgment quality, "correct" use of cues, consistency, and the degree to which the environment is predictable given the cue set.

Although the Lens model was initially formulated to examine a single organism interacting with the environment, extensions of the model exist which can be used to compare multiple judgment systems, including judgments made by decision aids or automated agents (Bass & Pritchett, 2008; Seong & Bisantz, 2002, 2008; Seong, et al., 2006). Here, the relationships between operator judgments and those provided by one or more automated systems can be measured, including the extent to which automated algorithms have similar policies and outputs as the human operator. Such findings can be applied to the design of automated systems which allow better calibration of trust, an important intervening variable in decisions on the part of the human agent to utilize outputs from automation (Bisantz & Pritchett, 2003; Seong & Bisantz, 2008).

3. What relationships are captured in the model?

As noted above, the Lens model *is* a model of relationships – it explicitly captures and quantifies relationships among distal criteria, proximal cues, and judgments. In the most typical formulation, the relationships are expressed as correlations and provide parameters that measure various components of judgment. The parameters are based on two symmetric linear regression models: one capturing the relationship between sets of cues and actual environmental criteria, and the other capturing the relationship between the same cue sets and the judgments that are made based on those cues In particular, the following parameters can be computed:

Achievement: the correlation between criteria and judgments.

Environmental predictability: the correlation between criteria values associated with a set of cues, and criteria values predicted by a linear regression model of the cues and criteria (essentially, the degree to which a linear model can predict the criteria values).

Consistency: the correlation between actual judgments and judgments predicted by a linear model. This is a parallel measure to environmental predictability, and measures the degree of fit a linear model of cue values and judgments.

Linear Knowledge: the correlation between predicted judgments, and predicted criteria. This parameter measures the degree to which the linear model of the judge matches the linear model of the environment.

Unmodeled Knowledge: the correlation between the residuals of both models. This parameter captures model similarities not captured by the linear regression models.

The Lens Model equation (Hammond et al., 1975) shows that achievement is a function of linear and unmodeled knowledge, and consistency (both of which measure performance of the judge) as well as environmental predictability (a function of the judgment environment).

4. For what problems is the framework best suited?

This framework is appropriate for modeling joint human-automation judgments, a common paradigm in human-automation interaction in which people make judgments in parallel with an automated system, and then to consider, accept or reject the automated output as appropriate. Often, displays are used to help operators understand the context under which the judgment is being made and to convey the applicability or limitations of automated outputs given that context. Successful design of such systems requires understanding the attributes of judgment that they are to support, and the impact of their design on human judgment. Brunswik's Lens model can support such an understanding.

For example, we have used the approach to understand similarities between pilot judgments of potential collisions with candidate alerting algorithms (Bisantz & Pritchett, 2003). Pilot judgment characteristics were least similar to the most sophisticated (and therefore preferred) algorithm, indicating the need for displays which could convey the automation's reasoning and support pilot trust in the alerting system. Similarly, the Lens Model has been used to measure and provide feedback to operators regarding the performance of automated judgment aids, improving operator trust in and reliance on these aids (Seong, et al., 2006).

5. What are the limitations of the modeling framework?

Challenges associated with using the lens model are similar to any modeling technique: adequately identifying and mapping aspects of the world to model constructs, and insuring the computational needs of the framework are satisfied. To construct a lens model, one must identify – and assign values to – criteria, cues, and judgments. To determine values of the lens model parameters, one must have repeated instances of judgments on which to base the regression models. While it is possible to construct such circumstances in a laboratory setting (e.g., repeatedly ask participants to make judgments in response to sets of multiple cue values, where the cues are indicative of some known criterion value) deriving these values in a real world setting, or even in more realistic experimental settings, is challenging. For example, on a complex air warfare situation display, what cues are available to make judgments about the identity of unknown aircraft? How can the values of those cues be known and recorded at the time of judgment? What happens if these cues change over time? In a simulator-based study, it might be possible to log or record information, but this may be more difficult if data are captured in a field setting. One must also make assumptions about cue values being used at the time of judgment, particularly if those values are changing. Similarly, in a real-world setting, the truth value of the criterion might not always be known, or even knowable. Also challenging are circumstances where cues or judgments are not continuously valued, or where the cue-criterion or cue-judgment relationships are not adequately modeled using multiple regression techniques (i.e., where there are substantive non-linear relationships). Researchers have explored various strategies to address these issues in order to apply the Lens Model successfully to more complex, dynamic judgment situations (Bisantz, et al., 2000; Bisantz & Pritchett, 2003; Jha & Bisantz, 2006; Rothrock & Kirlik, 2003).

