INTERNATIONAL JOURNAL OF COGNITIVE ERGONOMICS, 2001, *5*(3), 335–355

ARTICLES

Comparison of Display Requirements Generated Via Hierarchical Task and Abstraction–Decomposition Space Analysis Techniques

Christopher A. Miller
Smart Information Flow Technologies

Kim J. Vicente
Department of Mechanical and Industrial Engineering
University of Toronto

ABSTRACT

Cognitive work analysis (CWA) techniques are the primary methods available for designers to obtain the knowledge required to create interfaces to complex systems involving cognitive work. There are a wide and growing variety of analysis methods available with a variety of claims for their relative strengths and weaknesses, but it is extremely rare for anyone actually to apply different analytic techniques to the same analysis problem. The work reported here begins to address this gap by directly comparing the information requirements produced by what are probably the 2 most commonly used analysis techniques—Rasmussen's (1985) Abstraction–Decomposition Space (ADS) or "Abstraction Hierarchy" and Shepherd's (1989) Hierarchical Task Analysis (HTA) technique. These 2 approaches were selected because each is well known in the literature, yet they rarely have been directly compared on a common problem. Our comparison shows that the techniques produce different yet complementary information about the interaction needs that human users of a system will have. Both approaches have strengths and weaknesses, but ultimately they reflect different perspectives on (and different avenues to) the knowledge needed for good system and interface design.

1. INTRODUCTION

The practice of human factors or cognitive ergonomics begins with work analysis. This is true both of the field's history, (e.g., time and motion studies [Taylor, 1911]), and of most textbooks' recommendations for proceeding with interface design (Booth, 1989; Norman, 1989; Wickens, 1992). Interface design is the process of shaping displays and controls so that they provide information or interaction capabilities for a user (Woods, 1991), but to know what information and interactions are needed or helpful, the designer must know what data are pertinent to the observer's needs, intentions, expectations, and interests, in interaction with some

Requests for reprints should be sent to Christopher A. Miller, SMArt Information Flow Technologies, 1272 Raymond Avenue, St. Paul, MN 55108. E-mail: cmiller@sift.info.com

system and in what order and relations (Woods, 1986). As the field has evolved, the most powerful methods of providing this knowledge have been work analysis methods.

Much has been written (e.g., Hollnagel & Woods, 1983; Norman, 1984) on the shift in work analysis techniques from observable, physical tasks and manual actions that were the principle concern of the field through the 1960s and 1970s, to increased concern about the cognitive tasks that make up a growing proportion of human work. Work analysis methods have had to adapt from the process of observing and recording physical activities to inferring or sparking reports of cognitive activities (e.g., Diaper, 1989).

There are, however, a wide and growing variety of methods for analyzing cognitive work. Various writers have made claims about the relative strengths and weaknesses of alternate work analysis approaches (Eggleston, 1998; Hollnagel & Woods, 1983; Kirwan & Ainsworth, 1992; Miller & Vicente, 1998b; Vicente, 1999a), but it is extremely rare for researchers to actually apply different analytic techniques to the same design problem, much less tools from separate analytic traditions. When they do (e.g., Ham, 2000), their goal is more likely to be a practical one of more complete examination of the design problem rather than an academic one of examining the strengths and weaknesses and similarities and differences of the analytic tools themselves. As a result, claims for the capabilities of each technique, and their utility to specific tasks and applications of interest to the cognitive ergonomics community, remain somewhat speculative.

The work reported here begins to address this gap in the literature by directly comparing the information requirements produced by what are probably the two most commonly used analysis techniques—Rasmussen's (1985) Abstraction–Decomposition Space (ADS) or "Abstraction Hierarchy" and Shepherd's (1989) Hierarchical Task Analysis (HTA) technique. These two approaches were selected because each is well known in the literature, yet they have not been directly compared on a common problem. We discuss both approaches in separate sections and then present results derived from applying them to the same interface generation problem. Our results illustrate that the techniques produce different yet complementary information about the interaction needs that human users of a system will have. Both approaches have strengths and weaknesses, but ultimately they reflect different perspectives on (and different avenues to) the knowledge needed for good system and interface design.

2. METHODS FOR COMPARISON

2.1. The Analytic Techniques

2.1.1. ADS. Easily the most prominent, well-documented, and frequently used of work analysis techniques that focus on the system or plant to be analyzed is Rasmussen's (1985; Rasmussen, Pejtersen, & Goodstein, 1994) ADS, commonly referred to as the Abstraction Hierarchy. Vicente's techniques for applying the ADS are now also well documented in Vicente (1999a). Extensive worked examples of the ADS in the domain we chose to analyze can be found in Vicente (1996, 1999a), Bisantz and Vicente (1994), Vicente and Rasmussen (1990), and Hunter, Janzen, and Vicente (1995)—and these were the primary sources used to construct the requirements list for the ADS analysis included in Section 3.

The ADS approach involves a thorough analysis of the constraints and capabilities that the physical plant (a.k.a. "system" or "work domain") imposes on work that can be done. An ADS is a two-dimensional modeling tool that captures the means–ends and part–whole relations in the functional structure of a physical system for achieving work goals. These two dimensions together form a matrix, as in Figure 1. This matrix is the ADS.

Each cell in the ADS represents a complete model of the plant and could, conceivably, stand alone. However, much of the power of the ADS comes from understanding the relations between the cells. Thus, a typical ADS analysis will construct multiple models to populate several of the cells in the ADS matrix.

The part–whole, or "decomposition" dimension of the ADS is straightforward. Here, the plant's physical entities are aggregated moving up the axis (or, alternatively, decomposed in moving down the axis). The relation between an entity at an upper level and one at a lower level is "is composed of"—the system as a whole is composed of subsystems that are composed of components.

The means–ends or "abstraction" dimension is somewhat more complicated. Here, moving up the axis means moving from a more concrete to a more abstract description of the system, but the dimension of this abstraction is one of functionality. This means that the lowest level descriptions are highly concrete descriptions of the form and appearance of plant components, but as one moves up the levels, one "abstracts away" from these concrete details and adds more general information not present at the lower levels. For example, there may be no physical component responsible for producing a chemical reaction—thus, the reaction would not show up at the lower, Physical Function level. It would, however, appear at a higher General Function level, and its effects (in terms of mass and energy) would show up at the still higher Abstract Function level. Movement upward along the abstraction dimension is toward progressively more general descriptions of the functions performed by specific, concrete entities.

A useful way of thinking about the abstraction dimension (after Rasmussen, 1985) is as a hierarchy of means–ends relations. This means that relations between any three layers can be characterized by a How–What–Why triad. Attending to a given level means that that is "what" the observer is focused on. The level above answers "why?"—"why is that component or function present in the plant?" Moving down a level from "what" answers "how?"—"how is

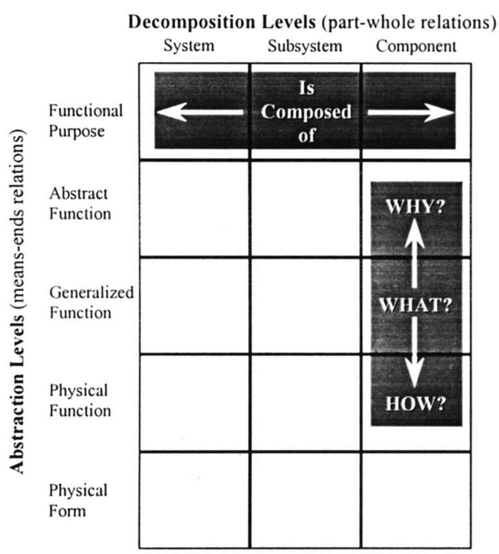

FIGURE 1 The Abstraction Decomposition Space (Rasmussen, 1985; Vicente & Rasmussen, 1990).

the function accomplished?" (in structural and functional terms, not user actions). Note that this How–What–Why window can be overlaid over any three vertical cells in the ADS space to answer the same set of questions about the relations between entities in those cells.

