Comparative Analysis of Display Requirements Generated via Task-Based and Work Domain-Based Analyses: A Test Case Using DURESS II

Christopher A. Miller
And
Kim J. Vicente

CEL 98-08

Cognitive Engineering Laboratory Department of Mechanical & Industrial Engineering University of Toronto
5 King's College Rd. Toronto, Ontario, Canada M5S 3G8
Phone: +1 (416) 978-7399 Fax: +1 (416) 978-3453
Email: benfica@mie.utoronto.ca URL: www.mie.utoronto.ca/labs/cel/

Director: Kim J. Vicente, B.A.Sc., M.S., Ph.D.

The Cognitive Engineering Laboratory (CEL) at the University of Toronto (U of T) is located in the Department of Mechanical & Industrial Engineering, and is one of three laboratories that comprise the U of T Human Factors Research Group. CEL began in 1992 and is primarily concerned with conducting basic and applied research on how to introduce information technology into complex work environments, with a particular emphasis on power plant control rooms. Professor Vicente's areas of expertise include advanced interface design principles, the study of expertise, and cognitive work analysis. Thus, the general mission of CEL is to conduct principled investigations of the impact of information technology on human work so as to develop research findings that are both relevant and useful to industries in which such issues arise.

Current CEL Research Topics

CEL has been funded by Atomic Energy Control Board of Canada, AECL Research, Alias|Wavefront, Asea Brown Boveri Corporate Research - Heidelberg, Defense and Civil Institute for Environmental Medicine, Honeywell Technology Center, Japan Atomic Energy Research Institute, Natural Sciences and Engineering Research Council of Canada, Nova Chemicals, Rotoflex International, and Westinghouse Science & Technology Center. CEL also has collaborations and close contacts with the Mitsubishi Heavy Industries and Toshiba Nuclear Energy Laboratory. Recent CEL projects include:

- Studying the interaction between interface design and adaptation in process control systems.
- Understanding control strategy differences between people of various levels of expertise within the context of process control systems.
- Developing safer and more efficient interfaces for computer-based medical devices.
- Designing novel computer interfaces to display the status of aircraft engineering systems.
- Developing and evaluating advanced user interfaces (in particular, transparent UI tools) for 3-D modelling, animation and painting systems.

CEL Technical Reports

For more information about CEL, CEL technical reports, or graduate school at the University of Toronto, please contact Dr. Kim J. Vicente at the address printed on the front of this technical report.

Unified Modeling Project

UT/NCL/HTC/NSERC

Comparative Analysis of Display Requirements Generated via Task-Based and Work Domain-Based Analyses; A Test Case Using DURESS II

A Report of work under Tasks 4 ("Task Model Analysis") and Task 5 ("Develop Model Integration Approach") of the NOVA/UofT Task Breakdown (Jan. 13, 1998)

Release Date: 20 September, 1998
Document Version: 1.01
Filename: ReqtsComp.doc

Submitted to:
Jamie Errington, NOVA Chemicals, Ltd.

Prepared by: Chris Miller and Kim Vicente
Honeywell Technology Center &
University of Toronto Cognitive Engineering Laboratory

1. Document History

.01 First draft, compiled on 7/9/98.
.02 Fleshed out the placeholders for sections 3.2, 3.3and 6. Added. 7.11. Produced on 7/24/98.
.03 Fleshed out placeholders for sections 3.4 and 8, added section 9, made minor corrections to section 4 and to the HTA analyses in the appendices, added illustration to section 3.3, minor additions to 5, refined and reorganized section 7. Completed 8/7/98.
1.0 Added Summary and cleaned up references. Distributed on 8/9/98.
1.01 Minor revisions in response to comments from Jamie Errington. Inclusion of Cognitive Engineering Laboratory (CEL) numbers and cover pages. Distributed 9/20/98.

2. Summary

2.1 Objectives and Outcomes

The purpose of this report is to provide a direct comparison of the types of information which a work-domain analysis and a task analysis can provide for the purpose of interface design. While we have made claims about strengths and weaknesses of each analytic approach in the past (see the NSERC proposal and Miller &Vicente, 1998a) it is rare for anyone to analyze the same application problem twice using separate tools. Thus, a direct, 'face-to-face' comparison of the results produced by two representative work domain- and task-based analysis methodologies was important as a part of our overall investigations.

We chose to investigate the interface *requirements* produced by different analytic tools (instead of interfaces produced from those requirements) because the analyses naturally produce requirements; interfaces must be developed from them through human creativity. We chose to do this comparative analysis on Vicente's DURESS II system because it is representative of many industrial process control systems, is well-studied, yet is small enough to be manageable and because work domain analyses using the Rasmussen's (1985) Abstraction Decomposition Space (ADS) analysis technique (more commonly known as the Abstraction Hierarchy) had already been repeatedly performed on it. To provide the basis for comparison between the two types of analytic techniques, we performed a task analysis on the DURESS II system using a representative, familiar, yet easy-to-use technique known as Hierarchical Task Analysis (HTA-- Shepherd, 1989).

We were not attempting to conduct a 'pure,' side by side, 'shoot off' comparison designed to show which analytic method was 'better'. Instead, we were interested in the *complimentary* information produced by task analyses and work domain analyses when used in conjunction. Ultimately, if we are to justify the creation of a modeling technique and/or representation which unifies task- and work domain-based analyses, we must prove that there are unique contributions from each perspective. In essence, performing one analysis after the other, building on its outputs, is a conservative approach to demonstrating that point. Related questions included:

- Would the two different techniques produce qualitatively different types of knowledge about how an interface should be designed?
- Would they produce the same types of information but in quantitatively different ways (that is, by using one technique after the other, would it be possible to get more, if similar, display requirements knowledge)?
- Would doing one type of analysis first facilitate the doing of the other analysis? Would it improve the quality of the results produced?

The most general conclusion from our comparison of the two analytic techniques is that the analyses *do* have unique contributions to offer the interface design process, even when performed sequentially. As can be seen from the table in section 6, not only are the sets of display requirements produced by the two

analyses substantially different, they are also highly complimentary. Loosely speaking, the following conclusions seem valid:

The ADS work domain analysis:

- Does a much better job at providing 'deep knowledge' about the full set of constraints and capabilities for system behavior which are inherent in the work domain.
- More readily 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.)

The HTA task analysis:

- Provides 'compiled' procedural knowledge which, while being easier to learn and follow for anticipated cases, hides the deeper rationale for why procedures work and risks unexpected behavior in unexpected situations.
- Is more 'human-centered' in that it focuses more on what the operator must or can do and how s/he divides the set of operational behaviors into discrete chunks (i.e., tasks).
- More readily identifies when, how and with what priority information will be needed to perform expected tasks.
- Is less independent of context, but requires a more comprehensive consideration of the full set of factors which influence operator behavior.

2.2 Report Organization

Section 3 presents a variety of background information important for understanding the comparative analysis which is the focus of this report including (1) an argument for why a comparison between the two techniques should be done at the level of the display requirements they produce rather than of the displays themselves, (2) a more detailed description of each of the analytic techniques and a conceptual comparison of them, (3) a description of the physical system we analyzed for our comparison—the DURESS II feedwater simulation system, and (4) a more detailed description of the nature and objectives of the comparison we performed. Section 4 presents the display requirements produced as the results of an ADS analysis. Section 5 and Appendices A and B present the results of the HTA analysis of DURESS II. Section 6 provides, in tabular form, a comparison of the types of display requirements knowledge produced by each analysis. Section 7 reports our conclusions and observations about the complimentary nature of the two techniques.

Section 8 takes the results of the HTA analysis one step further by using them to suggest task-based modifications to the current DURESS II interfaces (which were designed using the outputs of ADS analyses alone), thus illustrating that not only do the two analytic techniques produce different types of display requirements knowledge, but they also can be used to produce different, and complimentary, visual interface forms. Finally, our thoughts about what the performance of these analyses in sequence suggests for a unified modeling and analysis technique are reported in section 9. This section contains thoughts about the relative merits of taking either a task-based or a work domain-based point of view *first* in a paired analysis approach, as well as thoughts about the points of intersection of the two types of analyses and their implications for constructing a unified modeling approach. Section 10 contains the references cited throughout the report, and the two Appendices, as mentioned above, contain the results of the HTA analysis in two different formats.

3. Objectives, Rationale and Caveats

The purpose of this report is to provide a direct comparison of the types of information which a task and a work-domain analysis can provide for the purpose of interface design. While we have made claims about strengths and weaknesses of each analytic approach in the past (see the NSERC proposal and Miller &Vicente, 1998a) and these claims have the benefit of our experience and intuitions, it is rare for anyone to analyze the same application problem twice using separate tools. Thus, a direct, 'face-to-face' comparison of the results produced by two representative task- and work domain-based analysis methodologies was important as a part of our overall investigations.

3.1 Why compare requirements?

In this report, we have created lists of requirements for the generation of a visual display from two different analyses of Vicente's (1996) DURESS II system. While the utility of a list of requirements is ultimately less than a full display, there are various reasons for believing that this may be a more profitable way of comparing the outputs of the alternate modeling and analysis techniques than comparing complete displays. All analysis techniques we are exploring naturally end at the requirements phase of the design process, as illustrated in Figure 1. That is, they don't explicitly tell the interface designer what the display should look like. Instead they provide information about what the display's contents should be and, perhaps, how individual items should relate to each other—that is, requirements for the visual form of the display itself. The designer must then apply creativity, skill and intuition to creating a visual form which meets those requirements, or as many of them as possible.

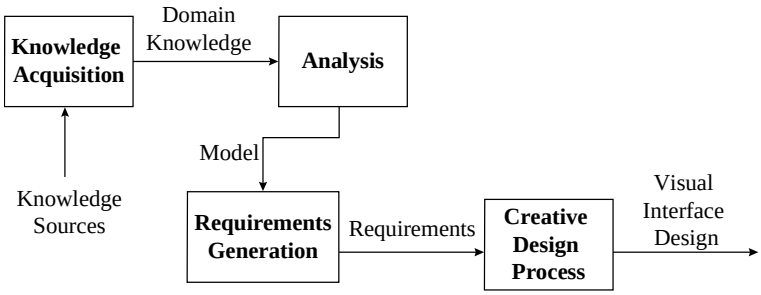


Figure 1. Analysis and design in the interface generation process.

Figure 1 provides some implications for how alternate analytic methods should be compared. First, since the analysis method at best produces requirements which are then interpreted and acted upon 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 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 which is ultimately produced. This, again, is the 'fault' of the designer (and/or of the display resources available) and not of the analytic technique. A final reason for examining the requirements produced by the various analytic techniques is the prevalence of requirements as a means of communicating across diverse and distributed work groups in large, complex industrial work settings. As interface development efforts become larger, the plausibility of a single individual who first performs an analysis and then proceeds immediately to interface design decreases. Thus, awareness of the types of requirements that can be produced using various techniques has important implications in its own right.

For the above reasons, we have decided to examine requirements rather than the end product of design as a means of comparing analytic techniques. Nevertheless, there can be little doubt that the ultimate proof is 'in the pudding.' Any analytic technique which consistently fails to produce superior visual interface designs (as measured by comparative performance studies) should be regarded with skepticism.

3.2 Analytic Methods Compared

.

The most common work analysis techniques used for the purpose of interface design can be divided into two types based on their primary focus. Task analysis (TA--Kirwan & Ainsworth, 1992; Diaper, 1989) describes the actions that an actor can or should take to accomplish goals. Work domain analysis (WDA_ techniques (Rasmussen, Pejtersen & Goodstein, 1994; Rasmussen, 1985), which we have also called "system-based" analysis techniques for reasons described below, examine the functional structure of the domain (specifically, the plant or system) in or on which work must be done.

Prior to the work described in this report, we claimed (Miller & Vicente, 1998a) that each approach has strengths and weaknesses, though ultimately they reflect different perspectives on (and different avenues to) the full set of knowledge needed for good human-centered system design. A comparison of the strengths and weaknesses of these techniques (derived primarily from introspection) is presented in Table 1. One purpose of the work described in this report was to validate and refine the claims made in Table 1.

Table 1. Relative advantages and disadvantages of TA and WDA forms of work analysis (and, by extension, of interfaces designed from information obtained via these analytic techniques).

	TASK	WORK DOMAIN
Mental economy	efficient	effortful
Ability to adapt to unforeseen contingencies	brittle	flexible
Scope of applicability	narrow	broad
Ability to recover from errors	limited	unlimited

Below, we describe both task and work domain analysis approaches in separate sections and our selection of specific, representative analytic techniques within each category.

3.2.1 Task-based analysis and design methods

Task analysis (TA) techniques have a long and productive history in human factors. Kirwan and Ainsworth (1992), in their comprehensive work on the vast variety of TA methods, define TA "… as the study of what an operator… is required to do, in terms of actions and/or cognitive processes to achieve a system goal." Thus, TA methods are explicitly about the actions that an actor can or should take to achieve a goal. The focus of TA is the action, however, not the work domain. Knowledge about tasks captured in analysis typically includes either hierarchical, means-ends relationships (how subtasks may be composed to accomplish higher level tasks) or sequential relationships (how tasks must be performed temporally in order to be successful), or both. Sources of information for TAs are typically user interviews, though observation, experimentation and training or procedural manuals may also be used (Diaper, 1989). Where these sources are absent, and in those circumstances where task knowledge breaks down (e.g., unanticipated situations), TA will be impossible, or worse, misleading. When these sources do exist reliably, however, failure to incorporate them into design 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 relationship information described above, can serve as the basis for prioritizing, clustering, filtering, or sequencing information presentation elements in an interface design. Task-linked information requirements serve as a particularly powerful basis for constructing "context" sensitive (actually, user intent, goal or procedure) interfaces (Funk and Miller, 1997) since they can dynamically filter information on the basis of the current user information needs (Rouse, Geddes, and Curry, 1988; Miller, Funk and Hannen, 1997).

For the purpose of our comparative analysis, we chose to use a specific task analysis method known as Hierarchical Task Analysis (HTA--Shepherd, 1989). While not the most complex or representationally powerful TA technique, HTA has the strengths of being extensively used in a wide variety of application areas, familiar to most researchers and practitioners, and is quick and easy to use and to adapt to most analytic needs.

3.2.2 Work domain analysis and design methods

Work domain analysis (WDA) techniques are more recent additions to the repertoire of interface design tools. WDA techniques emphasize the structure of the work domain—that is, the plant or equipment on and with which the user must achieve some set of functional goals. This is why we have also referred to WDA techniques as "system-based" analyses, in contrast with the task-based analyses described above.