FINITE STATE MACHINES

1. What is the origin of the modeling framework?

The focus of the modeling framework is on a behavioral description of the system (i.e., its states, modes, and transitions), with special emphasis on user interaction and display feedback. The origin of the framework is Finite State Machine theory (Turing, 1936). Parnas (1969) was probably the first to use Finite State Machine models to describe user interactions with a computer terminal. This formalism enabled him to pinpoint several design errors such as “almost-alike” states, inconsistent ways to reach a state, and data entry problems. Foley and Wallace (1974) also used this notation to describe their concept of a language of interaction between human and computer. Their state transition diagram describes the actions needed to make the transition from one state to another, as well as the system's response during the transition. Jacob (1983) described several variants of the Finite State Machine, as well as other formalisms such as the Backus-Naur Form (BNF) (Woods, 1970) in the design of specifications for human-computer interaction with a communication system. He showed how such formalisms can be applied to a system with many commands. Since then, many researchers have used Finite State Machine theory to model user interactions (Kieras and Polson, 1985; Wasserman, 1985) as well as a variety of newer extensions such as Petri Nets (Palanque & Bastide, 1996), Discrete Control Networks (Mitchell and Miller, 1986), OFM (Mitchell, 1987), and Statecharts (Harel, 1987).

2. What is modeled in the framework?

A Finite State Machine model is a way of describing a system with its finite possible configurations— where the machine can be in any one of a finite number of configurations, or “states.” The model has only finite input and output sets:it can respond only to the specified set of stimuli and produce only the specified set of behaviors (Turing, 1936, p. 232-235). Finite State Machine theory captures the behavioral aspects of a system in a very precise and complete way; i.e., how it works and, in the context of human-machine interaction, how the system responds to user inputs and what information and feedback it provides to the user. The general structure of such a machine is described by state transitions of the following form: when event *alpha* occurs in state A, the system shifts to state B. The model can also be represented graphically as a state transition diagram that presents this behavioral information (states and transitions) as a perceptual code of nodes and arcs. This combination of theoretical and graphic formats has been used to represent human interaction with computer-based systems (Degani, Heymann, & Gellatly, 2011).

3. For what problems is the framework best suited?

The idea is to describe the machine’s behavior and all possible user interaction and display information as a way to help designers understand, evaluate, and formally specify the system. Another objective is to identify situations where the interface is incorrect (Degani & Heymann, 2002), as well as situations where there are opportunities to improve (e.g., simplify) the interface (Heymann & Degani, 2007). There are a number of benefits from using a state-machine based formalism for this type of design approach: (1) it constitutes a clear description of the (interaction) design that enables review and discussion among multidisciplinary teams, (2) it articulates overarching design requirements well as generic design patterns, (3) it uses a formal description for specifications, (4) it establishes a platform for analysis, heuristic or otherwise, of the design, and (5) it informs and supports the design of the graphical user interface (e.g., screen layout). Thus, the overall intent is to provide a formal approach to the design of human-machine interactions to improve not just the design but also the quality and rigor of the specifications. Quality means that the description is detailed and leaves nothing to interpretation or possible ambiguity. Rigor means that all system events and transitions are accounted for and described in the specifications (Leveson, 1995).