Rasmussen's (1985) ADS approach shares the Gibsonian (Gibson & Crooks, 1938) emphasis on the importance of the "field" in which an actor behaves for "affording" or "constraining" the set of actions that are necessary or appropriate. There is a growing body of empirical work showing that interfaces based on such work domain analyses can lead to better performance than traditional design approaches, particularly in abnormal situations (Vicente, 1996).

ADS analyses typically rely on detailed knowledge of the plant and its interactions with the environment—and on the rules, equations, or models governing these interactions. ADS analyses are performed with data collected not from observations but from discussions with engineers and other experts, and review of design and engineering documents, to understand how and why the structures work together to produce the results they do. When these sources are inadequate, the analysis will be correspondingly inadequate—but even partial and incomplete knowledge can be used to provide a helpful understanding of the work domain (Sharp & Helmicki, 1998).

2.1.2. HTA. There is a great range of work analysis techniques that focus on user tasks and actions (Kirwan & Ainsworth, 1992). For the purpose of our comparative analysis, we chose to use HTA (Shepherd, 1989). HTA is a simple, informal, and representationally streamlined task analysis method, yet one that can be readily extended to capture and organize information requirements. It is also a "basic" tool in that it contains (perhaps simplified versions of) most of the characteristics of even the most complex task modeling tools. HTA also has the advantage of being widely known and used in the task-analytic community: Kirwan and Ainsworth referred to HTA as the "best known task analysis technique" (p. 396). Thus, not only is there substantial written guidance in how to use it, but using HTA makes it easier to communicate our results to the rest of the academic and industrial community.

HTA, as with all task analysis techniques, focuses on "what an operator … is required to do, in terms of actions and/or cognitive processes to achieve a system goal" (Kirwan & Ainsworth, 1992, p. 1). Knowledge about tasks captured in an HTA typically includes both hierarchical, action (as opposed to structural) means–ends relations (how subtasks may be composed to accomplish higher level tasks) and sequential relations (how tasks must be performed temporally). Sources of information for an HTA are typically user interviews or through observation, experimentation, and training; procedural manuals may also be used (Diaper, 1989). Where these sources are absent or break down (e.g., unanticipated situations), the HTA will be impossible or, worse, misleading. When these sources exist reliably, however, failure to incorporate them will result in inefficiencies or errors in training and operations. Information needs (both input and output) are typically deduced for the tasks, and these, combined with the task relation information described previously, can serve as the basis for prioritizing, clustering, filtering, or sequencing information presentation in an interface design.

HTAs can typically be presented or used in at least two formats (cf. Shepherd, 1989), which emphasize different types of knowledge they can capture and represent. A graphical format, like that in Figure 2, shows the hierarchical and aggregate relations between tasks. Each layer of the hierarchy represents a series of tasks or actions that accomplish the higher level ("parent") task in some fashion. A "Plan" is always placed along the vertical line connecting the

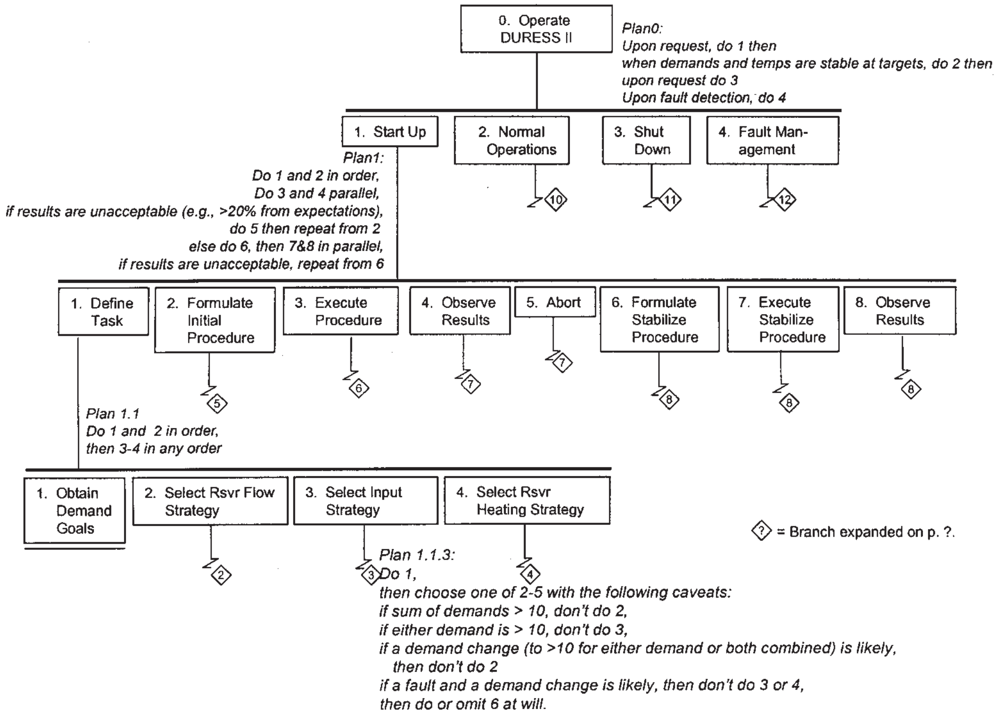

FIGURE 2 Top level HTA for DURESS II.

child tasks to the parent to show how, when, and in what order they must be performed to accomplish the parent task. The plan is where information about the parallel or sequential relations among the tasks and their initiation and completion conditions is represented.

These hierarchical relations captured in this format are means–ends relations, but it is important to note that they are "action" means–ends links (i.e., what actions need to be performed to achieve ends at a higher level). By contrast, an ADS represents "structural" means–ends relations (i.e., what structural degrees of freedom of the system are available to achieve higher level ends). This distinction, although subtle, is at the core of the comparison of the two approaches, as will be seen in the following sections.

HTAs can also be used in a tabular form with progressive indenting and task numbering used to track task decomposition (Shepherd, 1989). Although it is harder to visualize task relations in this format, it is easier to link additional information to tasks—such as frequency, duration information, or both; sequencing information (such as named temporal relations); potential or likely human errors; and information or other resources required when performing the task.

2.2. Analytical Comparison

2.2.1. Motivation. Our purpose was to conduct a direct comparison of the information that an HTA and an ADS, when conducted on the same work domain, provided for a common objective: interface design. As noted earlier, a direct, face-to-face comparison of the results produced by the two methodologies is important to enhance and validate understanding of their relative strengths and weaknesses. Our comparison was not, as will be discussed later, an attempt to determine which technique was better, but rather was focused on whether the techniques provided unique knowledge useful for design.