Most current work in this area derives from Rasmussen's (1985) abstraction-decomposition space (ADS)—commonly, if somewhat incorrectly, referred to as the 'Abstraction Hierarchy' (AH). An ADS is a two-dimensional modeling tool that can be used to conduct a WDA in complex sociotechnical systems. Rasmussen's approach, shares the Gibsonian (Gibson & Crooks, 1938) emphasis on the importance of the "field" or ecology in which an actor behaves for determining or "constraining" the set of actions which are necessary or appropriate. There is a growing amount of empirical support showing that interfaces based on such work domain analyses can lead to better performance than traditional interface approaches, particularly in abnormal situations (Vicente, 1996).

The ADS provides a comprehensive analysis of the means-ends and part-whole relationships in the functional structure of the process being controlled. It is important to note, however, that while some TA approaches represent means-ends relationships, these are 'action' means-ends (i.e., what actions need to be performed in order to achieve ends at a higher level). By contrast, an ADS represents 'structural' means-ends relationships (i.e., what structural states of the system are required in order to achieve higher level ends).

WDA relies on a detailed knowledge of the plant and its interactions with the environment—and on the rules, equations or models governing these interactions. When these sources are inadequate, the analysis will be correspondingly inadequate—but this situation is less common than might be expected. The greatest threat to the safety of process control systems is events that are not familiar to operators and that have not been anticipated by designers (Vicente & Rasmussen, 1992). Under these challenging circumstances, the operator's role is one of adaptive problem solver. Because the event has not been anticipated by system designers, the available procedures, experience, and automated aids are not directly applicable. The one thing that does remain unchanged, however, is the functional structure of the plant and the principles that govern its interactions with the environment. Further, it is precisely within these constraints that the operator must improvise a solution.

3.2.3 Comparison of the Techniques

Task-based models are like directions for navigation: they identify the actions that human operators should take for particular situations; system-based models are more like maps because they emphasize the overall structure of the plant, independent of any particular situation. Task models are efficient because they identify the information and prioritize it for pre-defined classes of situations, whereas system models are more robust because they identify the functional relationships that are potentially relevant for all situations.

Table 1 above shows our initial assumptions about the comparative strengths and weaknesses of TA and WDA. TAs (and interfaces designed from them) are efficient because they identify what needs to be done, and perhaps how. But as a result of this economy, TAs do not provide the support required to adapt to unanticipated events. TAs are narrow in their generality because they are only applicable to the tasks that have been identified up front, and generally, only to specific ways of doing those task. In task-sensitive interfaces, efficiency is accomplished by suppressing information not pertinent to specific, active tasks, but this may risk loss of accurate, overall knowledge of process state. While context-sensitivity can be

accomplished by adapting the interface to specific work domain states, this frequently presupposes an implicit task-orientation and may undercut the comprehensiveness of information availability described above. Finally, again due to their narrow, brittle, procedural orientation, TAs are also limited in their ability to support recovery from errors.

WDAs (and interfaces derived from them) have a complementary set of strengths and weaknesses. Their primary disadvantage is that they do not tell workers what to do or support them specifically in what they are currently doing. As a result, WDAs put greater demands on workers and may lose efficiency by failing to support specific methods that are known to work in specific conditions. Yet WDAs are generally flexible because they provide workers with the information they need to generate an appropriate response, on-line in real-time, to events which have not been anticipated by system designers. Moreover, WDAs also have a broader scope of applicability. Because they show what the system is capable of doing they provide workers with the discretion to meet the demands of the job in a variety of ways that suit their preferences or the particular needs of the moment. Finally, for the reasons already discussed, WDAs also provide workers with the support they need to recover from errors.

We have assumed that the complementary strengths and weaknesses of TAs and WDAs imply that it would be useful to include both techniques in a single, integrated framework for work analysis and interface design. Initial thoughts about methods for accomplishing this integration can be found in Miller & Vicente (1998a) and in the prior research report from this project (Miller & Vicente, 1998b).

3.3 Application Problem—DURESS II

The following description is from Vicente, (in press):

> 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 reservoirs at a prescribed temperature (generally, 40° C and 20° C), and to satisfy the current mass (water) output demand rates. 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 seconds for the heaters and 5 seconds for the remaining components. DURESS II is described in more detail in Vicente, 1996.

The physical layout of DURESS II is illustrated in Figure 2. 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 and elsewhere—hence, there is a substantial amount of local expertise in it. Furthermore, this

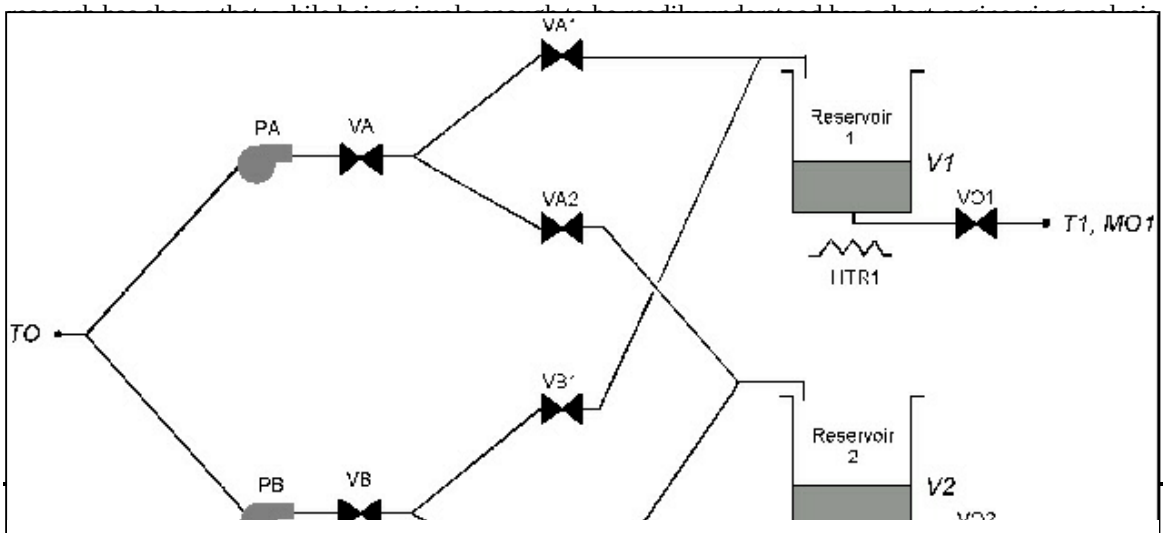


Figure 2. Physical Layout of the feed water system simulated in DURESS II.

3.4 Nature of the "Comparison"

The comparison of analytic techniques reported in this document is certainly not, nor was it intended to be, a 'pure,' side by side comparison designed to show which analytic method was 'better'. Such a comparison is sometimes called a 'shoot off'. To have performed such a comparison fairly and accurately, we would have had to have at least two individuals, both unfamiliar with the DURESS II system 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 were interested in the *complimentary* information produced by task analyses and work domain analyses when used in conjunction. Ultimately, if we are to justify the creation of a modeling technique and/or representation which unifies task- and work domain-based analyses, we must prove that there are unique contributions from each perspective. In essence, performing one analysis after the other, building on its outputs, is a conservative approach to demonstrating that point. 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.

Related questions included:

- Would the two different techniques produced qualitatively different types of knowledge about how an interface should be designed?
- Would they produce the same types of information but produce it in quantitatively different ways (that is, by using one technique after the other, would it be possible to get more, if similar, display requirements knowledge)?
- Would doing one type of analysis first facilitate the doing of the other analysis? Would it improve the quality of the results produced?

Our decision to conduct the task analysis after, and using the results of, the work domain analysis was one of practicality. As noted in Section 3.3 above, work domain analyses of the DURESS II system had already been performed and could be leveraged in the work reported here.

The results from the two analyses are reported in sections 4 and 5. Section 6 provides, in tabular form, a comparison of the types of display requirements knowledge produced by each analysis. Section 7 summarizes our conclusions and observations about the complimentary nature of the two techniques. Section 8 takes the results of the HTA analysis one step further by using them to suggest task-based modifications to the current DURESS II interfaces (designed using the outputs of AH analyses alone), thus illustrating that not only do the two analytic techniques produce different types of display requirements knowledge, but they also can be used to produce different visual interface forms. Finally, our thoughts about what the performance of these analyses in sequence suggests for a unified modeling and analysis technique are reported in section 9 below.

4. Requirements from Abstraction-Decomposition Analysis

Extensive information about, and worked examples of, the use of Rasmusen's (1985) Abstraction-Decomposition analytic technique as applied to the DURESS II system can be found in Vicente (1996), Bisantz & Vicente (1994), Vicente and Rasmussen (1990), and Hunter, Janzen and Vicente (1995)—and these were the primary sources used to construct the requirements list included below.

Tentative list of requirements from the Abstraction-Decomposition Space Analysis of DURESS II

1. All physical components of DURESS II (as identified by the Physical Form level of the ADS) should be represented. These are:
   a) Pump A
   b) Pump B
   c) Valve A
   d) Valve B
   e) Valve A1
   f) Valve A2
   g) Valve B1
   h) Valve B2
   i) Reservoir 1
   j) Reservoir 2
   k) Heater 1
   l) Heater 2
   m) Outlet valve D1
   n) Outlet valve D2
2. Information about the appearance and location of physical components listed in 1 should be included.
3. All physical functions of DURESS II (as identified by the Physical Function level of the ADS) should be represented. That is, the state of the components identified above should be illustrated. These are:
   a) Pump A setting
   b) Pump B setting
   c) Valve A setting
   d) Valve B setting
   e) Valve A1 setting
   f) Valve A2 setting
   g) Valve B1 setting
   h) Valve B2 setting
   i) Reservoir 1 level
   j) Reservoir 2 level
   k) Reservoir 1 temperature
   l) Reservoir 2 temperature
   m) Heater 1 setting
   n) Heater 2 setting
   o) Valve D1 setting
   p) Valve D2 setting
4. In addition to actual state, the range of possible states for each component should also be shown.
5. Connections between physical functions should be illustrated. These include the following facts:
   a) Pump A and Pump B both take water input from a common inlet
   b) Valve A is connected via a pipeline after Pump A
   c) Valve B is connected via a pipeline after Pump B
   d) Valve A is not connected to valve B
   e) Valve A1 is connected via a split pipeline to Valve A
   f) Valve A2 is also connected via a split pipeline to Valve A, but is not connected to Valve A1
   g) Valve B1 is connected via a split pipeline to Valve B
   h) Valve B2 is connected via a split pipeline to Valve B, but is not connected to Valve B1
   i) Valve A1 and B1 are connected via pipelines to Reservoir 1
   j) Valve A2 and B2 are connected via pipelines to Reservoir 2
   k) The Heaters are not physically connected to the reservoirs
   l) Valve D1 is connected by pipeline to Reservoir 1
   m) Valve D2 is connected by pipeline to Reservoir 2

6. All generalized functions from the DURESS II ADS should be illustrated at the component level. That is, the actual flow and storage of heat and water should be represented distinctly from the settings described above. In practice, this means the following values should be represented:
   a) The flow of water through Pump A
   b) The flow of water through Pump B
   c) The flow of water through Valve A
   d) The flow of water through Valve B
   e) The flow of water through Valve A1
   f) The flow of water through Valve A2
   g) The flow of water through Valve B1
   h) The flow of water through Valve B2
   i) The quantity of water stored in Reservoir 1
   j) The quantity of water stored in Reservoir 2
   k) The flow of heat into Reservoir 1
   l) The flow of heat into Reservoir 2
   m) The quantity of heat stored in Reservoir 1
   n) The quantity of heat stored in Reservoir 2
   o) The flow of heat from Heater 1
   p) The flow of heat from Heater 2
   q) The flow of heat and water through Valve D1
   r) The flow of heat and water through Valve D2
7. In addition to actual, current value for each of the items in 6, the range of possible values for each component should also be shown.
8. The fact that the value of each component's generalized function can be affected by changes in that component's physical function state should also be clearly illustrated either through the appearance or the behavior of the interface (or both). These include the following facts:
   a) The flow of water through any secondary valve (VA1, VA2, VB1, VB2) is affected by the setting of that valve (assuming proper functioning) according to the following equation (but omitting first order time lags):

      $$\text{Flow through secondary valve (t)} = \frac{\text{Flow through corresponding primary valve (t)} * \text{Valve Setting (t)}}{\text{Valve Setting (t)} + \text{Setting of other secondary valve (t)}}$$

   b) The flow of water through any primary valve (VA or VB) is determined by the flow through the corresponding pump (assuming proper functioning, but omitting first order time lags).
   c) The flow of water through either pump is determined by the following equation pump (assuming proper functioning, but omitting first order time lags):

      If pump is OFF, the flow = 0, else

      If the Σsettings of the pump's downstream secondary valves is greater than the setting of the downstream primary valve, the pump flow = primary valve setting, else

      Pump flow = the Σsettings of the pump's downstream secondary valves.

   d) The flow of heat from the heaters is affected by the setting of that heater (assuming proper functioning) according to the following equation (but omitting first order time lags):
      Flow from heater (t) = Heater Setting (t)
9. Connections between generalized component functions should be illustrated indicating direction of affectivity or causality. The fact that the value of each generalized function can be affected by changes in other component's generalized function states should also be clearly illustrated either through the appearance or the behavior of the interface (or both). These include the following facts:
   a) The flow of water from each reservoir both constrains and is constrained by the amount water in the reservoir
   b) The flow of heat from each reservoir both constrains and is constrained by the amount heat in the reservoir

c) The amount of water in each reservoir is constrained by the flow out of the reservoir (through the corresponding demand valve) and the flow into the reservoir (through the two inlet valves for each reservoir).

d) The amount of heat in each reservoir is constrained by the flow out of the reservoir (through the corresponding demand valve) and the temperature and flow into the reservoir (through the two inlet valves for each reservoir).

e) The flow of water through valve A1 is constrained by the flow of water through Valve A and through Valve A2 according to the following equation (first order time lags are not included, assumes proper functioning):

$$\text{Flow through Valve A1(t)} = \frac{\text{Flow through Valve A(t)} * \text{Setting of Valve A1(t)}}{\text{Setting of Valve A1(t)} + \text{Setting of Valve A2 (t)}}$$

f) Flows through Valves A2, B1 and B2 behave similarly to the description in e.

g) Flow through Valve A is constrains and is constrained by the flows through valves A1 and A2 according to the following equation (first order time lags are not included, assumes proper functioning):

$$\text{Flow through Valve A(t)} = \text{Flow through Valve A1(t)} + \text{Flow through Valve A2(t)}$$

h) Flow through Valve A is also constrained by the flow through Pump A according to the following equation (first order time lags are not included, assumes proper functioning):

$$\text{Flow through Valve A(t)} = \text{Flow through Pump A (t)}$$

i) Flow through Valve B is similar to A as described in g and h.