4. What relationships are captured in the model?

From a human factors and user interaction perspectives, the modeling framework is best suited to capture four types of relationships:

(1). The models describe and illustrate how external events trigger changes and reconfigurations in the system. For example, in an aircraft autopilot the angle of attack is a critical parameter that is constantly monitored. When the aircraft's angle of attack exceeds a given value, the aircraft may be entering stall. When this value is reached (around 15 degrees in most commercial airliners), the envelope protection system "kicks in" and will automatically advanced the throttles and/or reduce pitch attitude. In modeling autopilots systems, we pay outmost attention to such external events and their impact on the system (see Degani, 1994, Ch. 15). Along the same lines, we also describe important internal events such as timeouts (that may shut the system down), and outputs of internal computations (that may trigger a mode change). Naturally, we also focus on user initiated events such as global events (on/off switching), mode changes, reference values changes (e.g., aircraft altitude), and any other events that changes the state of the system and/or its interface.

(2). We carefully consider relations between the different components of the system under consideration. For example, in Figure 2, consider the behavior of the AIR-SOURCE component. The model shows that there will be (an automatic – note the magenta broken line) state change from "recycled-air" to "fresh-air" when the driver switches the AIR DELIVERY mode to "defrost." Namely, a side effect here on the AIR-SOURCE due to changes that take place in another component (AIR DELIVERY). Generally speaking, we look for things like side effects, guards, and conditionals that are either within a given component or, in particular, those that affect other components in the system. Research on human-automation interaction had consistently shown that such coupling tends to confuse users (Degani, 1994; Sarter &Woods, 1994; 1995) and are usually easily forgotten (Crow, Javaux, & Rushby, 2000).

(3). The model helps the analyst to consider and evaluate the potential impact of external and internal events as well as side effect and guards on the user. (This, by the way, is the objective of the work and model described in Figure 2 – see Degani, Heymann, & Gellatly for the full analysis). That is, we analyze the relations between system events and user understanding and performance. We can define different severity categories of such events on the user interaction and evaluate their potential impact on ease of use, elegance of interaction, and safety. We ask question like "will it cause confusion?" "Can and will users eventually understand system behavior?" And how much support is provided in the interface and in the user manual?

(4). Finally, we use to model to consider relations between machine model (that describes the behavior of the machine) and the interface model (a reduced and modified projection of the user model). Are all the important events in the machine model are projected to the user interface? Are there situations where the interface becomes non deterministic (error-states) or the interfaces blocks or (e.g., unnecessarily) augments the machine model (Degani & Heymann, 2007)? In addition to the machine modes and interface model, it is possible to add also the user model where the user's "mental" model of the interface and machine is described (Romera, 2000). Here we can account for situations where users forget (over time) how the machine works or simplifications and heuristics that people discover and employ (and evaluate their correctness).

5. What are the limitations of the modeling framework?

First and foremost, it focuses on the system or machine under consideration. Therefore it does not address many human factors issues such as the cognition, decision making, perception, physical limitations. In terms of the modeling of the machine and user interaction, this modeling framework requires a thorough engineering understanding of the system as well as technical savvy in system analysis and verification techniques. The analysis of the model is only as good as the properties that are at the disposal of the analyst. As it stands now, we have only a limited set of properties to verify in a given system. Last but not least, like most modeling frameworks it is negatively affected by the level of abstraction used to describe the system. Naturally if the level of abstraction is high (overly simplified description) the results may be incomplete.

MULTILEVEL FLOW MODELING

1. What is the origin of the modeling framework?

Multilevel Flow Modeling (MFM) has one of its roots in the abstraction hierarchy (AH) of cognitive system engineering (CSE) (Rasmussen & Lind, 1981)(Rasmussen & Lind, 1982)(Rasmussen, 1983). Initially the aim of MFM was to provide a formalization of the AH. But current MFM shares only the basic end-means and part-whole concepts with the AH; the concepts of hierarchy and levels of abstraction have been abandoned as leading concepts. The constitution of levels of end-means abstraction is addressed explicitly through domain dependent modeling schemata.

Other important influences are coming from Artificial Intelligence research, especially knowledge representation and qualitative reasoning. MFM is also influenced by cognitive semantics and current developments of MFM seek to provide a logic foundation for MFM in concepts of action. The aims of these developments are to address completeness, consistency and validation issues. MFM concepts and applications are still under development. An introduction to MFM and recent extensions are presented in (Lind, 2010), (Lind, 2011a) and (Lind, 2011b).