2.2.2. Comparison domain—DURESS II. For a comparative analysis, we needed a system both simple enough to produce a manageable list of requirements yet complex and realistic enough to maintain face validity vis-á-vis real-world applications. We chose Vicente's (1999a) DURESS II Feedwater simulation as a domain that met both criteria. The following description of DURESS II is from Vicente (1999a; see Vicente, 1996, for more details):

> DURESS (DUal REservoir System Simulation) II is a thermal-hydraulic process control microworld that was designed to be *representative* … of industrial process control systems, thereby promoting generalizability of research results to operational settings. … The physical structure of DURESS II … consists of two redundant feedwater streams (FWSs) that can be configured to supply water to either, both, or neither of two reservoirs. Each reservoir has associated with it an externally determined demand for water that can change over time. The work domain purposes are twofold: to keep each of the reservoir temperatures (T1 and T2) at a prescribed temperature (40 °C and 20 °C, repectively), and to satisfy the current mass (water) output demand rates (MO1 and MO2). To accomplish these goals, workers have control over eight valves (VA, VA1, VA2, VO1, VB, VB1, VB2, and VO2), two pumps (PA and PB), and two heaters (HTR1 and HTR2). All of these components are governed by first order lag dynamics, with a time constant of 15 s for the heaters and 5 s for the remaining components. (pp. 141–142)

The physical layout of DURESS II is illustrated in Figure 3. We chose to work with DURESS II for a variety of reasons. First, it has been used extensively in experiments and analyses at the University of Toronto—hence, there was substantial local expertise in it.

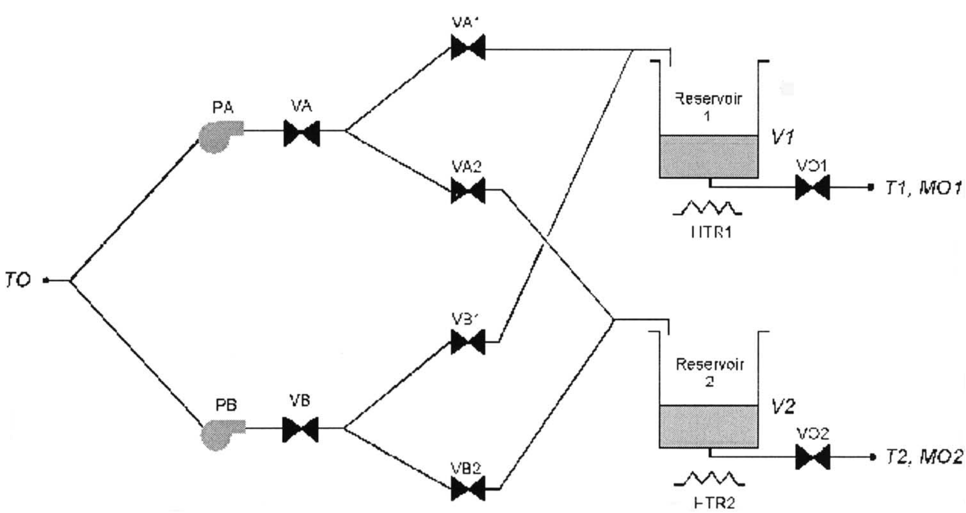

FIGURE 3 Physical layout of the feedwater system simulated in DURESS II.

Furthermore, this research has shown that, although simple enough to be readily understood by a short engineering analysis, it is nevertheless complex enough to permit a wide range of operational strategies and the development of both correct and incorrect mental models when naïve users interact with it (Pawlak & Vicente, 1996). Finally, although extensive ADS analyses of DURESS II have been performed, traditional task analysis methods have generally not been applied to the system. Bisantz and Vicente (1994), Vicente (1996), and Vicente and Pawlak (1994) gave detailed reports of ADS analyses of DURESS II, and we compiled the models produced by those studies to develop the list of ADS requirements knowledge for our comparison described in this article. Thus, DURESS II offered the promise of speeding the work described here while ensuring a measure of independence between the ADS and HTA analyses we performed.

For conducting our HTA of the DURESS II system, we relied on the expertise of engineering graduate students in the Cognitive Engineering Laboratory at the University of Toronto who had extensive experience in the design, implementation, and operation of the DURESS II simulation, as well as documentation of possible and observed user strategies in use of DURESS II (Vicente & Pawlak, 1994). The top level of our HTA for DURESS II, along with a partial expansion of the Start-Up procedure, is included in Figure 2.

2.2.3. Methodology. We chose to compare the requirements produced by the separate analytic techniques, rather than specific displays produced from them, for the following reasons. The natural output of both techniques is a list of requirements around which a user interface may be designed, as illustrated in the simplified depiction of the interface design process in Figure 4. That is, they don't explicitly tell the designer what the display should look like. Instead they provide information about what the display's content should be—requirements for the visual form of the display itself. The designer must then apply creativity, skill, and intuition to creating a visual form to meet those requirements, or as many of them as possible.

The flow of interface design illustrated in Figure 4 provides some implications for how alternate analytic methods should be compared. First, because the analysis method at best produces requirements that are then interpreted and acted on by a designer, comparing designs (as opposed to requirements lists) introduces the confounding factor of the creativity of the designer. Two designers (or the same designer on different days) might produce

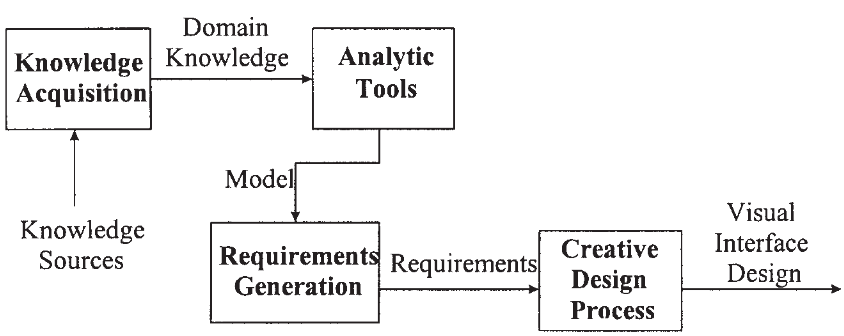

FIGURE 4 Analysis and design in the interface generation process.

better or worse visual designs from the same set of requirements. Similarly, the differences between two designs might be due to the skill and creativity of the designer rather than to the outcomes of the analytic techniques. Second, it is possible that not all requirements can be met (or met equally well) by a given design. Thus, although they are requirements, they may not be manifested in the display ultimately produced. Finally, the prevalence of requirements as a means of communicating across diverse and distributed work groups in large, complex, industrial work settings (e.g., Kruchten, 2000) makes awareness of the types of requirements that can be produced using various techniques important in its own right.

Nevertheless, there can be little doubt that the ultimate proof is "in the pudding." Any analytic technique that consistently fails to produce superior visual interface designs (as measured by comparative performance studies) should be regarded with skepticism. On this front, both ADS and HTA have a proven track record of use in the production of good interfaces for a variety of work domains (cf. Vicente & Rasmussen, 1990, and Rasmussen et al., 1994, for examples and case studies of ADS-based designs and Reed, 1992; Carey, Stammers, & Astley, 1989; and Hackos & Redish, 1998, for similar reports on HTA and other task-analytic approaches to design).