j) Flow through Pump A constrains and is constrained by the flows through valves A, A1 and A2 according to the following equation (first order time lags are not included, assumes proper functioning):

If pump is OFF, the Flow through pump A = 0, else
If [setting of valve A1(t) + setting of Valve A2(t)]>=
Setting of Valve A (t), then Flow through pump A(t) = setting of Valve A(t),
Else flow through pump (t) = Setting of Valve A1(t) + Setting of Valve A2 (t)

10. Generalized functions should also be represented at the Subsystem level, and the interface should make it clear that the subsystems consist of the following components. These are:
    a) Heat and Water Input subsystem 1: Pump A and Valves A, A1 and A2
    b) Heat and Water Input subsystem 2: Pump B and Valves B, B1 and B2
    c) Water Holding system 1: Reservoir 1
    d) Water holding system 2: Reservoir 2
    e) Heat holding system 1: Reservoir 1
    f) Heat holding system 2: Reservoir 2
    g) Heat transfer system 1: Heater 1
    h) Heat transfer system 2: Heater 2
    i) Heat and Water removal system 1: Valve D1
    j) Heat and Water removal system 2: Valve D2
11. Connections between generalized subsystem functions should also be illustrated indicating direction of affectivity or causality either through the appearance or the behavior of the interface (or both). These include the following facts:
    a) That Heat and Water input system 1 can be used to input both heat and water to both Water Holding system 1 and 2 and to Heat holding system 1 and 2.
    b) That Heat and Water input system 2 can be used to input both heat and water to both Water Holding system 1 and 2 and to Heat holding system 1 and 2.
    c) That Heat transfer system 1 can only input heat to Heat holding system 1.
    d) That Heat transfer system 2 can only input heat to Heat holding system 2.
    e) That the water and heat in Water holding system 1 and heat holding system 1 can both affect and be affected by Removal system 1.
    f) That the water and heat in Water holding system 2 and heat holding system 2 can both affect and be affected by Removal system 2.
12. All abstract functions identified in the ADS should be represented in the interface. Current values and ranges for these entities should be represented at the subsystem level. These are:

a) Energy Source 1
b) Energy Source 2
c) Mass Source 1
d) Mass Source 2
e) Energy Inventory 1
f) Energy Inventory 2
g) Mass Inventory 1
h) Mass Inventory 2
i) Energy Sink 1
j) Energy Sink 2

13. The fact that the value of each subsystem's abstract function can be affected by changes in the subsystem's generalized function (or lower level functions) should also be clearly illustrated in the interface either through the appearance or the behavior of the interface (or both). This includes the following facts:

a) The only mass sources are the corresponding heat and water input subsystems.
b) The mass input at any time is the output of the heat and water input subsystem for that mass source according to the following equations (first order time lags are not included, assumes proper functioning):
Mass Input 1(t) = Flow through Valve A1(t) + Flow through Valve B1(t)
Mass Input 2(t) = Flow through Valve A2(t) + Flow through Valve B2(t)
c) The mass inventory is achieved by the corresponding water holding systems.
d) The rate of change in the mass in an inventory at any time is the mass in the corresponding water holding system, which is in turn the flow in less the flow out according to the following equations (first order time lags are not included, assumes proper functioning):

$$\text{rate of change in Volume in Mass Inventory 1(t)} = \frac{\text{Flow thru Valve A1(t) + Flow thru Valve B1(t) - Flow thru Valve D1(t)}}{\text{density of water}}$$

e) Energy source 1 (for each subsystem) corresponds to (is achieved by) the heat and water input system for that subsystem.
f) Energy source 2 (for each subsystem) corresponds to (is achieved by) the heat transfer system for that subsystem.
g) The energy input at any time is the energy output of the heat and water input subsystem for that energy plus the energy output of the heat transfer system for that subsystem according to the following equations (first order time lags are not included, assumes proper functioning):
Energy Input 1(t) = Energy from Heater 1(t) + specific heat capacity of water * inlet temperature of the water * Mass Input 1(t)
Energy Input 2(t) = Energy from Heater 2(t) + specific heat capacity of water * inlet temperature of the water * Mass Input 2(t)
h) The energy inventory for each subsystem is achieved by the heat holding system for that subsystem.
i) The energy in an inventory at any time is the energy in the corresponding holding system according to the following equation (first order time lags are not included, assumes proper functioning):
Energy in Inventory(t) =
Temperature of the water(t) * Volume of water(t) * specific heat capacity * density
j) The rate of change in energy in an inventory at any time is the energy in the corresponding energy holding system, which is in turn is given by the following equation (first order time lags are not included, assumes proper functioning):
Rate of change in Temperature of Energy Inventory 1(t) =

$$\frac{\text{Flow from HTR1(t) - [Valve A1flow (t) + Valve B1flow(t)] * [Temp Rsvr1(t) - Inlet Temp] *}}{\text{Volume of Reservoir 1 * specific heat capacity *}}$$

specific heat capacity * density density

k) Mass sink and Energy sink for each subsystem is achieved only by the Removal system for that subsystem.
l) The flow of mass in the sink is determined by the setting of the corresponding outlet valve (D1 or D2).
m) The flow of energy in the sink is given by the following equations (first order time lags are not included, assumes proper functioning):
Flow of Energy thru Sink(t) =
Setting of outlet Valve(t) * Temperature of the water (t)

14. Connections between abstract subsystem functions should be illustrated indicating direction of affectivity or causality. The fact that the value of each abstract function can be affected by changes in other abstract function states should also be clearly illustrated either through the appearance or the behavior of the interface (or both). These include the following facts:
    a) Mass Sources increase Mass inventories but mass inventory has no impact on mass source.
    b) Mass sinks decrease mass inventory (as well as draining mass sources), but mass sinks are constrained by the availability of inventory and source flow.
    c) Energy sources increase energy inventory but energy inventory has no impact on energy sources.
    d) Energy sinks decrease energy inventory (as well as draining energy sources), but energy sinks are constrained by the availability of the inventory and source flow.
15. Functional purposes uncovered in the ADS analysis should be represented at the system level and included in the interface. These are:
    a) Target Temperature (for Reservoir) 1
    b) Target Temperature (for Reservoir) 2
    c) Demand Target (for Reservoir) 1
    d) Demand Target (for Reservoir) 2
16. In addition to these targets, actual current values for functional purposes should also be shown. These are:
    a) Current Temperature of Reservoir 1
    b) Current Temperature of Reservoir 2
    c) Current Flow through D1
    d) Current Flow through D2
17. The fact that the value of each subsystem's functional purposes can be affected by changes in the subsystem's abstract functions (or lower level functions) should also be clearly illustrated in the interface either through the appearance or the behavior of the interface (or both). This includes the following facts:
    a) That changes in mass source, inventory or sink can affect the corresponding demand goal.
    b) That changes in either mass source, inventory or sink or energy source, inventory or sink can affect the corresponding temperature goal.

5. Requirements from Task Analysis

5.1 Task Analysis Methodology and its Rationale

While the ADS methodology does not itself capture information about task organization or task information requirements, the broader approaches to Cognitive Work Analysis being studied at the University of Toronto do provide tools for capturing and representing task knowledge. These include the Decision Ladder (Rasmussen, 1974, 1976) for analysis of tasks and information flow maps (Rasmussen, 1980, 1981) for analysis of strategies for performing tasks. These tools have been applied to DURESS II, albeit less

formally and repeatedly than ADS, in the following work: Vicente, (in press), Bisantz & Vicente (1994), and Vicente & Rasmussen, (1990).

These sources contain information about the tasks and strategies for operating DURESS II generated not from user interviews or observations, but rather from an analysis of the capabilities and constraints inherent in the work domain. I used these sources, along with a few interviews with engineers trained in the operational model DURESS II uses, to construct a Hierarchical Task Analysis (HTA) of the DURESS II system. A more traditional approach to conducting a task analysis would have been to interview "domain experts" trained in the use of the system (but not necessarily in its operational model from an engineering perspective) to derive their 'mental model' or way of thinking about operating the system. Instead, I relied on engineering models, the Abstraction Hierarchy analysis itself, control task and strategy analyses performed from the engineering model and interviews with users who were more familiar with the system performance and behavior than in its regular operation.

This deviation from normal task analysis begs two questions. First, why perform the task analysis at all? Why not rely on the control task and strategy analyses performed by UT personnel on the engineering model of the system? My reasons for this course are largely pragmatic: I am more familiar with HTA than I am with the methods described above and more confident in deriving display requirements from it than from them. More importantly, the control task and strategies analyses have been investigated much less thoroughly than the ADS analyses at UT, and they have not been used to systematically derive or organize display requirements. Thus, I felt that the HTA would be a reliable and familiar way to translate the knowledge obtained from the control task and strategy analyses into display requirements, as well as perhaps adding some more human-centered task knowledge on its own.

Second, why use HTA as opposed to any of the myriad other task analytic methods available? HTA is a simple, informal and comparatively impoverished task analysis method, yet one which can be readily extended to capture and organize information requirements. It is, however, 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. Thus, not only is there substantial written guidance in how to use it, but using HTA would make it easier to communicate our results to the rest of the academic and industrial community. Finally, we are investigating the use of alternative task representations in another thrust within this project (cf. Miller & Vicente, 1998a).

As Shepherd (1989) and others have pointed out, the purpose for which one performs a task analysis can have a profound impact on the types of information collected. Very loosely speaking, there are three primary purposes for which a task analysis can be conducted: (1) to provide knowledge about how an interface to support the tasks should be designed, (2) to identify operational knowledge to be conveyed to a novice user in training, and (3) to create procedures for use by any user in operating the plant. Our primary purpose in this exercise was #1. A task analysis focused on producing design requirements places more emphasis on identifying the information needs of users following the tasks in the analysis—but less emphasis on ensuring that the tasks are decomposed to a fine enough level to ensure performance by a novice. At least one of our lessons learned (cf. Section 7.15) is directly related to this emphasis.

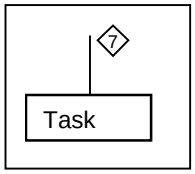
Finally, I should say a few words about the short cuts taken in performing this HTA. Since my primary purpose was the comparative analysis of HTA and ADS, I pursued only that much of the HTA as I thought would provide valuable insights for my purpose. I expanded only the 'start up' branch of the DURESS II HTA in depth, with moderate expansion on Normal Operations and Shut down and very shallow expansion on Fault Management. In part, this was to save time. In part, it was because of progressively diminishing research returns (having expanded 'Start Up' first, I found myself less likely to learn new things in each successive additional branch expanded). In part, as mentioned above, the purpose of this HTA (acquiring knowledge to support interface design) did not require a deep, procedural 'program' for every branch of tasks and activities. Finally, specifically with regards to the Fault Management branch, it was also an acknowledgement of the fact that representing comprehensive strategies for this task is ultimately hopeless.

Instead, we represented a few known faults and their management strategies—an approach similar to that taken in much of the process control and aviation industries currently.

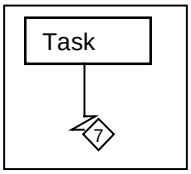
5.2 Analysis Results and Formalisms

The results of the HTA for DURESS II are presented in two different formats in Appendices A and B (after Shepherd, 1989). Appendix A presents the task analysis in graphical form, emphasizing the 'layout' of the tasks—their hierarchical and aggregate relationships. In this format, each layer of the the hierarchy represents a series of tasks or actions which accomplish the higher level ('parent') task in some fashion. A 'Plan' is always placed along the vertical line connecting the child tasks to their parent to show how/when/in what order they must be performed in order to accomplish their parent task. The plan is where information about the parallel or sequential relationships among the tasks and their initiation and completion conditions is located.

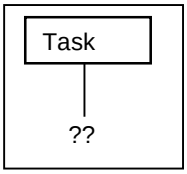
Since the analysis is far too large to fit on a single page, the following conventions were used to link the hierarchical graph across pages:



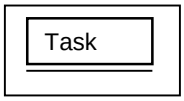
Indicates that the task named in the box is an expansion from a parent task that is found on page 7.



Indicates that the task named in the box is expanded on page 7.



Indicates that the task named in the box is expandable and, in a full task analysis, probably should be expanded, but that the task was not expanded for our analysis in the interests of time, resources and since it was deemed unlikely to produce any novel findings.



Indicates that the task is at its 'primitive' level and should be operational for a user. Not even a full HTA should bother decomposing this task further. See Shepherd (1989) for a discussion of 'stopping criteria' in HTA.

Appendix B presents the HTA for DURESS II in tabular form. While it is harder to visualize task relationships in this format, it is easier to link additional information to tasks. We have included three additional columns of information beyond the task relationships themselves. The first contains frequency and/or duration information about the tasks, as well as information about their sequencing (shown as shaded boxes spanning the table cells with a named temporal relationship between tasks: e.g., sequential, parallel, etc.) The second column contains general notes about the tasks. The final additional column contains information requirements identified for each task. Note that only some cells in the information requirements column are filled in. This is not because the other tasks have no information requirements, but because we have only provided information for the lowest level or 'leaf' task in any hierarchy. This

implements the heuristic that information requirements for parent tasks are simply the aggregate of the information requirements of their children.

6. Comparison of Results

Table 2 provides a side-by-side comparison summarizing the types of knowledge obtained from the two analyses. It necessarily summarizes the specific data provided by the analyses and, therefore, eliminates many of the critical specifics from the two analyses. Thus, for more detail, the reader should review the analyses themselves carefully. Further, in the interests of providing a concise comparison, it has occasionally been necessary to make generalizations in the table below. While exceptions to these claims are possible, we believe they hold true in general.

It is important to keep in mind the cumulative nature of the analyses. Since the HTA was performed *after*, and using the results of, the ADS, the presence of an information type in the HTA column does not mean that an 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 that it was needed. Since the ADS was performed first, without access to the HTA results, the presence of an information type is evidence that ADS alone can identify that type of information. 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 that information, especially if the ADS had been performed after the HTA.

Some explanation should be provided with regards to the entries which 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 outcomes of the two approaches. It is important that some types of information are only 'implicitly' provided by each technique. 'Implicit' in this context, generally means that some sensitivity to the type of display requirements knowledge was required in order to complete the analysis, but that the knowledge required 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' analytic technique *might* do as thorough a job of understanding and capturing that knowledge type as the one using the explicit technique, but that the nature of the technique itself made this less likely. For example, the procedures produced in the HTA require an understanding of the underlying functioning of the DURESS II system, but this knowledge could come in the form of 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 that it is obscured) into procedural rules by the HTA. Thus, an HTA 'implicitly' conveys knowledge about the DURESS II system functions, but they do not 'explicitly' capture or convey that knowledge in depth (see also sections 7.10 and 7.12 below).