2. What is the object of modeling?

Current MFM models provide representations of the goals and functions of a physical process (energy, chemical etc.) and its controls. MFM makes a distinction between a system seen as an object of design and seen as an object for the actions of an operator. MFM can be used for both purposes.

In system design MFM can be used in the conceptual phase as a means for defining the relations between the goals and function of the plant process and the controls, and for control design. The allocation of control functions between computers or operators can be represented. But MFM does not include the information to make the allocation decision. Such decisions must be based on performance evaluations, which require models of the behavior of the automated systems and the operator.

In system operation an MFM model can be used as a basis for defining and organizing the information to be presented for the operator (i.e. what to present). MFM makes it possible to specify qualitative knowledge of the process and its control which is important for plant operation but which is implicit in current approaches to display design (Lind, 2009)(Lind, 210b). The symbolic language of MFM is not necessarily efficient for operator interaction and designing the presentation of MFM information and the structure of the operator dialogue with the computer is seen as a separate issue requiring other model types.

MFM is also a basis for the development of knowledge bases for intelligent decision support in diagnosis and counteraction planning and risk management (Petersen, 2001) (Lind, 2011c)(Gofuku, 1999).

3. What relationships are captured in the model?

An MFM model represents means-end and part-whole relations of the process and its control. MFM models are composed of multiple levels of abstraction often combining the principles of end-means and whole-part decomposition. MFM differentiates between different several goal types and separates functions into flow functions and control functions. Flow functions representing the purpose of physical plant equipment and subsystems are combined into flow structures. Control functions represent the purpose of control agents (system or operator) and can be combined in control structures.

MFM distinguish between several types of end-means relations. These relations are defined by underlying conceptual schemata including temporal aspects and semantic roles. These foundational schemata play a significant role as templates in the MFM modeling process.

MFM models provide a clear representation of the relative roles of process and control functions in achieving plant goals and show clearly that control functions serve a particular role in the constitution of functional levels.

MFM models represent systems as hierarchical or heterarchical and sometimes having cyclic structures depending on the situation.

4. For what problem is the framework intended or best suited?

The aim of MFM is to support both the design of the process and its controls and for design of decision support systems for diagnosis and planning in supervisory control.

The modeling framework is especially suited for problems where a high degree of formalization is required. In system design MFM can help in formalizing qualitative process and control knowledge which cannot be properly be specified with existing modeling techniques used in process and control design. MFM supports reasoning and can therefore be used to build knowledge bases for decision support.

MFM is used to model industrial systems involving the processing and control of energy and materials. A large number of models have developed including fossil and nuclear power generation plants (Gola, 2011)(Lind et.al, 2011) and chemical engineering plants (Gofuku et.al., 1999). These models have been used for risk analysis, fault diagnosis and for counteraction planning.

5. What are the known limitation/development opportunities for the framework?

The limitation of MFM to industrial processes is seen as a necessary and unavoidable consequence of the level of formalization achieved by the modeling framework. End-means and part whole concepts are widely applicable but this is not the case for the specific interpretation these concepts have in MFM. This limitation is actually a requirement for the kind of applications MFM is aiming at.

However, the high level of formalization is also an obstacle for acquiring the knowledge required to build MFM models. We are therefore considering how the knowledge acquisition process can be facilitated through a less formal application of end-means and part-whole concepts (see e.g. the case study presented in (Lind et.al, 2011)).

Applications of MFM are highly dependent on proper computer based tools for modeling and reasoning. Several tools are under development in Denmark, Norway, Japan and China. One of the aims of these developments is to provide tools to handle the challenges in building and validating MFM models of systems with realistic complexity.

There is need for development of problem solving models based on end-means and whole-part reasoning which can be used together with MFM as templates for human machine interaction patterns in supervisory control. These interaction patterns could be used together with "presentation" models that specify the presentation at the interface.

COMPARISON MATRIX

Having allowed each proponent to respond to agreed questions about their chosen framework, we now reduce those descriptions to a matrix to aid in comparing them.