The comparison of analytic techniques reported here is certainly not, nor was it intended to be, a "pure," side-by-side comparison designed to show which analytic method is better. To have performed such a comparison fairly and accurately, we would have had to have at least two individuals, both unfamiliar with the application domain at the start of the experiment and both with at least approximately equal experience with their respective techniques and in interface design in general, perform the respective analyses "from scratch" and in isolation from each other. Not only did we not have access to such individuals, but the question of which analytic technique was better in some absolute sense was not what we were trying to answer.

Instead, we performed the HTA after, and with full knowledge of, the results of the ADS. We were interested in the complementary information produced by the two analyses when used in conjunction. Our hypothesis was that, because both HTA and ADS techniques focus on different aspects of the work environment (tasks and the work domain itself), the two analytic techniques would provide unique information and that information from either analysis would be beneficial but that both together would offer a more complete set of requirements for interface design. In essence, performing one analysis after the other, building on its outputs, is a conservative test of this hypothesis. It might be expected that two separate analyses would produce different results, but if a second analysis can be performed with the full knowledge of the first and still produce novel information, that would be stronger evidence for the unique contribution of each approach.

Our decision to conduct the HTA after, and using the results of, the ADS (rather than vice versa) was one of practicality. As noted previously, work domain analyses of the DURESS II had already been performed and could be utilized.

As Shepherd (1989) pointed out, the purpose for which an HTA is performed can have a profound impact on the information collected. Vicente (1999b) made a similar observation for ADS. Our primary purpose in this exercise was to derive information and control requirements for the human users of DURESS II around which an interface could be designed. Generally speaking, analyses that are focused on producing design requirements place more emphasis on identifying interaction needs but, perhaps, less on decomposing the domain to a fine-grained level (useful to produce procedures or training programs for novices).

Finally, there were a few shortcuts taken in performing this HTA. Because our primary purpose was the comparative analysis of HTA and ADS, we pursued only that much of the

HTA as we thought would provide valuable insights for our purpose. We expanded the Start-Up branch of the DURESS II HTA in depth, with moderate expansion on Normal Operations and Shutdown and limited expansion on Fault Management (six equipment failure faults). In part, this was because of progressively diminishing research returns: having expanded Start-Up first, we found ourselves less likely to identify new classes of requirements in each additional branch expanded. In part, as mentioned previously, the purpose of this HTA (acquiring knowledge to support interface design) did not require a deep, procedural program for every branch. Finally, specifically with regard to the Fault Management branch, we acknowledged the fact that representing comprehensive strategies for this task is ultimately hopeless. Instead, we represented known faults with management strategies—an approach similar to that taken in the process control and aviation industries currently.

3. RESULTS OF ANALYTICAL COMPARISON

The development of requirements sets for even such a moderately complex system as DURESS II produces large quantities of data. The requirements lists produced in this work, themselves summaries of the actual analyses, occupy some 21 single-spaced pages in the laboratory technical report documenting them (Miller & Vicente, 1998a). Clearly, some further summarization is required for presentation in the literature. Table 1 summarizes the results of our two analyses and organizes them in side-by-side fashion for comparison.

The first column in the table presents not specific requirements obtained from either analysis, but rather a general type or class of requirements knowledge that may have been represented by several instances in the analyses. For example, the first line in the table states that the ADS identified the "physical appearance and location of work domain components" as required. In fact, our ADS analysis identified that the physical appearance and location of 14 specific components should be included as follows (from Miller & Vicente, 1998a):

1. All physical components of DURESS II (as identified by the Physical Form level of the ADS) should be represented. These are: Pump A, Pump B, Valve A, Valve B, Valve A1, Valve A2, Valve B1, Valve B2, Reservoir 1, Reservoir 2, Heater 1, Heater 2, Outlet valve D1, Outlet valve D2.
2. Information about the appearance and location of physical components listed in number 1 should be included.

An X in either column means that the corresponding analysis technique clearly and unequivocally identified the type of interface knowledge represented in the row as necessary for an interface in this domain. Other entries claim that an information type was "implicitly" identified by an analytic technique. Note that both HTA and ADS are intended and, in current usage, are generally used as the sole method of identifying display requirements for interface design. Thus, it is not surprising that either approach provides most of the full set of display requirements represented by the union of the two approaches.

It is important that some types of information are only implicitly provided by each technique. Implicit in this context means that some sensitivity to the knowledge type was required to complete the analysis, but that the knowledge wasn't as complete or deep, or as easily or explicitly represented in the implicit technique's outputs as it was in the more explicit one. Therefore, the designer using the implicit technique might do as thorough a job of

TABLE 1
Comparison of the Types of Display Requirements Knowledge Produced by the Two Analytic Techniques

Type of Interface Knowledge Identified in Analysis	*Identified in ADS Analysis?*	*Identified in HTA Analysis?*
1. Physical appearance and location of work domain components	X	
2. Physical connections between components	X	
3. Function and current state of physical components	X	X
4. Range of possible states for physical components	X	Implicit from multiple comparisons
5. Actual current behavior of components (generalized function states: flows and quantities)	X	X
6. Range of possible behaviors of components	X	Implicit from multiple comparisons
7. Capability to achieve (and constraints on) general functional behaviors given the states of physical components	X	Implicit (and partial) in procedures and expectation generation
8. Causal relations between general functions	X	Implicit (and partial) in procedures and expectation generation
9. Aggregation of generalized functions into subsystems	X	X (with notion that subsystem definition might be dynamic)
10. Actual current generalized function state at subsystem level	X	X (with notion that subsystem definition might be dynamic)
11. Range of possible functional states at subsystem level	X	Implicit from multiple comparisons
12. Causal connections between subsystem behaviors	X	Implicit (and partial) in procedures and expectation generation
13. Current state of abstract functions at subsystem level	X	X (with notion that subsystem definition might be dynamic)
14. Range of possible abstract function states at subsystem level	X	Implicit from multiple comparisons
15. Capability to achieve (and constraints on) abstract functional behaviors given generalized functional states	X	Implicit (and partial) in procedures and expectation generation
16. Causal connections between abstract functions	X	Implicit (and partial) in procedures and expectation generation
17. Current state of functional purpose variables for the system as a whole	X	X
18. Range of possible states for functional purpose variables	X	Implicit from multiple comparisons
19. Capability for achieving (and constraints on) overall functional purpose behaviors given abstract functional states	X	Implicit (and partial) in procedures and expectation generation
20. Specific expected or goal value for physical functions	Implicit from functional behavior capability and constraint information	X

(*continued*)

TABLE 1 (*Continued*)

Type of Interface Knowledge Identified in Analysis	*Identified in ADS Analysis?*	*Identified in HTA Analysis?*
21. Specific expected or goal value for general functions	Implicit from functional behavior capability and constraint information	X
22. Specific expected or goal value for abstract functions	Implicit from functional behavior capability and constraint information	X
23. Specific expected or goal value for functional purpose	X (demand values)	X
24. Extra-system goal information (duration or cumulative volume; start, stop, and change requests)		X
25. Social-organizational priority and trade-off information		X
26. Social-organizational information about operational expectations (likelihood of faults, demand changes, etc.)		X
27. Explicit strategy choices and functional implications		Strategy choices only
28. Explicit information to support strategy selection (e.g., sum of D, interface availability)		X
29. Configuration-dependent subsystem groupings and capacities	Static groupings and implicit (derivable) capacities	X
30. Distinction between monitoring and controlling information elements	Capabilities discriminated but no information about when which was needed	X
31. Task-dependent, temporal information clustering (sequential vs. parallel presentation, etc.)	Some capability via means-ends relationships	X