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

Table 2. 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?

Type of Interface Knowledge Identified in Analysis	Identified in ADS analysis?	Identified in HTA analysis?
Physical appearance and location of work domain components	X	
Physical connections between components	X	
The function and current state of physical components	X	X
Range of possible states for physical components	X	Implicit from multiple comparisons
Actual current behavior of components (Generalized function states: flows and quantities)	X	X
Range of possible behaviors of components	X	Implicit from multiple comparisons
Capability to achieve (and constraints on) general functional behaviors given the states of physical components	X	Implicit (and partial) in procedures and expectation generation
Causal relationships between general functions	X	Implicit (and partial) in procedures and expectation generation
Aggregation of generalized functions into subsystems	X	X (with notion that subsystem definition might be dynamic)
Actual current generalized function state at subsystem level	X	X (with notion that subsystem definition might be dynamic)
Range of possible functional states at subsystem level	X	Implicit from multiple comparisons
Causal connections between subsystem behaviors	X	Implicit (and partial) in procedures and expectation generation
Current state of abstract functions at the subsystem level	X	X (with notion that subsystem definition might be dynamic)
Range of possible abstract function states at subsystem level	X	Implicit from multiple comparisons
Capability to achieve (and constraints on) abstract functional behaviors given generalized functional states	X	Implicit (and partial) in procedures and expectation generation
Causal connections between abstract functions	X	Implicit (and partial) in procedures and expectation generation
Current state of functional purpose variables for the system as a whole	X	X
Range of possible states for functional purpose variables	X	Implicit from multiple comparisons
Capability for achieving (and constraints on) overall functional purpose behaviors given abstract	X	Implicit (and partial) in procedures and expectation generation

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

7. Conclusions and Lessons Learned

The most general conclusion 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 the table in section 6 above, not only are the sets of display requirements produced by the two analyses substantially different, they are also highly complimentary. Loosely speaking, the following conclusions seem valid:

The ADS work domain analysis:

- Does a much better job at providing 'deep knowledge' about the full set of constraints and capabilities for system behavior which are inherent in the work domain.
- More readily 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.)

The HTA task analysis:

- Provides 'compiled' procedural knowledge which, while being easier to learn and follow for anticipated cases, hides the deeper rationale for why procedures work and risks unexpected behavior in unexpected situations.
- Is more 'human-centered' in that it focuses more on what the operator must or can do and how s/he divides the set of operational behaviors into discrete chunks (i.e., tasks).
- More readily identifies when, how and with what priority information will be needed to perform expected tasks.
- Is less independent of context andrequires a more comprehensive consideration of the full set of factors which influence operator behavior.

The remainder of this section provides a list of lessons learned while conducting the analyses. Many of these lessons involve considerations of the strengths and weaknesses of each approach. We have somewhat arbitrarily structured this list as follows: the first 7 items loosely present advantages to performing the task analysis after and in addition to an ADS work domain analysis. The later 8 items loosely present disadvantages of doing a task analysis alone and, therefore, advantages that the ADS provides when done alone or in addition to an HTA. Our conclusions about the complimentary nature of these two requirements analysis approaches are developed further in section 9 below.

7.1 Importance of method 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. Much of the user's interactions with DURESS are determined by strategy choice: initial demands and socio-organizational priorities constrain the set of 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.

While these strategies were, in fact, identified by a Cognitive Work Analysis of DURESS II, based on an initial ADS analysis, they were not present in the ADS analysis itself. The HTA more naturally shows how the strategies are chosen and used by an operator—as well as the information requirements both for making the choice and for implementing the strategy.

This prevalence of strategy-based reasoning argues for the inclusion of strategies in any training regime and, perhaps, as first-order, manipulable objects in the work environment. I would particularly suggest that any decision aiding be organized around a shared understanding of the strategy choices between human and machine. Again, suggestions for modification to the DURESS II interface using this recommendation are included in section 8.1 below.

7.2 Importance of expectations given method/task

A large proportion of the tasks for operating DURESS II involve either the generation of expected values for various DURESS II components or the comparison of current values to the expected ones. By 'expected values' here, I mean something like 'given my understanding of the current state of the system, I expect this value to be X'. Thus, an expected value is not necessarily the same as a commanded value (I've commanded a downstream valve to provide 10 units of water, but I know that I've constrained the flow to 8 units at an upstream valve, thus my expectation for flow from the downstream valve is only 8 units—after lag effects). It is also not necessarily the same as a goal value (I may want or need 10 units of flow downstream, but the fact that the upstream valve is stuck means there's no way I can get it.) Nevertheless, in a healthy, steady state, thoroughly understood system expected values should equal commanded values, goal values (and actual values, of course). In fact, in an extremely abstract sense, the Normal Operations task can be thought of as simply a monitoring to ensure that all current states are equal to goal/demand states, Fault Management is initiated whenever these are not true, and Start up and Shut down both involve

the translation of high level goal states into specific goal states for each piece of equipment and the generation of intermediary expectation states corresponding to a plan for transitioning from current state to goal state. The information requirements for many tasks in the HTA make explicit this need for expected states or values for many equipment variables.

With the exception of mass and temperature output goals, specific expectation states are not produced by the ADS analysis of DURESS II, nor are they generally included in the DURESS II interfaces. This is in keeping with the goal of ADS to capture the constraints present in the work domain, and not the specific targets associated with any single methodology or trajectory through the work domain. Not surprisingly, therefore, the DURESS II interface tends to be good at conveying absolute equipment-based constraint boundaries, but less good at indicating whether, for example, a specific valve setting is in keeping with a strategy or method of achieving the overall goal.

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 approach or strategy the operator is using, would be helpful to users. Various methods for conveying expectation values are described in 8.2 below.

7.3 Sequencing Constraints/Practices should be supported

The HTA identifies a number of places where multiple tasks must be done in sequence. The ADS does not explicitly identify sequential relationships, though some of them are captured via the means/ends and causal chain relationships which the ADS does capture. Because the HTA represents trajectories through the set of possible work domain actions, it is possible to represent any kind of sequential constraint which can be identified—but the HTA gives only weak analytic power for identifying those sequential constraints in the first place.

The sequential relationships in the HTA can be divided into two types: sequential constraints imposed by the work domain itself (e.g., you must have water in a reservoir before you can get flow out of the reservoir, you should have downstream flow enabled before turning on a pump to avoid damaging the pump) and those imposed by the nature of human cognition (e.g., it is necessary to have a plan before you can execute that plan, it is necessary to determine what an expected or normal value is before you can detect deviation from it). The discipline required to produce an ADS, and the level of 'deep knowledge' it requires the analyst to acquire, facilitates the identification of the first type of sequential constraints, though these are difficult to represent in an ADS model. This later difficulty could, perhaps, be overcome by the inclusion of explicit safety and equipment health goals as a part of the functional purpose level of the ADS. The second type of sequential constraints are not a part of the work domain per se, and thus are not captured by an ADS.

Both types of sequential relationships are useful for interface design for two reasons. First, if tasks must be done in sequence, then the operator will, in principle, only attend to one at a time. While this principle can be taken to ridiculous extremes, sequential relationships may provide opportunities to suppress information not relevant to a current task or step in order to facilitate greater concentration on the task at hand. Second, when tasks should be done in sequence, interfaces should be designed to support or, in extreme cases, to enforce that sequence.

Suggestions for incorporating sequential relationship knowledge into the DURESS II interface are provided in section 8.3 and 8.4 below.

7.4 Importance of parallelism

The HTA also identifies a number of instances where multiple tasks must be done in parallel. Again, the ADS does not explicitly identify these relationships, although it may suggest them via its identification of means/ends and causal relationships.

Tasks which must be done in parallel are also important guides to interface design since the information requirements for these tasks must be available concurrently to the operator (with associated workload difficulties). The interfaces designed for DURESS II have generally striven to present all information concurrently, therefore they do not suffer from any problems in supporting parallel tasks. This approach will become increasingly problematic with larger task domains and, therefore, the importance of identifying parallel and sequential information needs will take on higher priority.

7.5 Distinction between Display and Control

By discriminating between planning or monitoring tasks and execution tasks, the HTA makes it possible to recognize when the operator needs access to both control capabilities and displayed information relevant to a given variable or piece of equipment versus just the displayed information alone. While this distinction is not always useful for interface design (especially if the transition from a task requiring monitoring alone to one requiring both monitoring and control must happen rapidly and unpredictably), it can sometimes be used to minimize display clutter and focus attention. While the ADS does identify those variables which can be controlled (as well as the means for exerting control over them) versus those which can only be monitored, it does not support the identification of periods during which display alone might be acceptable because it does not explicitly include the notion of sequencing or temporal flow.

Suggestions for filtering information in the DURESS II displays on the basis of the display/control distinction can be found in section 8.4.

7.6 Importance of Social-Organizational Knowledge

The need for the operator to choose between startup and operational strategies or 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. These factors include information about the importance or priority of speed to completion, speed to initiation, consistency of output, and the operator's perceived likelihood of demand changes, faults, excessive workload levels, etc. The operator must either have access to this information (though that doesn't mean that it has to be included in the interface) or s/he will make assumptions about those variables—with potentially erroneous results.

These factors are are not present in the ADS (which can be taken as an argument against building interfaces on the basis of an ADS alone). It should be noted, however, that the ADS technique is envisioned as only the first step in a series of constraint-based analyses (Rasmussen, 1994). Vicente (in press) has labeled this series of analyses 'Cognitive Work Analysis' (CWA) and has formalized their sequence and content as follows: (1) the ADS which focuses on the Work Domain, (2) the Decision Ladder which focuses on the control decisions and actions which must be taken, (3) Information Flow Maps which can be used to analyze viable control strategies, (4) an integration of the other tools used to analyze constraints imposed by the socio-organizational structure, and (5) the Skills, Rules and Knowledge taxonomy which can be used to analyze worker competency requirements. A typical HTA, by contrast, strives to represent every thing an operator must do, and everything s/he must know to do those things, thus it naturally incorporates considerations at all of the levels of a full CWA (which, of course, includes many considerations not included in an ADS analysis). It is important to note, however, in keeping with the map vs. directions analogy developed in section 3.2 above, that the HTA encompasses these considerations only along a specific trajectory and does not capture the full 'space' of constraints and capabilities at each of the CWA levels.

7.7 Tasks/Procedures require assumptions about all levels of CWA

This point is essentially a generalization of 7.6 and 7.8, but it is significant enough to call it out separately. The need to reason about an effective procedure requires information at all the separate levels analyzed in the Cognitive Work Analysis technique (CWA-- see section 7.6 for a description) as well as information

about the available methods available to the human operator for performing control tasks (e.g., the specific interface and automation available, cf. 7.8). For example, note the information about social-organizational priorities described in section 7.6 above, or more extremely, the assumptions made about worker competencies throughout the HTA. While it would certainly be possible to create a series of procedures to enable a preschool child to control DURESS II, those procedures would look somewhat different than what is included in my HTA.

During the performance of an HTA, making these assumptions constrains the set of tasks or procedures which are represented which, in turn, hides work domain capabilities and/or potential means of interacting with DURESS II. These assumed constraints on the possible range of procedures are essential if the analyst is to produce a manageable set of effective procedures but, at the same time, it opens the doors for violations of those assumptions in the usage of the procedures as designed (what if, for some reason, DURESS II *has* to be controlled by a child sometime?)

Finally, in some cases, it is important not only for the analyst, but also for the operator to have access to some of the information from other levels of CWA—the social-organizational priorities in section 7.6 serve as an example. Where there is known variability in this information—that is, where it is known that multiple reasonable assumptions cannot be made—a task analysis will bring in these variables as information requirements. By contrast, due to its concentration on the work domain alone, an ADS analysis by itself would not capture these information requirements. Thus, one virtue of an HTA in conjunction with an ADS is that each broadens the other—the ADS provides 'deep knowledge' about the structure and relationships in the work domain that the HTA will be likely to miss, while the HTA requires integration across multiple layers of considerations and thus provides control task, strategy, social-organizational, worker competency, and even interface-imposed information requirements that the ADS alone would miss. It is probably the case that a *full* CWA would do a still better job at capturing the range of information requirements, but in the absence of this level of commitment, an ADS plus an HTA might be a reasonable compromise.

7.8 Sensitivity to Current Displays = Lack of Device Independence

While both the ADS and the HTA require certain assumptions about the device being analyzed, it would appear that the HTA requires more extensive assumptions than the ADS. The ADS must assume the existence of a certain method of performing feedwater delivery and, even, the existence of a specific configuration of pumps, valves and heaters. In this sense, the only 'device' ADS is sensitive to is the physical plant itself. It makes no assumptions about control equipment, interfaces, etc. The HTA, at least as performed here, must make the same assumptions about the physical plant as well as specifics about the interface available to perform the tasks being examined. For example, in our analysis, deciding which Reservoir Strategy to use (Plan 1.1.4.2 and 1.1.4.3) is critically dependent on whether or not a specific kind of interface is available, and similarly, the execution of balancing reservoir input flows to outputs (Plans 1.6 and 1.7.1 and 1.7.2) would all be performed differently if a graphical presentation of mass in to mass out were not available.

While it might be possible to create a more device-independent HTA, it would certainly be more difficult and would likely be of less value. It would be more difficult because the ability of operators and analysts to reason about how to accomplish given goals is facilitated by the ability to remember or envision oneself in a work environment. It would be made less valuable because, in the absence of specific interactions with human-level interface devices, the notion of 'task' must be abstracted and generalized to a level which is likely to have less overall utility for the specific questions of interest in interface design.

Generally speaking, 'device independence' is more useful during the early stages of design or redesign, when fewer device-relevant decisions have been made. It is also more useful to the degree that major changes in current work domain or operational practice are being contemplated. Thus as a gross generalization, it seems fair that HTA is most useful when minor improvements to current interface design

and operational practice are intended, whereas more substantial modifications will be better served by an ADS analysis or, better yet, an ADS followed by an HTA.

7.9 Lack of Physical Form information

A glaring absence in the display requirements generated from the HTA compared with those from the ADS is the inclusion of physical form, appearance and location information. With the possible inclusion of this information in the Root Cause Diagnosis Branch of the Fault Management task (which was not expanded in depth), no need for physical form information was discovered in the HTA. It would be quite possible for a designer to create an interface of dials, gauges and numerical readouts from the requirements generated by the HTA which had no resemblance whatsoever to the layout or appearance of the values measured in the 'real world' (though a good designer would tend to avoid this).