Framework	Origin	Object of Modeling	Relations captured	Intended/Best use	Limitations
Abstraction Hierarchy	Empirical observation	Physical and functional elements of work domain under control	goal-relevant structural means-ends relations	Knowledge representation in complex domains	Limited application to representing automated systems
Integrated AH and perception-action cycle	Specified based on need	Human-automation interaction	1. means-ends relations 2. perception, decision-making and action	Analysis of human-automation interaction	1. trades depth in analysis for a rapid system overview 2. Explored only at small scale
Lens Model	Psychological Theory	Human judgments of unknown environmental states, based on a set of informational cues which are typically correlated, and which have probablisitic relationships to the unknown environments state.	Relationships between the judgment, the unknown environmental criterion to be judged, and the informational cues available on which the judgment can be based	Modeling joint human-automation judgments	Deriving criteria, cues, and judgments in real systems.
Finite State Machines	Computer science theory	Configurations of the controlled system		1.Describe machine behavior and all possible user interaction and display information 2.Identify situations where the interface is incorrect or can be improved.	Does not address limitations of the user
Multi-level Flow Model	CSE, AI and Cognitive Semantics	Goals and function of the process and its controls.	Means-end and part whole relations.	Integrated Process, automation and HMI design including intelligent decision support.	Knowledge acquisition bottlenecks.

A cursory review of the comparison matrix reveals remarkably little overlap between the frameworks. There are at least two explanations for this. First, it is possible that, whereas the proponents would all claim that their preferred framework is useful for designing for human-automation interaction, the nature or goals of the intended design activity may be quite different. Second, with the exception of the Integrated AH and Perception-Action cycle, none of the frameworks were explicitly intended for human-automation interaction. Rather, the balance of the frameworks were conceived for different purposes and have been adapted to the human-automation interaction problem. This may be attributable to the tendency of engineers to see design as an extension of analysis. Therefore, it is natural to attempt to extend analytical frameworks to design.

CONCLUSIONS

We hesitate to draw strong conclusions in advance of the panel session itself. However, one could argue that the variety of entries in the comparison matrix support the proposition that developers of human-automation interaction analysis and design frameworks would do a great service to practitioners if they would provide direct comparisons to alternative frameworks. This would allow designers to select from candidate frameworks witch a greater chance of finding a suitable match for their design problem.

REFERENCES

Adelman, L., Yeo, C., & Miller, S. L. (2006). Understanding the effects of computer displays and time pressure on the performance of distributed teams. In A. Kirlik (Ed.), *Adaptive perspectives on human-technology interaction: Methods and models for cognitive engineering and human-computer interaction* (pp. 43-54). New York: Oxford University Press.

Andersson, J. (2008). *Levels of Automation and User Control*, Project Report No. NKS-179. Roskilde: Nordic nuclear safety research.

Andersson, J. (2010) *A Conceptual Model for Analysis of Automation Usability Problems in Control Room Settings*. Licentiate Thesis, Department of Product and Production

Development, Chalmers University of Technology, Gothenburg, Sweden

Bass, E. J., & Pritchett, A. (2008). Human-automation judge learning: A methodology for examining human interaction with information analysis automation. *IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans*, 38(4), 759-776.

Bisantz, A. M., Kirlik, A., Gay, P., Phipps, D., Walker, N., & Fisk, A. D. (2000). Modeling and analysis of a dynamic judgment task using a lens model approach. *IEEE Transactions on Systems, Man, and Cybernetics: Part A: Systems and Humans*, 30(6), 1 - 12.

Bisantz, A. M., & Pritchett, A. (2003). Measuring judgment in complex, dynamic environments: A Lens Model analysis of collision detection behavior. *Human Factors*, 45(2), 266-280.

Bisantz, A. M. & Vicente, K. J. (1994). Making the abstraction hierarchy concrete. *Int. J. Human-Computer Studies*, 40, 83-117.

Brunswik, E. (1955). Representative design and probabilistic theory in a functional psychology. *Psychological Review*, 62, 193 - 217.

Burns, C. M., Bisantz, A. M., & Roth, E. M. (2004). Lessons from a comparison of work domain models: Representational choices and their implications. *Human Factors, 46*(4), 711-727.