Note. ADS = abstraction–decomposition space; HTA = hierarchical task analysis.

understanding and capturing that knowledge as the one using the explicit technique, but the nature of the technique itself made this less likely. For example, the procedures produced by the HTA are based on the underlying functioning of the DURESS II system, but this knowledge could come as reported procedural rules from domain experts. There is no guarantee that such reports would be complete or even necessarily accurate. Further, the understanding of the system's general capabilities and constraints required to produce accurate procedures is not explicitly captured anywhere in the HTA analysis. Instead, this knowledge is "compiled" (which necessarily means it is obscured) into procedural rules by the HTA. Thus, an HTA implicitly conveys knowledge about the DURESS II system functions, but it does not explicitly convey that knowledge in depth (see also Section 4.7.).

It is important to keep in mind the cumulative nature of the analyses. Because the HTA was performed after, and with the results of, the ADS, the presence of an information type in the HTA column does not mean that HTA alone would have been sure to capture display requirements of that type. Furthermore, the absence of an information type in the HTA column means that the HTA had no reasonable or convenient way of incorporating that type of information, in spite of the fact that the ADS analysis said it was needed. Because the ADS was performed first, without access to the HTA results, the presence of an information type in the ADS column is evidence that ADS alone can identify that requirement type. On the other hand, the absence of an information type in the ADS column means only that the ADS failed to identify that type of information need—not that it could not have incorporated it, especially if the ADS had been performed after the HTA.

Finally, it is important to remember that the generation of display requirements is only a contributor to the ultimate display designed. The fact that an information type is missing from either column leaves open the possibility that a smart designer might intuitively fill that information in. On the other hand, the absence of a display requirement places a heavier burden on the designer's intelligence and creativity, thereby making errors of omission more likely.

4. LESSONS LEARNED AND IMPLICATIONS FOR INTERFACE DESIGN

The most general conclusion from the results summarized in Table 1 is that the two types of analyses do have unique contributions to offer the interface design process, even when performed sequentially. As can be seen from Table 1, not only are the sets of display requirements produced by the two analyses substantially different, they are also highly complementary.

The remainder of this section provides lessons learned from conducting the paired analyses. Many of these involve considerations of the strengths and weaknesses of each approach. When possible, we have drawn specific implications for interface design. We have structured the list as follows: The first five items present advantages to performing the HTA after and in addition to an ADS. The later seven items present disadvantages of doing an HTA alone and, therefore, advantages that the ADS provides when done alone or in addition to an HTA.

4.1. Importance of Method or Strategy Selection

The HTA shows that the operation of DURESS II can be thought of in terms of a handful of task-like strategies or methods (cf. lines 27 and 28 in Table 1). Vicente and colleagues (e.g., Vicente & Pawlack, 1994) have discovered this from engineering control analyses of DURESS II as well, but their interfaces based on the results of ADS analyses alone (e.g., Bisantz & Vicente, 1994; Vicente, 1996; Vicente & Rasmussen, 1990) have not taken full advantage of the fact. Much of the user's interactions with DURESS II are determined by strategy choice (cf. lines 20 to 23 in Table 1): Initial demands and socio-organizational priorities constrain useful strategies, and, once a strategy is chosen, it is reasonably straightforward to determine what specific equipment settings and values should be. Expectations and performance monitoring are also determined by strategy choice, and equipment failures may make a current strategy no longer feasible, therefore mandating a transition to another strategy. Although the ADS provides the information required to derive these strategies, the strategies themselves are not present in the ADS. The HTA more naturally shows how strategies are chosen and used by an operator—as well as identifying the information requirements for making the choice and implementing the strategy. This prevalence of strategy-based reasoning argues that strategies be included in training regimes and, perhaps, as selectable objects in the work environment.

4.2. Importance of Expectations Given Method or Task

A large proportion of the HTA's tasks involve either the generation of expected values for various DURESS II components or the comparison of current values to expected ones. With the exception of mass and temperature output goals (cf. line 23 in Table 1), specific expectation states for intermediate goals or states are not produced by the ADS analysis, though they are specifically included in the HTA (cf. lines 20 to 22 in Table 1). This is in keeping with the ADS goal to capture the constraints present in the work domain and not the specific values associated with any single methodology. The prevalence of expectation values in the HTA tasks suggests that some method of graphically conveying these values, perhaps in a manner sensitive to the current strategy the operator is using, would be helpful to users (cf. lines 27 and 28 in Table 1).

4.3. Ordering Constraints or Practices Should Be Supported

The HTA identifies places where multiple tasks must be done in sequence or in parallel, either because of work domain constraints (e.g., you must have water in a reservoir before you can get flow out of it) or of human cognitive constraints (e.g., you must have a plan before you can execute it). The discipline required to produce an ADS, and the level of "deep knowledge" it requires, facilitate the identification of the first type of constraints (cf. lines 7, 8, 12, 15, 16, and 19 in Table 1), though these are difficult to represent in an ADS model (cf. line 31 in Table 1). The second type of constraints is not a part of the work domain per se and thus is not captured by ADS. Ordering relations are useful for interface design for two reasons. Sequential relations may provide opportunities to suppress information not relevant to a current task (thereby facilitating greater concentration), whereas information for parallel tasks must all be present concurrently. Second, when tasks should be done in sequence, interfaces should be designed to support or, in extreme cases, to enforce that sequence.

4.4. Distinction Between Display and Control

By discriminating between planning or monitoring versus execution tasks, the HTA shows when operators need both control capabilities and displayed information versus displayed information alone (cf. line 30 in Table 1). Although this distinction is not always useful for design (especially if the transition from monitoring to control tasks must happen rapidly and unpredictably), it can sometimes be used to minimize display clutter and focus attention. Although the ADS does identify those variables that can be controlled versus those that can only be monitored (cf. lines 4 to 8 in Table 1), it does not support the identification of periods when display alone might be acceptable because it does not explicitly include the notion of sequencing or temporal flow.

4.5. Importance of Social–Organizational Knowledge

The need for the operator to choose between methods (primarily in Plan 1.1 and its children) implies the need for social–organizational knowledge, which is not a part of the work domain (i.e., the plant) itself and is, therefore, not included in the ADS (cf. lines 24 to 26 and 28 in Table 1). These factors include information about the importance of speed to completion, speed to initiation, consistency of output, perceived likelihood of demand changes, faults, excessive workload levels, and so on. The operator must have this information (though not necessarily through the interface) or he or she will make assumptions about those variables—with potentially erroneous results.