There are several potential explanations for this finding. First, it may be an artifact of the fact that the work domain upon which actors are acting is, in fact, a simulation. Thus, in reality, the physical appearance and behavior of the interface *is* the physical form with which operators are interacting. Thus, the need for considering, monitoring or interacting with a separate physical reality simply isn't present in the tasks associated with operating DURESS II, and reasoning about that reality may be inefficient or even misleading. Second, the focus in our HTA was explicitly on the DURESS II operator—equivalent to a Board Operator in refinery operations. While it helps a board operator to have knowledge about the physical reality of equipment, this knowledge is far more important for the field operator who must locate, monitor, and manipulate the actual equipment in the field. Another possible explanation is the lack of expansion of the Root Cause Diagnosis branch of Fault Management as mentioned above. While plausible, this explanation is disheartening since, as we have already argued elsewhere (ref**) the enumeration of root causes (and thus, the enumeration of root cause diagnosis task trajectories) is impossibly large. If we need to fully expand this branch to capture all the information required for an interface, then HTA is clearly not adequate on its own.

A final explanation seems the most significant, that the lack of physical form requirements is another manifestation of the lack of 'deep knowledge' obtained via HTA relative to that from ADS. The trajectory-based, directions-like aspect of procedures captured and represented via HTA effectively eliminate the need for 'deep knowledge', including knowledge about the physical form and location of equipment—as long as the contextual assumptions under which the 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 following the instructions in the HTA (as long as initial assumptions hold true), I don't need to know anything more about the system.

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.

7.10 Lack of relationship propagation knowledge

Perhaps the most serious lack noted in the display requirements generated by the HTA is the absence of requirements about the propagation of effects from one equipment variable or state to another. That is, the HTA showed little need to include the relationships identified and represented as equations in the ADS analysis.

Again, the primary reason for this stems from the philosophy and approach taken in the HTA. The intention is to produce (or describe) effective procedures or rule-like plans for accomplishing specific goals. Thus, the designer must reason about the propagation relationships 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 effectiveness of procedures (and interfaces built to support them) comes from. Of course, again, 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 various work domain variables (something which the interface could and should support), then that new procedure may very well be critically flawed.

The one potential exception to this general claim is task 1.2.3.2, where the user develops expectations about the values s/he anticipates seeing when the system is started up. This branch was not expanded in detail in my analysis. In fact, users could accomplish this task in a variety of ways: a superior interface might make it possible for them to simply perceive the relationships derived from various settings chosen, thereby making the task skill based. A more traditional procedural approach would involve determining heuristics or rules which would tell users what values to expect in various circumstances. Finally, users could derive expectations from first principles knowledge about the working of the system and the relationships between its components. To expand this branch in detail would require assumptions about which course users would take, and that would, in turn require assumptions about the training and interfaces they had available. It should be noted, though, that the the work domain knowledge required to solve the problem in a knowledge-based approach is precisely the knowledge which ADS captures—and attempts to present visually to enable at least partial skill-based approaches. By contrast, the more procedural approaches are more readily supported by the HTA, though these collapse (or compile) out that 'deeper' knowledge as discussed in 7.12 below.

7.11 Simplifications for procedure's sake

We note also the tendency for the analyst to create procedural simplifications that help to ensure that the user of the procedures is 'on track'—that is, that s/he is entering the procedure from an expected state to which the procedure applies, rather than from any of the possible system states. The "Abort" task (1.5 and 4.2.1.1.3) is an example of such a simplification in two ways. First, the rule defining its applicability is a simplification. The 'Abort' task's parent plan says that an Abort should be performed if results of a startup are 'not acceptable'—which I have notionally defined as more than 20% off of expectations. At best, this is a gross simplification and a conservative one since many situations would permit larger deviations and still be recoverable.

Second, and more importantly, the 'Abort' task itself operates to place the system back into a configuration from which the written procedures are applicable. The activities in this task (turning off pumps and heaters, draining reservoirs, resetting valves, etc.) are not, strictly speaking, necessary—at least not in all contexts. For example, the plan for task 1.5 states that if the level in either reservoir is more than 50%, then the reservoir should be drained to less than 50% capacity. It is entirely possible to do repeated start up attempts with more than 50% capacity in the reservoirs, but the risk of exceeding the reservoir capacity rises rapidly. Also, the time required to affect the temperature of the water in the reservoirs increases as more water is stored there. Similarly, start up can be attempted with pumps and heaters on, and even with valves in configurations other than shut. In fact, in some cases, starting from a near-desired configuration might actually be quicker than resetting all systems and starting 'from scratch'.

The problem, however, is that writing procedures for all these possibilities becomes exponentially difficult. Instead of tackling that problem, the procedure writer is tempted to enforce conservative 'good practice' rules such as the first example above, or to build 'parking configurations' into the procedures which get the work domain into a state where a more simplified procedure can be applied to it. While this simplification reduces workload for the analyst/designer and, more significantly, for the user who will attempt to follow the procedures, it obscures work domain capabilities which could, if used properly, lead to better context-adaptive performance.

7.12 Implicitness of rationale for procedural knowledge/Lack of "Deep Knowledge"

While the HTA is obviously better than the ADS at capturing and representing procedural knowledge, it is important to note that this benefit comes at the cost of losing some of the 'deep knowledge' required to understand the rationale for those procedures. For example, Plan 1.1.3 doesn't explain *why* you shouldn't choose the Single FWS strategy if the sum of the demands is greater than 10, it simply says that you should not. To understand why this is the case 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 individual feedwater system is only 10 units, thus output greater than that cannot be sustained.

This might imply that a task-based approach makes a poor foundation for training and, while there is some truth to that claim, the reality is more complex. In fact, a procedural, task-based training approach will probably enable a novice operator to conduct useful work more quickly than can be accomplished by learning the deep, structural and functional knowledge about the system. As Vicente has noted (Vicente, submitted) however, this operator will be lost when the situation deviates from that anticipated in the procedures (either because the procedures are in error or incomplete) while the deeply trained operator will have the knowledge required to, perhaps, invent a new procedure on the fly in reaction to a novel situation.

7.13 Limitation of operational behaviors (Hiding work domain capabilities)

Related to the above point about "deep knowledge", the HTA clearly imposes some limits on the set of possible behaviors available to the operator. This is, in part, how it achieves the efficiency (or 'speed to productive work') described above. There are various reasons for this. First, it is difficult to represent all the possible combinations of actions and machine states in separate task trajectories, even in as comparatively limited a domain as DURESS II. Second, by preparing task trajectories, the analyst is acting as a filter on those trajectories: bad trajectories should be screened out for efficiency's sake. But what exactly is a "bad" trajectory? Certainly, ones which fail to accomplish the goal are bad, but what about those which are generally inefficient or unsafe? If the HTA is to be prescriptive at all (and, it must be if it is to make decisions about what to facilitate and what to inhibit via interface design), then shouldn't it filter those trajectories out too? Even so, these inefficient or unsafe trajectories might be the appropriate or the only available ones in some contexts.

In the HTA I conducted, there are a few examples of my decisions to limit the set of available operational behaviors. One occurs at 1.1.3 ("Select Input Strategy") one optional strategy described in Vicente and Pawlack (1994) is the "Valve Complexity Reduction" strategy—opening the initial feed valves (VA and VB) fully and performing all control by limiting this flow via the secondary valves (VA1, VA2, VB1, VB2). This is generally a good strategy. It pays off by reducing the number of settings the operator has to worry about and provides more flexibility (at lower workload) later during the "Execute Stabilize Procedure" task and in the event of changing demands during Normal Operations. It's only known cost is temporary excess flow in the comparatively unlikely event of a secondary valve failing open. Thus, I have made what I believe to be a typical analyst/designer's decision to "build in" the Valve Complexity Reduction strategy into the overall set of tasks to be followed to achieve startup. (Steps 3 and 4 of Plan 1.2.1 "Interpret Strategy Choices" implements the strategy by requiring VA and VB to be set fully open). I have thereby obscured the possibility that one could achieve startup without doing these steps.

7.14 Difficulty of being comprehensive using HTA

Since HTA is best at capturing and representing task trajectories through the possible behavioral states of the work domain, it becomes increasingly unwieldy the more one tries to represent the full set of possible task- and work-domain state situations in it. While aggregation helps this problem somewhat, it only temporarily staves off a combinatorical explosion.

It is far easier for operators and analysts to report 'the normal case' or 'what I usually do'—and this is how HTA has been generally used. As an example, I had to continually remind myself about the possibility that the operator might be using the Continuous Flow, Variable Volume input strategy since this is an uncommon and complex strategy in the trials I have been exposed to—though it has significant implications for what steps should be taken for start up, shut down and normal operations.

The fact that familiar task trajectories are easy to report, unfamiliar ones are difficult to remember, and the overall scope of possible trajectories has three implications for analysis. First, it stresses the importance, and the difficulty, of maintaining comprehensiveness in analysis. ADS is a good approach to this since it captures functional capabilities and constraints of the work domain without trying to articulate all possible trajectories. Second, it stresses the ease of capturing familiar procedures—and the ease with which naïve workers understand procedures. This suggests both that we miss an important opportunity to facilitate learning and operations if we don't make use of known, familiar trajectories, and that we are very likely to be incomplete if we only rely on those trajectories. Finally, it also shows the advantages of doing a task analysis after doing an ADS analysis: 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 they are captured.

7.15 Leap to Information Requirements

It appears that an HTA carried out to the depth of the one presented here, and for the purpose of interface design (one performed for training might well produce different results) is most useful for generating requirements about how to organize information (spatially and temporally) for presentation. The HTA seems less useful (or at least less systematic) than an ADS for actually *identifying* the information to required for the tasks. I will attempt to illustrate this subtle point by an example.

The ADS provides a principled approach to identifying a series of variables and equations which describe the capabilities and constraints inherent in the work domain and then makes the claim, supported by empirical evidence (Vicente, 1996), that an operator needs to know these variable values and equation-based relationships if s/he is to understand and control the domain. Thus, the ADS identifies specific information requirements. An HTA can provide this level of concreteness about information requirements as well and does so most frequently when it attempts to describe tasks in terms of the cognitive operations required to perform them. In our HTA, for example, Task 1.1.2.1.1 describes a cognitive operation "Sum the Demands" which *requires*, as inputs, the two demand values D1 and D2. 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, especially for the purpose of interface design (again, a focus on training may produce different results) 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. I refer to this as making the "leap" to information requirements. By making this leap, the designer/analyst is making two assumptions: (1) that s/he has the *right* set of information requirements, and (2) that the operator will know how to combine them in order to perform the task.

It may be argued that this is simply the result of laziness on the part of analysts using HTA. While there might be truth to that charge, it is worth investigating why this 'laziness' may be prevalent. My belief is that, at least in industrial settings, the reasons revolve around the fact that the 'deeper' one drives the HTA, the bigger the branching logic becomes. Working through this combinatorial explosion becomes tedious and far too costly (at a point in the design cycle where organizations are unused to paying large sums for human factors analyses). Thus, while an HTA driven to the level of fine-grained cognitive procedures (as in 1.1.2.1.1) is entirely possible, it is generally far too tempting to make the 'leap to information requirements' illustrated above.

In this sense, then, another potential advantage of ADS is that it provides a more complete, specific and principled set of information requirements for less effort than it would take an HTA to provide the same

requirements. Though, as we have seen above, even an HTA performed at a coarser granularity can provide some types of information that an ADS will not provide (primarily sequence and organizational requirements for displays).

8. Implications for DURESS II Interface

In section 3.1 above, we presented our rationale for comparing the two analytic methods on the basis of the display requirements they produce rather than on the basis of displays generated using them. We noted, however, that the ultimate proof would have to come from user performance in interaction with displays produced using the alternate techniques. In recognition of the fact that an analysis has little utility if it does not produce concrete recommendations for display design, we have included this section of display recommendations for modifying the current DURESS II interfaces to meet some of the novel task-based display requirements produced in the HTA. The current DURESS II interfaces have been produced using ADS analyses alone and are described in detail in Bisantz and Vicente (1994). Note that the suggestions below are at the beginning of the "Creative Design Process" illustrated in Figure 1 above. They are intended to illustrate display concepts the designer might investigate and seek to integrate into a comprehensive display design.

8.1 Strategy Guides

Intended to support the importance of method selection described in 7.1.

The Importance of strategy selection as a determiner of both expectations and, more generally, how the user intends to interact with DURESS II, suggests that perhaps strategies should be made into first order, user manipulable and/or inspectable objects in the interface itself. One way of doing this might be to use pull down menus for selection combined with display graphics showing constraint regions (e.g., selecting the Single FWS strategy from the Feedwater strategy menu causes a regional boundary overlay to appear on the secondary valves and/or the reservoirs showing that this strategy can only provide 10 water units total over the two reservoirs). A more sophisticated (but also more intrusive) decision aid could analyze the demand goals and provide strategy recommendations, or (more like Guerlain's critiquing approach, 1995) could offer feedback on the user's strategy choices. Finally, the user could command DURESS II indirectly via strategy selection itself rather than (or in addition to) commanding it directly via component settings? Since the selection of a set of strategies does not uniquely or completely specify operational behavior, a smart planning and/or control system would need to make decisions among the remaining free variables left open by the user's strategy selections. This is akin to my 'tasking interface' approach for unmanned air vehicles as described in (Miller & Goldman, 1997).

8.2 Expectation Indicators

Intended to support the importance of expected values described in 7.2

The importance of user expectations identified in the task analysis indicates that an explicit presentation of task- or strategy-driven expectation indicators could be advantageous through facilitating user comparison of present values to expected ones. In their simplest form, these could be user-placed bugs or, in conjunction with increasingly sophisticated automation, expected trend lines, deviation alarms, etc. Note that expectation guides should change with task being performed and this might impose added workload and potential for error if the user has to update bugs frequently. On the other hand, if a more automated system were used, mismatch between the system's notion of the task being performed and that of the user could be very problematic.

8.3 Sequencing Information

Intended to address 7.3.

As noted in 7.3, it's more important for an interface to address and communicate work-domain imposed sequential constraints than cognitive ones. Either sort could be supported via a left to right (or top to bottom) orientation of components mirroring the task sequence flow (though this might violate physical form information). Alternatively, sequentially lighted backdrops, frames or unitizers (e.g., during start up, place a shaded unitizer around the pumps first, then once one or more of them is started, the unitizer moves to the corresponding primary valves, etc.). More intrusive still (but maybe appropriate for safety reasons) would be machine-mediated lockouts—the system will not let the operator start a heater until there is water in the reservoir. Again, note that these sequencing displays must change with task and strategy, with the corresponding potential for either workload and/or intent mismatch problems described above.