Burns, C. M., & Vicente, K. J. (2001). Model-based approaches for analyzing cognitive work: A comparison of abstraction hierarchy, multilevel flow modeling, and decision ladder modeling. *International Journal of Cognitive Ergonomics*, *5*, 357-366.

Cooksey, R. W. (1996). *Judgment Analysis: Theory, Methods, and Applications*. San Diego: Academic Press.

Crow, J., Javaux, D. & Rushby, J. (2000). Models and mechanized methods that integrate human factors into automation design. International Conference on Human-Computer Interaction in Aeronautics: (HCI-Aero 2000), pp.163-168, Toulouse, France.

Cummings, M.L. (2006). *Can CWA inform the design of networked intelligent systems?* Paper presented at the Moving Autonomy Forward Conference, June 21-23, Grantham, England.

Degani, A. (2004). Taming HAL: Designing interfaces beyond 2001. New York: Palgrave- MacMillan.

Degani, A., & Heymann, M. (2002). Formal verification of human-automation interaction. *Human Factors*, 44(1), 28–43.

Degani, A., Heymann, M., and Gellatly, A. (2011). HMI aspects of automotive climate control systems. *Proceedings of the 2011 IEEE Systems, Men and Cybernetics Conference*.

Foley, J. D., and Wallace, V. L. (1974). The art of natural graphic man-machine conversation. *Proceeding of the IEEE*, 62(4), 462-471.

Goldstein, W. M. (2006). Introduction to Brunswikian Theory and Method. In A. Kirlik (Ed.), *Adaptive Perspectives on Human-Technology Interaction: Methods and Models for Cognitive Engineering and Human-computer Interaction* (pp. 10-16). New York: Oxford University Press.

Hammond, K. R., & Stewart, T. R. (Eds.). (2001). *The Essential Brunswik*. New York: Oxford University Press.

Hammond, K. R., Stewart, T. R., Brehmer, B., & Steinmann, D. O. (1975). Social judgment theory. In M. F. Kaplan & S. Schwartz (Eds.), *Human Judgment and Decision Processes*. New York: Academic Press.

Jacob, R. J. K. (1986). A specification language for direct-manipulation user interface. *ACM Transactions on Graphics*, 5(4), 283-317.

Jha, P., & Bisantz, A. M. (2006). Applying the Multivariate Lens Model to Fault Diagnosis: Experimental Results and Sensitivity Analyses. In A. Kirlik (Ed.), *Adaptive perspective on human-technology interaction: Methods and models for cognitive engineering and human-computer interaction*. New York: Oxford University Press.

Harel, D. (1987). Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8, 231-274.

Heymann, M. & Degani, A. (2007). Formal analysis and automatic generation of user interfaces: Approach, methodology, and an algorithm. *Human Factors*, 49(2), 311-330.

Hollnagel, E. & Woods, D.D. (1983). Cognitive systems engineering: New wine in new bottles, *Int. J. of Man-Machine Studies*, 18 (6), 583-600

Hollnagel, E. & Woods, D.D. (2005). *Joint cognitive systems: Foundations of cognitive systems engineering*. CRC Press, Boca Raton

Jacob, R. J. K. (1983). Using formal specifications in the design of human-computer interfaces. *Communications of the ACM*, 26(4), 259-264.

Lee, J.D. (2006). Human Factors and Ergonomics in Automation Design. In G. Salvendy (Ed.), *Handbook of Human Factors and Ergonomics* (3rd ed.): John Wiley & Sons, Inc.

Leveson, N. (1995). Safeware: *System Safety and Computers*. New York: Addison-Wesley.

Lind, M. (2003). Making sense of the abstraction hierarchy in the power plant domain. *Cognition, Technology, and Work, 5*, 67–81.

Lintern, G. (2006). Foundational issues for work domain analysis. In Proceedings of the 39th Annual Meeting of the Human Factors and Ergonomics Society (pp. 432-436). Santa Monica, CA: HFES.