It should be noted, however, that the ADS technique is envisioned as only the first step in a series of constraint-based analyses (Rasmussen et al., 1994). Vicente (1999a) labeled this series *cognitive work analysis* (CWA) and has described their sequence and content as follows:

1. The ADS, which focuses on the Work Domain—that is, the physical plant.
2. The Decision Ladder, which focuses on the control decisions and actions.
3. Information Flow Maps, which analyze viable control strategies.
4. An integration of the other tools used to analyze constraints imposed by the socio-organizational structure.
5. The Skills, Rules, and Knowledge taxonomy, which can be used to analyze worker competency requirements.

Thus, a full CWA would likely incorporate the socio-organizational knowledge requirements described here, whereas an ADS alone would not. A typical HTA, by contrast, strives to represent all actions and considerations in a procedure regardless of why they are there (though note the limitations to this approach discussed in sections 4.8, 4.10, and 4.11). HTA naturally incorporates considerations at all five levels of a full CWA. It will, however, capture these considerations only along a specific trajectory and does not represent the full "space" of constraints and capabilities at each of the CWA levels.

4.6. Sensitivity to Current Displays = Lack of Device Independence

An HTA requires more extensive assumptions about the work context than the ADS—as Vicente (1999b) pointed out—but this can be either good or bad depending on the purpose of the analysis. The ADS must assume, and is therefore sensitive to, only the physical plant. It makes no assumptions about control equipment, interfaces, and so on. The HTA is sensitive to not only the physical plant, interfaces, control equipment, and automation available, but also the social context of goals and incentives in which they are performed (cf. lines 24 and 28 in Table 1). For example, in our analysis, choosing a Reservoir Strategy is critically dependent on whether a specific kind of interface is available (cf. Vicente & Pawlak, 1994).

Generally speaking, "device independence" is more useful in the early stages of design or redesign, when fewer device-relevant decisions have been made, or to the degree that major changes in current work domain or operational practice are being contemplated. Thus, as a gross generalization, HTA is most useful when minor improvements to current interface design and operational practice are intended and, therefore, when current practice and optimization knowledge can be useful, whereas more substantial modifications will be better served by an ADS analysis or, better yet, an ADS followed by an HTA.

4.7. Implicitness of Rationale for Procedural Knowledge or Lack of Deep Knowledge

Although the HTA is better at capturing procedural knowledge, this comes at the cost of losing the deep knowledge required to understand procedures' rationale. Plan 1.1.3 doesn't explain why you should not choose the Single Feedwater System strategy if the sum of demands is greater than 10. To understand why requires more of the deep knowledge about the structure and function of the plant itself—namely, that the capacity of the pumps associated with each feedwater system is only 10 units, thus greater output cannot be sustained. This better capability to capture deep knowledge is illustrated by the ADS's better performance on lines 1 through 19 in Table 1 and the explanatory power that derives from the knowledge represented by those lines.

This might imply that a task-based approach makes a poor foundation for training, but the reality is more complex. In fact, a procedural, task-based training approach will generally enable a novice operator to conduct useful work more quickly than learning deep, structural and functional knowledge. This operator will be lost, however, when the situation deviates from that anticipated in the procedures, whereas the deeply trained operator will have the knowledge required to, perhaps, invent a new procedure on the fly in reaction to a novel situation.

4.8. Difficulty of Being Comprehensive Using HTA

Because HTA captures and represents specific task trajectories, it becomes increasingly unwieldy the more one tries to represent the full set of possible task- and work-domain situations. It is far easier to report "the normal case" or "what I usually do"—and this is frequently how HTA is used. This relation is illustrated by the HTA analysis's partial or implicit performance in capturing many types of knowledge included on lines 1 through 19 in Table 1.

In fact, one of the strengths of the HTA methodology is that its tabular format (cf. section 2.1.2) makes it easy to abbreviate the expansion of branches of the task hierarchy and to incorporate by reference existing branches that have been expanded previously. This engenders two problems for the analyst conducting the HTA, however. First, it raises the problem of having to select which tasks to expand to determine complete coverage of the task domain. Second, even in those cases where all known tasks are analyzed, it leaves open the possibility that unknown or unexpected conditions of use may require the spontaneous creation of novel tasks that will not be well supported by an interface designed around the requirements of known tasks alone.

These facts have three implications for analysis. First, they stress both the importance and the difficulty of maintaining comprehensiveness. Although it may well be possible to design a good interface without performing a comprehensive task analysis (an analysis that examines the information needs of all possible tasks to be performed using the system), such a design leaves open the possibility of missed information requirements and, therefore, of interfaces that are not well suited to some circumstances that may arise. ADS is a good antidote because it captures functional capabilities and constraints of the work domain without trying to articulate all possible trajectories. Second, the facts presented stress the ease of capturing familiar procedures and, by extension, the degree to which workers think in procedures. This suggests that we miss an opportunity to facilitate learning and operations if we don't make use of known, familiar trajectories. Finally, they also show the advantages of doing a task analysis after an ADS: The comprehensiveness of the ADS analysis serves as a framework for the HTA, reminding the analyst about alternatives that need to be investigated and showing him or her where tasks ought to fit once captured.

Even when alternative strategies are known, to the degree that an HTA is prescriptive, it may filter out or suppress capabilities. For example, an optional "Valve Complexity Reduction" strategy is described in Vicente and Pawlak (1994)—opening the initial feed valves (VA and VB) fully and performing all control by limiting this flow via secondary valves (VA1, VA2, VB1, VB2). This is generally a good strategy. It reduces the number of settings the operator has to worry about and provides more flexibility (at lower workload) during later operations. Thus, in the HTA, we made a typical analyst's or designer's decision to build in the Valve Complexity Reduction strategy into the procedures to be followed to achieve start-up (under step 1.2 in Figure 2). We thereby obscured the possibility that start-up is possible without these steps, or under conditions where one of the initial feed valves is stuck open. The temptation to make such streamlining decisions increases as the work associated with a comprehensive HTA increases.

4.9. Lack of Physical Form Information

A glaring absence in the display requirements generated from the HTA is physical form, appearance, and location information (cf. lines 1 and 2 in Table 1). One likely explanation is that this is another manifestation of the lack of deep knowledge obtained via HTA. HTA's procedures compile out (cf. Section 4.7.) the need for deep knowledge, including knowledge about the physical form and location of equipment—as long as the contextual assumptions under which the task trajectories were created hold true. That is, if I wish to provide feedwater at a specified flow rate and temperature via DURESS II, I can do it by manipulating switches and setting values via the interface as prescribed in the HTA (as long as initial assumptions hold true). I don't need to know anything more about the system—such as where the pumps controlled by the interface are located or what they look like.

If true, the implications of this conclusion are that an ADS analysis might provide more detailed and deeper display requirements than are, in fact, necessary during normal (i.e., anticipated) operations, but this information may be critical in those situations where operators can no longer rely on "cookbook" procedures. Vicente made a similar point (1999a, 1999b).

4.10. Procedures for Procedure's Sake

We note also the tendency for the analyst to create procedures precisely because they fit the HTA analytic framework. One example of this is the use of procedures to describe working methodologies that may be more dynamic or less well structured. The HTA representation of a task may artificially impose a procedure on what is, in practice, a more adaptive, satisficing decision-making process for the human operator who, after all, must plan the order and method of conducting subtasks as a part of each task performed. As Suchman (1987) and Klein (1998) both documented thoroughly, procedural descriptions of this decision-making and planning process are rarely complex and situationally dependent enough to be completely accurate.