8.4 Information Suppression/Filtering

Addressses 7.5, 7.3 and general organization and prioritization of information requirements.

The sequencing of tasks in the HTA shows places where the information needs for some tasks will *never* overlap with those of another tasks (because the two tasks cannot be active concurrently). In these instances, and assuming that transition between tasks will not be overly rapid or unpredictable, it may make sense to suppress or filter out the information for non-active tasks. For example, input flow valves and heater controls (and perhaps displays) could be suppressed when doing Normal Operations (especially if not using CFVV strategy). Similarly, control capabilities for all components could be suppressed during 1.1 and 1.2—though a planning 'sketch pad' showing constraint propagation effects, recommendations and/or critiques might be nice. Physical form information might only be needed for Root Cause Diagnosis, but see 7.9 above.

8.5 Information Prioritization

The frequency, duration and importance of tasks in the HTA provide clues as to the relative priority of the information needed for those tasks. Using this approach, we make the following recommendations:

Temperature and flow demands and measures are repeatedly useful throughout the most important DURESS II tasks. Thus, they could be highlighted even more than they currently are given their salience for operations. Use brighter colors, enlarge the regions, set them apart from display clutter more, etc.

Deviations from expected values are also very important. If they are included, whatever method used should be highly salient.

8.6 Dynamic organization

Similar to dynamic prioritization or filtering of information on the basis of the active tasks and their information needs, so also the organization of elements in the system itself may be of interest in different ways given the strategies in use or the tasks being performed. For example, pumps, pipes and valves could be dynamically organized to show which are in use or not and which are routed where—that is, show the dynamic task- or strategy-based part-whole aggregation of the FWS, not necessarily the static part-whole composition determined in the ADS analysis. Many tasks must be done with reference to a different collection of components depending on this organization. Interface methods of accomplishing this might range from use of color for flow indication and destination, background unitizers to dynamically aggregate components or, with potentially serious disruption of both Physical form and Physical Function info, dynamic rearrangement of interface elements.

8.7 Additional Information

This is intentionally a catch-all category for information needs identified in the HTA but not in the ADS (for example, in section 7.6 and elsewhere). Such additional information might include:

- During the planning stages (of startup or in response to a demand change) the sum of demands should be available. Also, mass input capacity of current configuration, and perhaps mass input may also be helpful.
- Information pertinent to the satisfaction of social and organizational goals and priorities should be provided in addition to the functional purpose goals of output volume and temperature. There is currently no information provided about the relative need for or performance against such dimensions as speed to startup, consistency of output, error rate, etc., nor is there information provided about likelihood of faults or changes in demand goals. Not all of these dimensions will be important, but some of them are and they should be included. Interestingly, a timer is included in the DURESS II interfaces, and this does provide limited feedback on one such organizational goal: duration of the run and/or time at goal state. There is, however, no justification for this inclusion from the ADS analysis.

9. Implications for a Unified Modeling Approach

In some ways, the work reported in this document represents the first practical steps towards a unified task- and work-domain based analysis approach for interface design. We have shown that each type of analysis provides unique and complimentary knowledge about how an interface to support work in the domain should be designed, and we have identified at least some ways in which the modeling appraoches should be linked or integrated.

Simply doing as I have done here—performing both types of analyses in sequence—represents one way in which the two alternate methodologies can be unified. Although this may not be the most efficient method of performing an integrated analysis, nor are its results as 'unified' as we might like, there are still things we can learn from this work about the shape which a unified approach might take.

Doing the ADS first provided the analyst with a firm grounding in system capabilities and constraints—more thorough and better organized than is frequently the case when doing a task analysis alone. Precisely because the ADS is better at capturing the 'deep knowledge' about a system, doing it first will give the integrated model a firmer footing in the reality of system behavior. This argues that whenever the design task *requires* a deep grounding in system capabilities (perhaps because it involves the creation or modification of an interface, automation, procedures, etc; for a physical system that is well understood and/or will remain unchanged), it will be valuable to begin with an understanding of the plant as provided by an ADS analysis.

Would there be advantages to performing the task analysis first? While I did not take this approach, I can draw some inferences about the type of knowledge that might be gleaned. Because, as discussed in section 7.12, procedural task models inherently compile and therefore hide general or 'deep' knowledge about the constraints and capabilities of the system, I would expect the analyst using a task-based approach alone to develop a better sense of how the operator currently behaves, but comparatively impoverished knowledge about how or why those behaviors are effective. Doing the task analysis first, might provide the analyst with 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), s/he 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.

The approach we took in this study illustrates that it is possible to create a 'task model' from the ADS itself, creating and representing possible methods and strategies analytically derived from the ADS without the aid of users input. Such an approach is useful for a variety of reasons:

1. It illustrates strategies base on the full set of work domain capabilities and constraints, not simply those that users have been able to discover. The set of such strategies can then be used to evaluate observations and operator communications about the set of strategies actually used. By this method, operator misunderstandings and inconsistencies might be identified and novel strategies might be discovered.
2. The act of rethinking the work domain information obtained and formulated by the ADS into a procedural hierarchy forces the analyst to take a user-centered perspective which at a minimum identifies sequencing and prioritization information about the work. In addition, as we saw, it may force the integration of additional knowledge about the work context—such as social/organizational goals and priorities, worker competencies, available equipment, etc.
3. The generation of the work-domain based task model puts the analyst in a position to evaluate user reported task information against the model. The synergy between these two sources of knowledge could help to identify weaknesses in the WDA, weaknesses in user training programs, or simply opportunities for improvements to the work domain which do not require re-engineering of the physical system itself (e.g., when operators fail to use a possible strategy because it is too difficult for them to implement, we have identified an opportunity where an improved interface or added automation might make this strategy feasible).

The knowledge provided by the two analytic approaches seems to connect at a number of points:

- Functional goals of the system—what is the user attempting to accomplish? This knowledge takes a prominent role in each analytic tool and, in fact, is decomposed in sub-goals and sub-sub goals similarly in each technique.
- Methods or strategies for achieving those goals: the ADS provides information to support the generation of possible methods while the task analysis spells out procedures (including user actions) for accomplishing those strategies.
- System behavioral constraints often form the basis for procedures (especially, the plan steps within an HTA task), though the procedures may not make their rationale clear.

These points of connection should provide the basis for a unified analytic approach and/or representation.

Finally, what can this project tell us about how a unified analytic modeling approach will impact interface design. We have shown that the two different analytic approaches each produce different, but complimentary information about the information users will need to operate within the work domain, and about how they will need that information. Furthermore, section 8 provides hints that it will be possible to integrate the types of display requirements produced by each approach in a single interface. There are, however, a wide variety of ways in which this could be done. It is probably not the case that operators need a task-based interface *and* a separate, work domain-based one; rather a single interface design should incorporate the display requirements generated by each approach. More work needs to be done, however, to understand exactly how this can be accomplished most effectively in various domains and, ultimately, to ascertain whether such an integrated interface design will produce better human-machine performance across the range of situations it encounters than an interface designed from either perspective alone.

10. References

Bisantz, A. and Vicente, K. (1994). Making the abstraction hierarchy concrete. International Journal of human-computer studies 40:83-117.

Diaper, D. (1989). Task analysis for human-computer interaction. Ellis Horwood; Chichester, UK.

Funk, H. & Miller, C. (1997). Task sensitive versus 'context sensitive': What should an adaptive interface adapt to? In Proceedings of the International and Interdisciplinary Conference on Modeling and Using Context. Rio de Janeiro; February 4-6.

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

Guerlain, S. (1995). Using the critiquing approach to cope with brittle expert systems. In Proceedings of the Human Factors and Ergonomics Society 39th Annual Meeting (pp. 233-237). Santa Monica, CA: HFES.

Hunter, C., Janzen, M., and Vicente, K. (1995). Research on factors influencing human cognitive behavior (II). Technical Report CEL-95-08. Cognitive Engineering Laboratory, University of Toronto. September.

Kirwan, B. and Ainsworth, L. (1992). A Guide to Task Analysis. Taylor and Francis; Bristol, PA.

Miller, C., Funk, H., and Hannen, M. (1997). "Task-Based Interface Management; A Rotorcraft Pilot's Associate Example." In Proceedings of the American Helicopter Society's Crew Systems Technical Specialists Meeting. Philadelphia, PA; September 23-25.

Miller, C. and Goldman, R. (1997). 'Tasking Interfaces': Associate systems that know who's the boss. In Proceedings of the 4th International Conference on Human-Electronic Crewmembers. Krueth, Germany, Sept 24-29.

Miller, C. and Vicente, K. (1998). Toward and integration of task and work domain analysis techniques for human-computer interface design. To appear in Proceedings of the 1998 Meeting of the Human Factors and Ergonomic Society, October 5-8; Chicago, IL.

Miller, C. and Vicente, K. (1998). "Integrated Abstraction Hierarchy and Plan-Goal Graph Model for the DURESS II System; A Test Case for Unified System- and Task-based Modeling and Interface Design," v. 1.00, unpublished technical report, March 18, 1998.

Rasmussen, J. (1974). The human data processor as a system component: Bits and pieces of a model (Risø-M-1722). Roskilde, Denmark: Danish Atomic Energy Commission.

Rasmussen, J. (1976). Outlines of a hybrid model of the process plant operator. In T. B. Sheridan and G. Johannsen (Eds.), Monitoring behavior and supervisory control (pp. 371-383). New York: Plenum.

Rasmussen, J. (1980). The human as a systems component. In H. T. Smith & T. R. G. Green (Eds.), Human interaction with computers (pp. 67-96). London: Academic Press.

Rasmussen, J. (1981). Models of mental strategies in process plant diagnosis. In J. Rasmussen & W. B. Rouse (Eds.), Human detection and diagnosis of system failures (pp. 241-258). New York: Plenum.

Rassmussen, J. (1985). The role of hierarchical knowledge representation in decisoin making and system management. IEEE Transactions on Systems, Man and Cybernetics, 15, pp. 234-243.

Rasmussen, J., Pejtersen, A., and Goodstein, L. (1994). Cognitive Systems Engineering. John Wiley & Sons; New York.

Rouse, W., Geddes, N. and Curry, R. (1988). An Architecture for intelligent interfaces: Outline of an approach to supporting operators of complex systems. Human-Computer Interaction, 3. pp. 87-122.

Shepherd, A. (1989). Analysis and training in information technology tasks. In Diaper, D. (Ed.). Task analysis for Human-computer Interaction. Ellis Horwood, Ltd.; Chichester. Pp. 15-55.

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

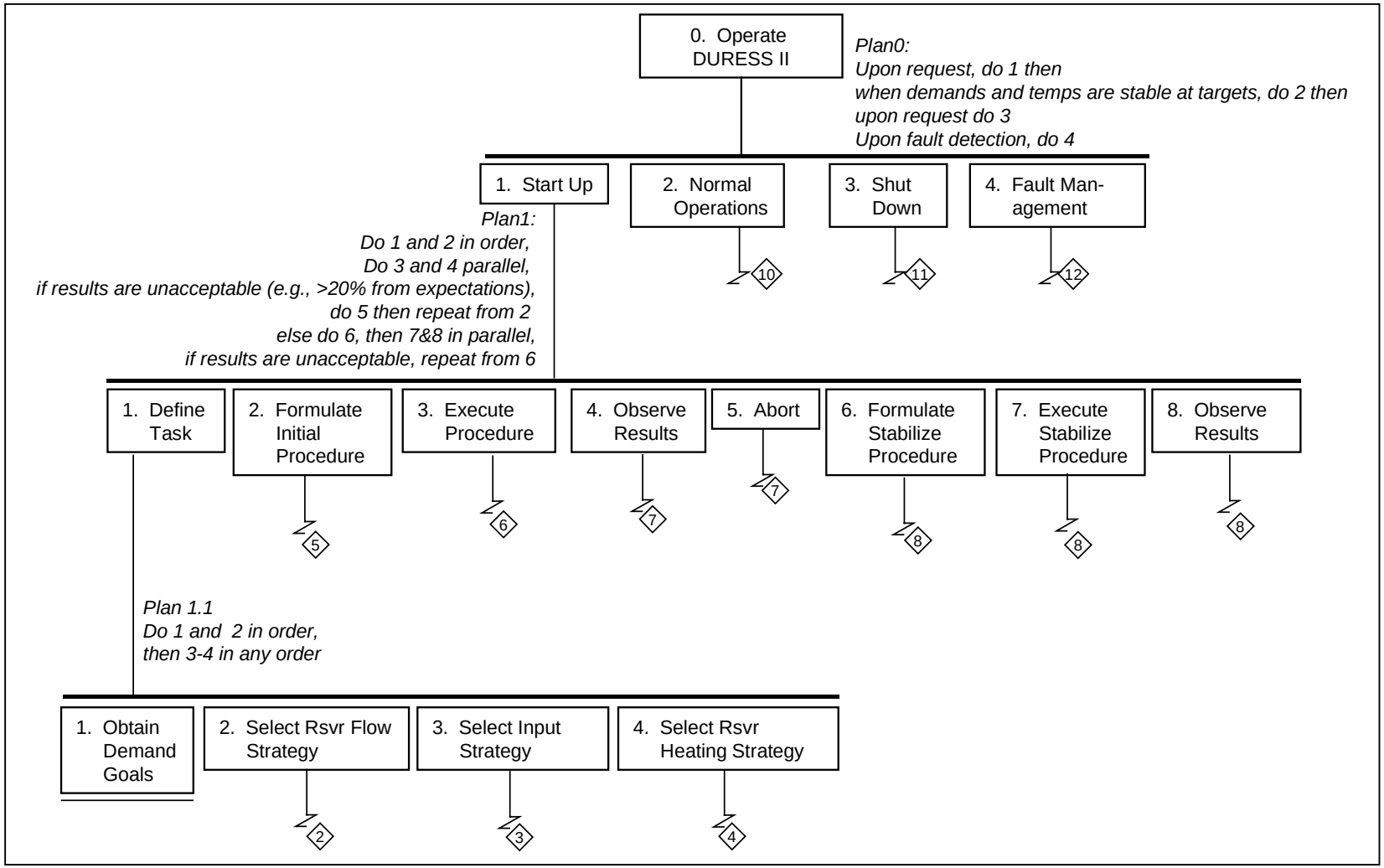
Vicente, K. J. (in press). Cognitive work analysis: Towards safe, productive, and healthy computer-based work. Erlbaum: Mahwah, NJ.