Militello, L.G., Dominguez, C.O., Lintern, G., Klein, G. (2010). The role of Cognitive Systems Engineering in the systems engineering design process. *Systems Engineering*, 13 (3), 261-273

Mitchell, C. M. (1987). GT-MSOCC: A domain for research on human-computer interaction and decision aiding in supervisory control systems. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-17(4), 553-572.

Mitchell, C. M., and Miller, R. A. (1986). A discrete model of operator function: A methodology for information display design. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-16(3), 343-357.

Naikar. N., Hopcroft, R., & Moylan, A. (2005) *Work domain analysis: Theoretical concepts and methodology* (DSTO-TR-1665). Fishermans Bend, Australia: Defence Science and Technology Organisation.

Neisser, U. (1976). *Cognition and reality: principles and implications of cognitive psychology*. San Francisco: Freeman.

Norman, D.A. (2002). *The design of everyday things*. New York: Basic Books

Palanque, P., and Bastide, R. (1996). A design life-cycle for the formal design of user interface. In *Proceeding of the BCS-FACS Workshop on the Formal Aspects of Human Computer Interaction*. Shefield, U.K.

Parnas, D. (1969). On the use of transition diagrams in the design of a user interface for an interactive computer system. In *Proceedings of the 24th Annual ACM Conference* (pp. 379-385).

Rasmussen, J. (1979). *On the structure of knowledge - a Morphology of Mental Models in a Man-Machine System Context* (No. Riso-M-2192). Roskilde.

Rasmussen J. & Jensen, A. (1974). Mental procedures in real life tasks: A case study of electronic trouble shooting. *Ergonomics*, 17, 293-307.

Rasmussen, J., Pejtersen, A. M., Goodstein L. P., (1994) *Cognitive systems engineering*. New York: Wiley.

Romera, M. (2000). *Using Finite Automata to Represent Mental Models*. Unpublished Masters Thesis. San Jose, California: San Jose State University.

Rothrock, L., & Kirlik, A. (2003). Inferring rule-based strategies in dynamic judgment tasks: Toward a noncompensatory formulation of the lens model. *IEEE Transactions on Systems Man and Cybernetics, Part A-Systems and Humans*, 33(1), 58-72.

Sarter, N. B., & Woods, D. D. (1994). Pilot interaction with cockpit automation II: An experimental study of pilot's mental model and awareness of the flight management and guidance system. International Journal of Aviation Psychology, 4(1), 1-28.

Sarter, N. B., & Woods, D. D. (1995). How in the world did we ever get into that mode? Mode error and awareness in supervisory control. Human Factors, 37(1), 5-20.

Seong, Y., & Bisantz, A. M. (2002). Judgment And Trust In Conjunction With Automated Decision Aids: A Theoretical Model And Empirical Investigation. In *Proceedings of the Human Factors and Ergonomics Society's 46th Annual Meeting*. Santa Monica: Human Factors and Ergonomics Society.

Seong, Y., & Bisantz, A. M. (2008). The impact of cognitive feedback on judgment performance and trust with decision aids. International Journal of Industrial Ergonomics, 38, 608 - 625.

Seong, Y., Bisantz, A. M., & Gattie, G. (2006). Trust, automation and feedback: An integrated approach. In A. Kirlik (Ed.), *Adaptive perspectives on human-technology interaction: Methods and models for cognitive engineering and human-computer interaction* (pp. 105-113). New York: Oxford University Press.

Strauss, R., & Kirlik, A. (2006). Situation awareness as judgment II: Experimental demonstration. *International Journal of Industrial Ergonomics*, 36(5), 475-484.

Turing, A. M. (1936). On computable numbers with an application to the Entscheidungsproblem. In *Proceedings of London Mathematical Society*, 42, 230-365.

Vicente, K. J. (1999). *Cognitive Work Analysis: Towards Safe, Productive and Healthy Computer-Based Work*. Mahwah, NJ: Erlbaum.

Wasserman, A. I. (1985). Extending state transition diagrams for the specification of human-computer interaction. *IEEE transactions on Software Engineering*, SE-11(8), 699-713.

Woods, W. A. (1970). Transition network grammars for natural language analysis. *Communications of the ACM*, 13, 591-606.