Another example of the overuse of procedural representations is the creation of procedural simplifications to ensure that the user is "on track"—that is, entering the procedure from an expected state to which it applies, rather than from other possible states. The Abort task (1.5 in Figure 2) is an example in two ways. First, Abort's parent plan says that Abort should be performed if results of a start-up are "not acceptable"—notionally defined as more than 20% off expectations. At best, this is a gross and conservative simplification because many situations would permit larger deviations and still be recoverable. Second, the Abort task itself is a plan to place the system into a configuration from which the written procedures apply. The activities in this task are not, strictly speaking, necessary in all contexts. Analyzing task sequences for all possibilities becomes exponentially difficult, so the analyst is tempted to include conservative good practice rules, or to build "parking configurations" that get the work domain into a state where a more simplified procedure can be applied to it. Although this simplification reduces workload for the analyst or designer, and frequently for the user as well, it obscures work domain capabilities that could, if used properly, lead to better context-adaptive performance. It also enables potential mismatches between the assumptions of a procedure and the intentions of the user, as Suchman (1987) documented. This is one reason that those performing an HTA also frequently perform a human error analysis (e.g., Reed, 1992) and include information requirements derived from that analysis along with those from the HTA to enable a broader range of error detection and recovery capabilities in the interface.

4.11. Lack of Relation Propagation Knowledge

Perhaps the most serious lack noted in the results of the HTA is the absence of requirements about the propagation of effects from one equipment variable or state to another (cf. lines 7, 8, 11, 12, 14 to 16, and 19 in Table 1). That is, the HTA showed little need to include the relations identified and represented as equations in the ADS analysis.

Again, the primary reason for this stems from the intent of the HTA to produce (or describe) effective procedures or rule-like plans for accomplishing specific goals. Thus, the designer must reason about the propagation relations and compile them into rules or procedures. This strategy of performing some work at design time so that the operator doesn't have to do it at run time is where the efficiency of procedures originates. Of course, if the designer has not correctly and completely anticipated the set of procedures needed, then the operator at run time will be forced to generate a new procedure on the fly. If the operator does not understand the propagation effects between work domain variables, then that new procedure may very well be critically flawed (cf. Vicente, 1999b).

4.12. Leap to Information Requirements

An HTA carried out to the depth here is most useful for generating requirements about how to organize information (spatially and temporally). HTA seems less useful than an ADS for directly identifying the information required for the tasks. We attempt to illustrate this subtle point by an example.

The ADS identifies a series of variables and equations that describe the work domain and then claims, supported by empirical evidence (Vicente, 1996) that an operator needs to know these variable values and equation-based relations if he or she is to understand and control the domain. Thus, the ADS directly identifies specific information requirements and provides a thorough justification for their inclusion in an interface design.

An HTA is capable of providing this level of directness and justification and does so most frequently when it describes fine-grained cognitive operations. In our HTA, for example, Task 1.1.2.1.1 describes a cognitive operation called "Sum the Demands," which requires, as inputs, the two demand values D1 and D2. To perform the parent task, we know both what information is needed and explicitly why it is needed (and how it is to be used). Thus, this level of decomposition provides both a specific identification of information requirements and thorough rationale for their inclusion. It is far more common in practice to decompose tasks to a level like that in Task 1.6.1.1, "Determine Flow Adjustments," and then use introspection or operator reports to generate a list of information requirements for this task without creating explicit sub-procedures for performing it. We refer to this as making the "leap" to information requirements. Again, as discussed in section 2.1.2, the tabular format available for conducting an HTA is useful precisely because it facilitates this leap to information requirements at a higher level task than they would otherwise arise in—it allows the inclusion of information requirements without a detailed decomposition of the cognitive tasks and processes that make use of that information. By making this leap, the designer or analyst is making two assumptions: (a) that he or she has the right set of information requirements and (b) that the operator will know how to combine them to perform the task.

Although analyses could be driven to the level where requirements are explicitly identified, it is worth investigating why the drive to make the leap is prevalent. The deeper one drives the HTA, the bigger the branching logic becomes. Working through this combinatorial explosion becomes tedious, time-consuming, and costly. In industrial settings, all three factors contribute pressure to speed analysis, but even in academic environments the first two may be sufficient.

5. CONCLUSIONS

Taken over the findings listed here, the following conclusions seem valid. The ADS work domain analysis

- Does a much better job of providing deep knowledge about the full set of constraints and capabilities for system behavior that are inherent in the work domain—that is, explicit knowledge about the affordances of the domain and their relation.
- More readily and directly identifies information requirements for monitoring, controlling, and diagnosing the system.
- Is more independent of the specific context in which the system is used (e.g., its interface, organizational goals, social structure, etc.).

In contrast, the HTA task analysis

- Provides compiled procedural knowledge that will generally be easier to learn and follow for anticipated cases but that hides the deeper rationale for procedures and risks unexpected behavior.
- Is more "human-centered" in that it focuses more on what the operator must or can do and how he or she divides the set of operational behaviors into discrete chunks (i.e., tasks)—that is, it takes the human and human action as its primary focus and not the system and system state.
- More readily identifies when, how, and with what priority information will be needed to perform expected tasks.
- Is less independent of the context of use, which is to say it requires a more comprehensive consideration of the full set of factors that influence operator behavior.

Our analyses also emphasized the complimentary nature of the two tools. ADS provides deep and comprehensive knowledge about the functional structure of the work domain, but (by itself) omits constraints imposed on work by dimensions outside the physical plant—by the social organization, human capabilities, available control and interface equipment, and so on. It also omits possible efficiencies in known operating procedures for specific contexts. By contrast, HTA provides these strengths but is prey to omitting work domain capabilities and is generally poor at capturing and conveying the rationale for the actions it identifies.

These conclusions are in keeping with, but extensions and validations of, the conceptual analysis of work domain- and task-based analytic techniques reported in Vicente (1999a, 1999b) and Miller and Vicente (1999). A useful analogy developed there helps to clarify the strengths and weaknesses of each approach: Work domain based techniques (of which ADS is an exemplar) provide map-like information about the work environment; task-based techniques (of which HTA is an exemplar) provide directions-like information. To understand the "terrain" of work to be performed in an environment, ADS maps are broader in their coverage and provide better and more comprehensive capabilities to adapt to unforeseen contingencies and recover from errors—but they are effortful to use, requiring users to determine their own set of directions for any given goal. By contrast, HTA provides a precompiled set of directions that can be more efficient and can include nondomain related information—but these can fail to capture the full set of constraints and capabilities in the domain and can therefore be more narrow, brittle, and limited in the knowledge they provide. The analytical comparison presented earlier provides data to support these previous theoretical analyses and also provides suggestions as to why these attributes are as they are.

There are significant advantages to doing both analyses. Completing the ADS first provided a firm grounding in system functioning—more thorough and better organized than is frequent when doing a task analysis alone. This supports the argument that when the design requires a deep grounding in system capabilities (perhaps because it involves a physical system that is novel or complex or must provide deep knowledge for a user), it will be valuable to begin with an understanding of the plant as provided by an ADS. On the other hand, completing the HTA provided information that the ADS did not and identified specific procedures within the general capabilities of the work domain that were known to be efficient and useful.