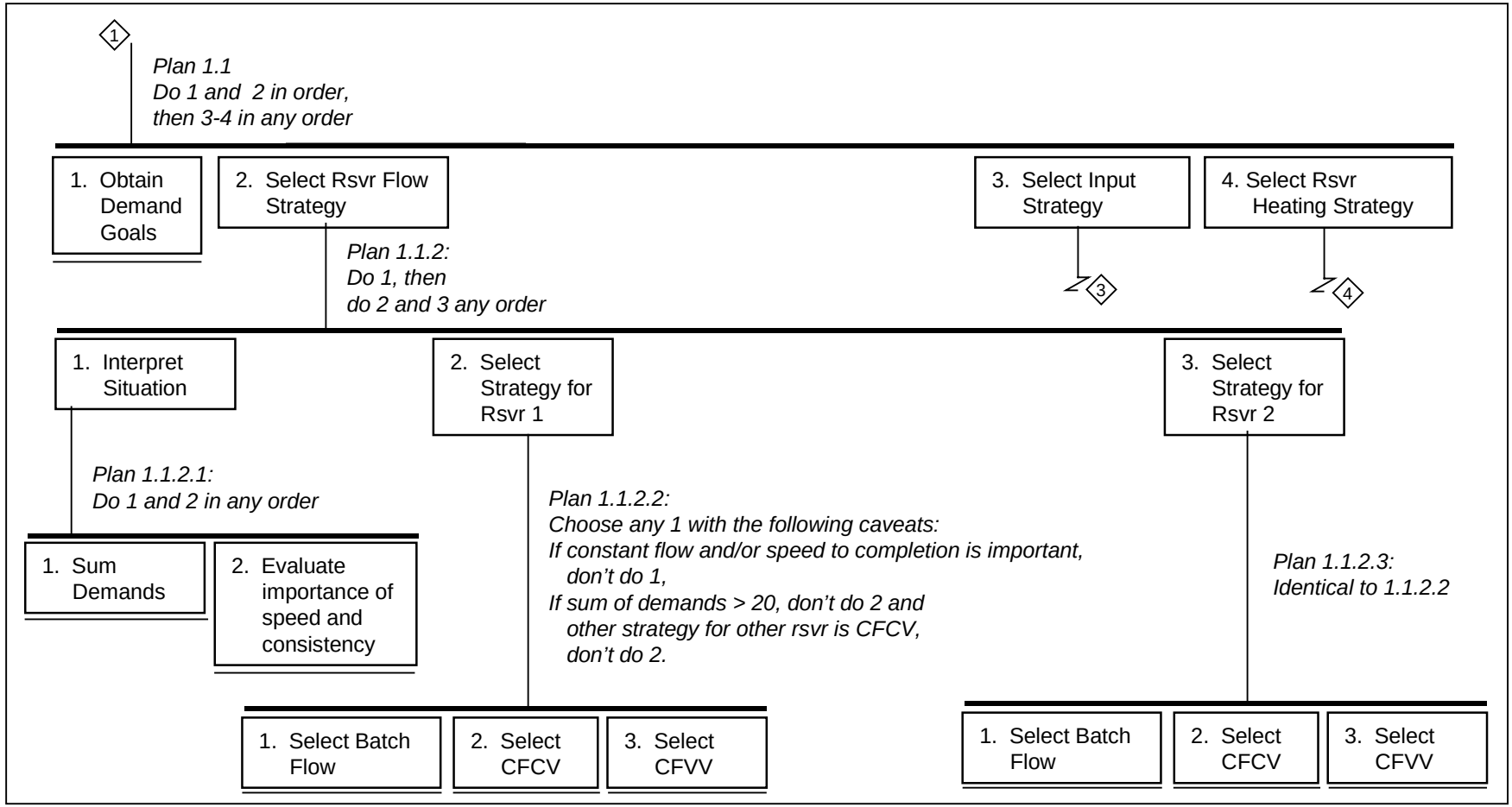
Vicente, K. Wanted: Psychologically relevant, device- and event-independent work analysis techniques. Submitted for publication in Interacting with Computers.

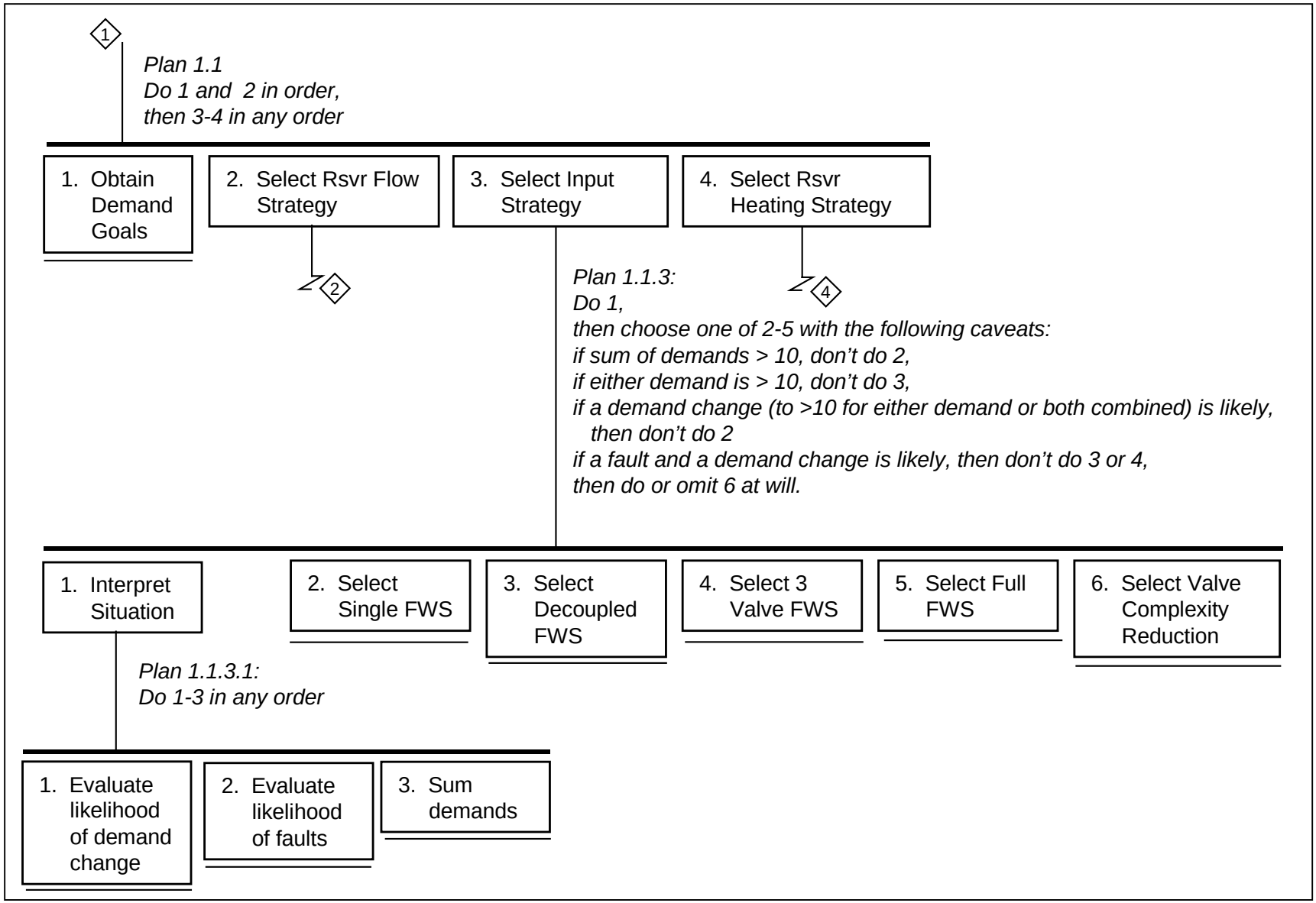
Vicente, K. J., & Rasmussen, J. (1992). Ecological interface design: Theoretical foundations. IEEE Transactions on Systems, Man, and Cybernetics, SMC-22, 589-606.

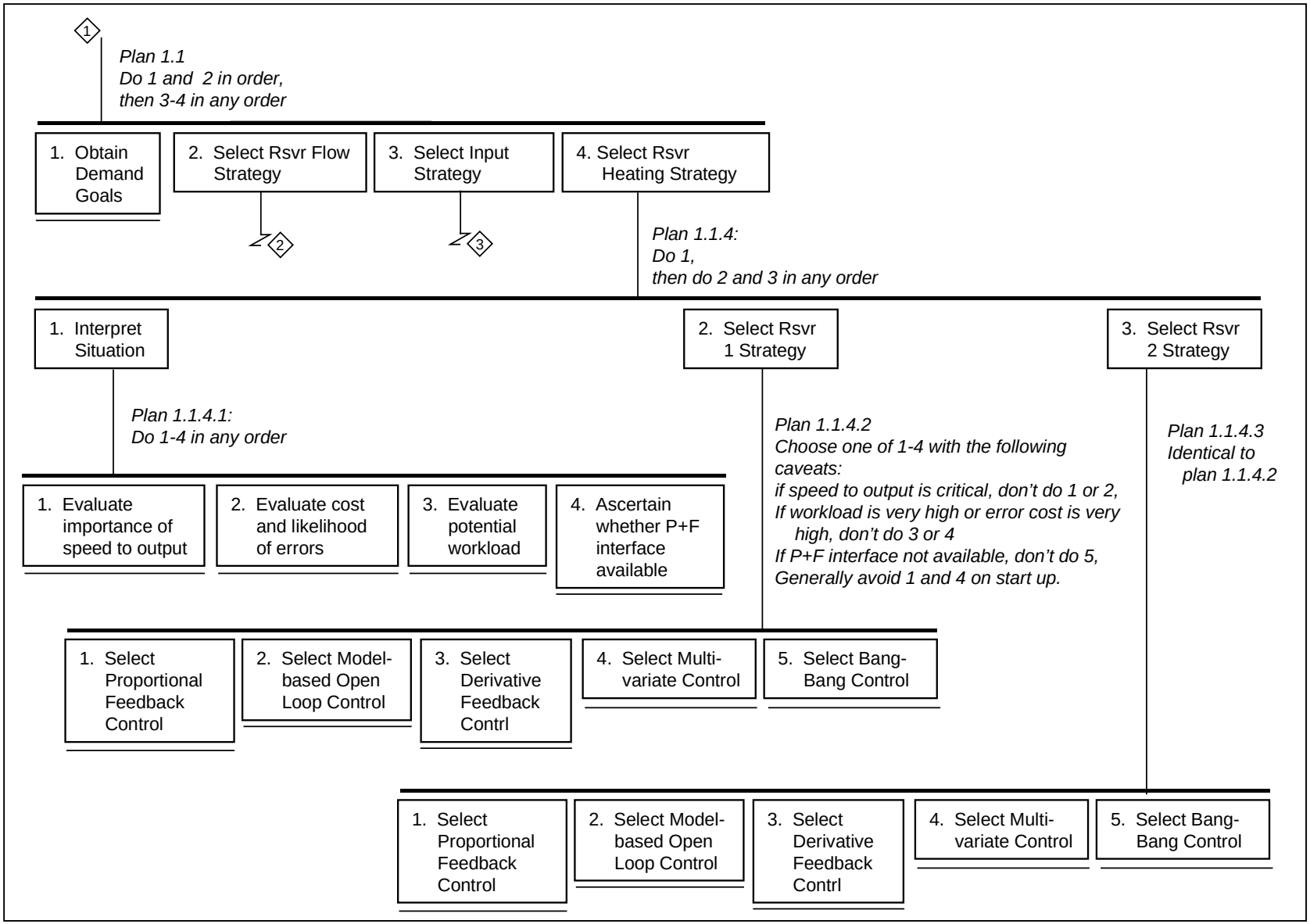
Vicente, K. and Rasmussen, J. (1990). The ecology of human-machine systems II: Mediating 'direct perception' in complex work domains. Ecological Psychology, 2(3). Pp. 207-249.