Would there be advantages to performing the task analysis first? Although we did not take this approach, we can draw some inferences about the type of knowledge that might be gleaned. We would expect the analyst using a task-based approach alone to develop a better sense of how the operator currently behaves but a comparatively impoverished knowledge about how or why those behaviors are effective. Doing the task analysis first might provide a better sense of the sequence of tasks, but to truly support those tasks in novel situations (e.g., with a novel interface or new automation), he or she would need to draw on deep knowledge to explain or predict new user behavior. This points to two observations: First, if the ADS were to be done after the HTA, then the focus should be on explaining observed or reported task sequences and perhaps identifying unusual or unreported cases for discussion with users. Second, one reason that we might want to use a task analysis before or even instead of an ADS analysis is if the problem under study required a deep understanding of how user's think about the task currently—for example, to create a training program to familiarize current workers with a novel interface or automation capability.

Those with a practical bent will ask if it is worth doing two separate analyses. We cannot, on the basis of this study, provide a definitive answer beyond pointing out that substantially different, complimentary, and useful types of information were produced by both techniques. Whether this additional information will result in interfaces that produce better human performance—the ultimate test—remains for future work.

ACKNOWLEDGMENTS

This work has been partially funded by a research grant from NOVA Chemicals Ltd., through the support of the Canadian Natural Sciences and Engineering Research Council, and through internal research support of the Honeywell Technology Center.

We would like to thank Jamie Errington, Dave Weizenbach, Greg Jamieson, and Peter Bullemer for their assistance in developing, steering, and conducting this project.

REFERENCES

Bisantz, A., & Vicente, K. (1994). Making the abstraction hierarchy concrete. *International Journal of Human–Computer Studies, 40,* 83–117.

Booth, P. (1989). *An introduction to human–computer interaction.* Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.

Carey, M., Stammers, R., & Astley, J. (1989). Human–computer interaction design: The potential and pitfalls of Hierarchical Task Analysis. In D. Diaper (Ed.), *Task analysis for human–computer interaction* (pp. 56–74). Chichester, England: Ellis Horwood.

Diaper, D. (1989). *Task analysis for human–computer interaction.* Chichester, England: Ellis Horwood.

Eggleston, B. (1998). Cognitive engineering: The latest fad or a true step forward as an approach to complex multi-person system analysis and design? In *Proceedings of the Human Factors and Medical Panel of RTO (AGARD) Collaborative Crew Performance in Complex Operational Systems Symposium* (pp. 15.1–15.3). Edinburgh, England: AGARD.

Gibson, J. J., & Crooks, L. E. (1938). A theoretical field-analysis of automobile-driving. *American Journal of Psychology, 51,* 453–471.

Hackos, J. T., & Redish, J. C. (1998). *User and task analysis for interface design.* New York: Wiley.

Ham, D.-H. (2000). *Ecological design of information content and layout for process control based on abstraction hierarchy.* Unpublished doctoral dissertation, Korea Advanced Institute of Science and Technology.

Hollnagel, E., & Woods, D. (1983). Cognitive systems engineering: New wine in new bottles. *International Journal of Man–Machine Studies, 18,* 583–600.

Hunter, C., Janzen, M., & Vicente, K. (1995). *Research on factors influencing human cognitive behavior (II)* (Tech. Rep. CEL–95–08). Toronto: Cognitive Engineering Laboratory, University of Toronto.

Kirwan, B., & Ainsworth, L. (1992). *A guide to task analysis.* Bristol, PA: Taylor & Francis.

Klein, G. (1998). *Sources of power.* Cambridge, MA: MIT Press.

Kruchten, P. (2000). *The rationale unified process: An introduction.* Reading, MA: Addison-Wesley.

Miller, C., & Vicente, K. (1998a). *Comparative analysis of display requirements generated via task-based and work domain-based analyses: A test case using DURESS II* (Tech. Rep. CEL–98–08). Toronto: Cognitive Engineering Laboratory, University of Toronto.

Miller, C., & Vicente, K. (1998b). Toward an integration of task and work domain analysis techniques for human–computer interface design. *Proceedings of the 1998 meeting of the Human Factors and Ergonomic Society* (pp. 336–340). Chicago: HFES.

Miller C., & Vicente, K. (1999). Task "versus" work domain analysis techniques: A comparative analysis. In *Proceedings of the 1999 meeting of the Human Factors and Ergonomic Society*. Houston: HFES.

Norman, D. (1984). Cognitive engineering principles in the design of human–computer interfaces. In G. Salvendy (Ed.), *Human–computer interaction* (pp. 11–16). Amsterdam: Elsevier.

Norman, D. (1989). *The design of everyday things.* New York: Doubleday.

Pawlak, W., & Vicente, K. (1996). Inducing effective operator control through ecological interface design. *International Journal of Human–Computer Studies, 44,* 653–688.

Rasmussen, J. (1985). The role of hierarchical knowledge representation in decision making and system management. *IEEE Transactions on Systems, Man and Cybernetics, 15,* 234–243.

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

Reed, J. (1992). A plant local panel review. In B. Kirwan & L. Ainsworth (Eds.), *A guide to task analysis* (pp. 267–288). Bristol, PA: Taylor & Francis.

Sharp, T. D., & Helmicki, A. J. (1998). The application of the ecological interface design approach to neonatal intensive care medicine. In *Proceedings of the Human Factors and Ergonomics Society 42nd annual meeting* (pp. 350–354). Santa Monica, CA: Human Factors and Ergonomics Society.

Shepherd, A. (1989). Analysis and training in information technology tasks. In D. Diaper (Ed.), *Task analysis for human–computer interaction* (pp. 15–55). Chichester, England: Ellis Horwood.

Suchman, L. (1987). *Plans and situated actions: The problem of human–machine communication.* Cambridge, England: Cambridge University Press.

Taylor, F. (1911). *The principles of scientific management.* New York: Harper & Row.

Vicente, K. J. (1996). Improving dynamic decision making in *complex* systems through ecological interface design: A research overview. *System Dynamics Review, 12,* 251–279.

Vicente, K. J. (1999a). *Cognitive work analysis: Towards safe, productive, and healthy computer-based work.* Mahwah, NJ: Lawrence Erlbaum Associates, Inc.

Vicente, K. (1999b). Wanted: Psychologically relevant, device- and event-independent work analysis techniques. *Interacting with Computers, 11,* 237–254.

Vicente, K., & Pawlak, W. (1994). Cognitive Work Analysis for the DURESS II System (Tech. Rep. CEL–94–03). Toronto: Cognitive Engineering Laboratory, University of Toronto.

Vicente, K., & Rasmussen, J. (1990). The ecology of human–machine systems: II. Mediating "direct perception" in complex work domains. *Ecological Psychology, 2,* 207–249.

Wickens, C. (1992). *Engineering psychology and human performance* (2nd ed.). New York: HarperCollins.

Woods, D. (1986). Paradigms for intelligent decision support. In E. Hollnagel, G. Mancini, & D. Woods (Eds.), *Intelligent decision support in process environments* (pp. 153–173). New York: Springer-Verlag.

Woods, D. (1991). The cognitive engineering of problem representations. In G. Weir & J. Alty (Eds.), *Human–computer interaction and complex systems* (pp. 169–188). London: Academic.