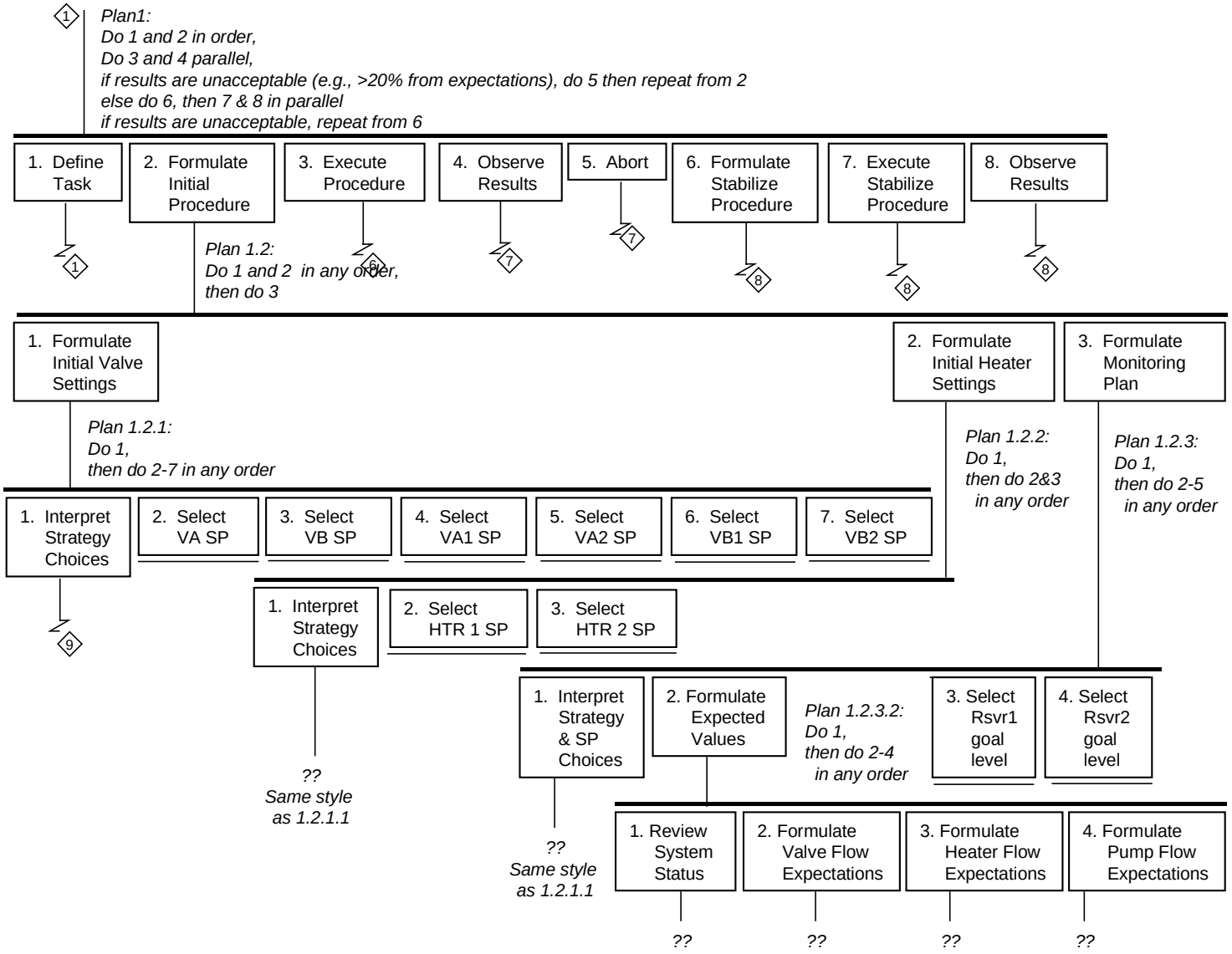
11. Appendix A

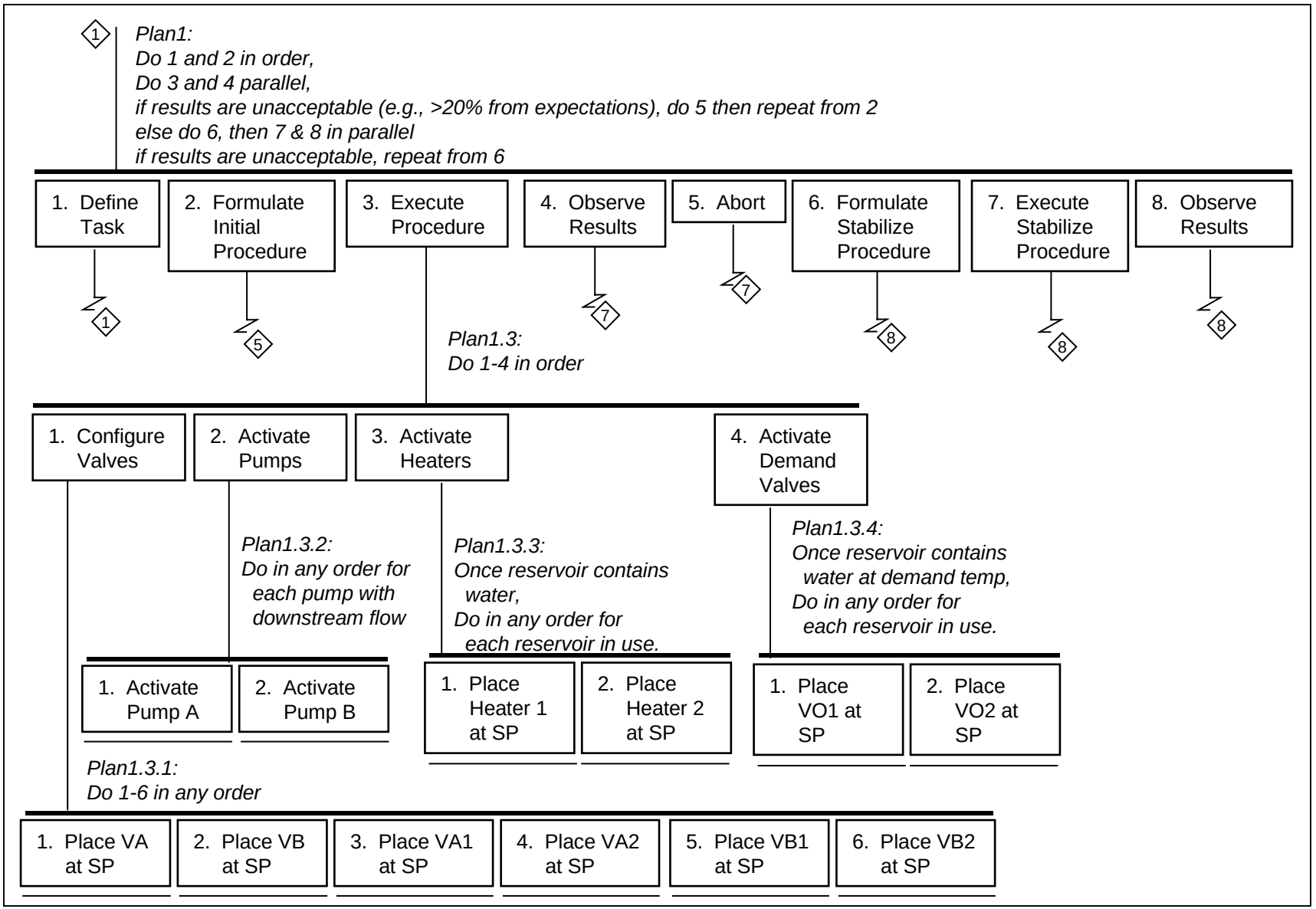


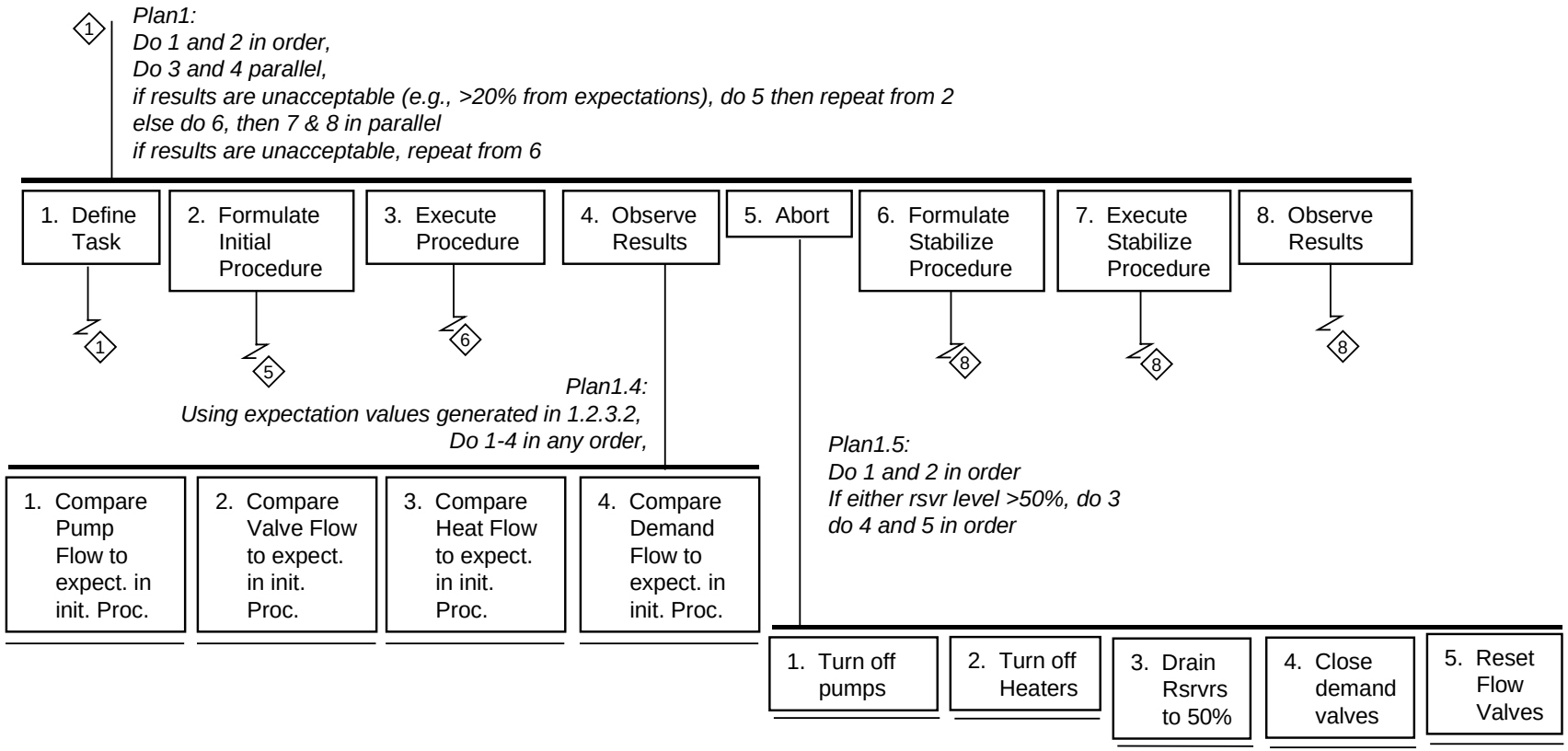

1
Plan 1.1
Do 1 and 2 in order,
then 3-4 in any order
1. Obtain Demand Goals
2. Select Rsvr Flow Strategy
3. Select Input Strategy
4. Select Rsvr Heating Strategy
Plan 1.1.2:
Do 1, then
do 2 and 3 any order
3
4
1. Interpret Situation
2. Select Strategy for Rsvr 1
3. Select Strategy for Rsvr 2
Plan 1.1.2.1:
Do 1 and 2 in any order
1. Sum Demands
2. Evaluate importance of speed and consistency
Plan 1.1.2.2:
Choose any 1 with the following caveats:
If constant flow and/or speed to completion is important,
don't do 1,
If sum of demands > 20, don't do 2 and
other strategy for other rsvr is CFCV,
don't do 2.
Plan 1.1.2.3:
Identical to 1.1.2.2
1. Select Batch Flow
2. Select CFCV
3. Select CFVV
1. Select Batch Flow
2. Select CFCV
3. Select CFVV

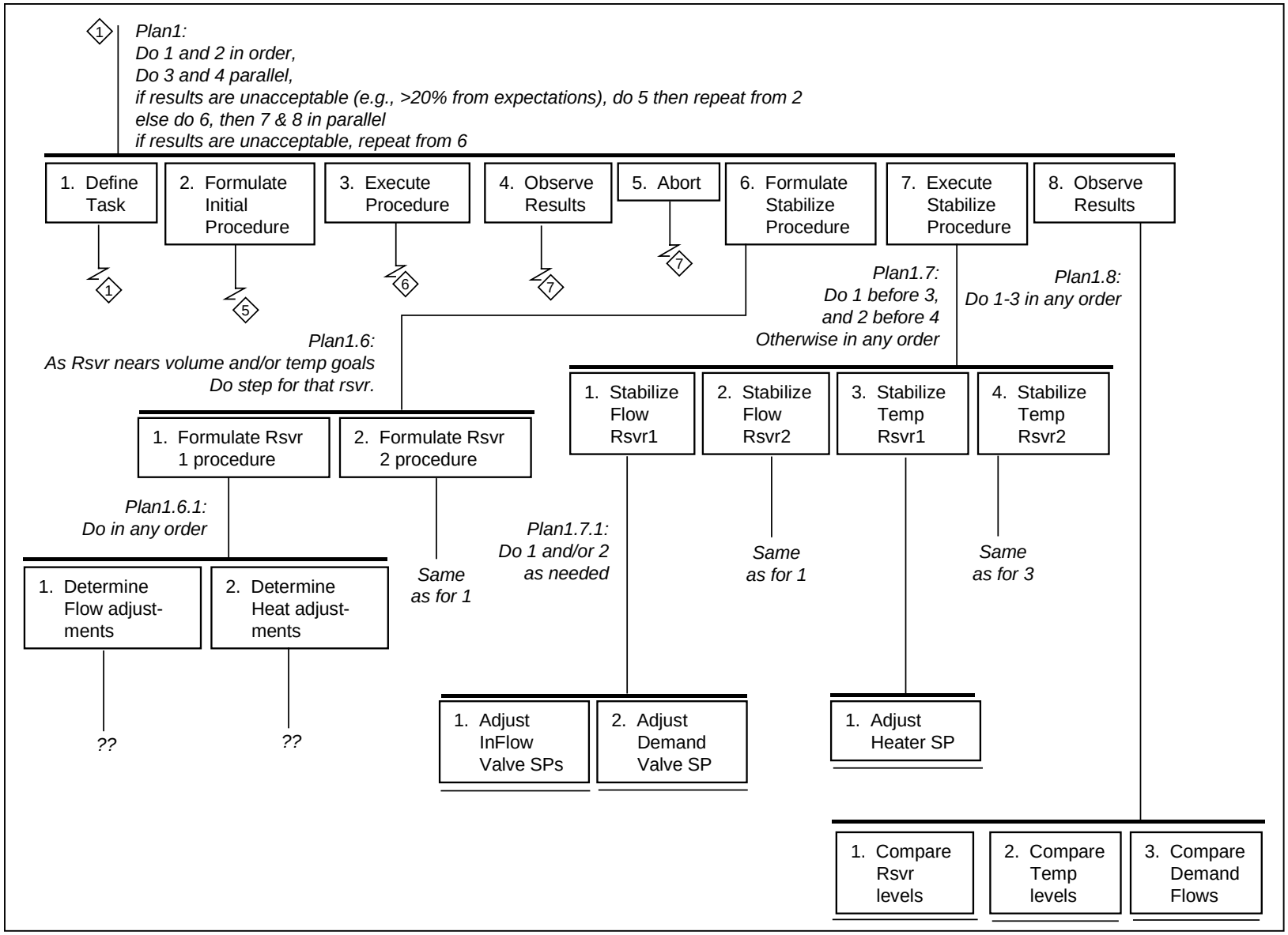

1
Plan 1.1
Do 1 and 2 in order,
then 3-4 in any order
1. Obtain Demand Goals
2. Select Rsvr Flow Strategy
3. Select Input Strategy
4. Select Rsvr Heating Strategy
2
4
Plan 1.1.3:
Do 1,
then choose one of 2-5 with the following caveats:
if sum of demands > 10, don't do 2,
if either demand is > 10, don't do 3,
if a demand change (to >10 for either demand or both combined) is likely, then don't do 2
if a fault and a demand change is likely, then don't do 3 or 4,
then do or omit 6 at will.
1. Interpret Situation
2. Select Single FWS
3. Select Decoupled FWS
4. Select 3 Valve FWS
5. Select Full FWS
6. Select Valve Complexity Reduction
Plan 1.1.3.1:
Do 1-3 in any order
1. Evaluate likelihood of demand change
2. Evaluate likelihood of faults
3. Sum demands

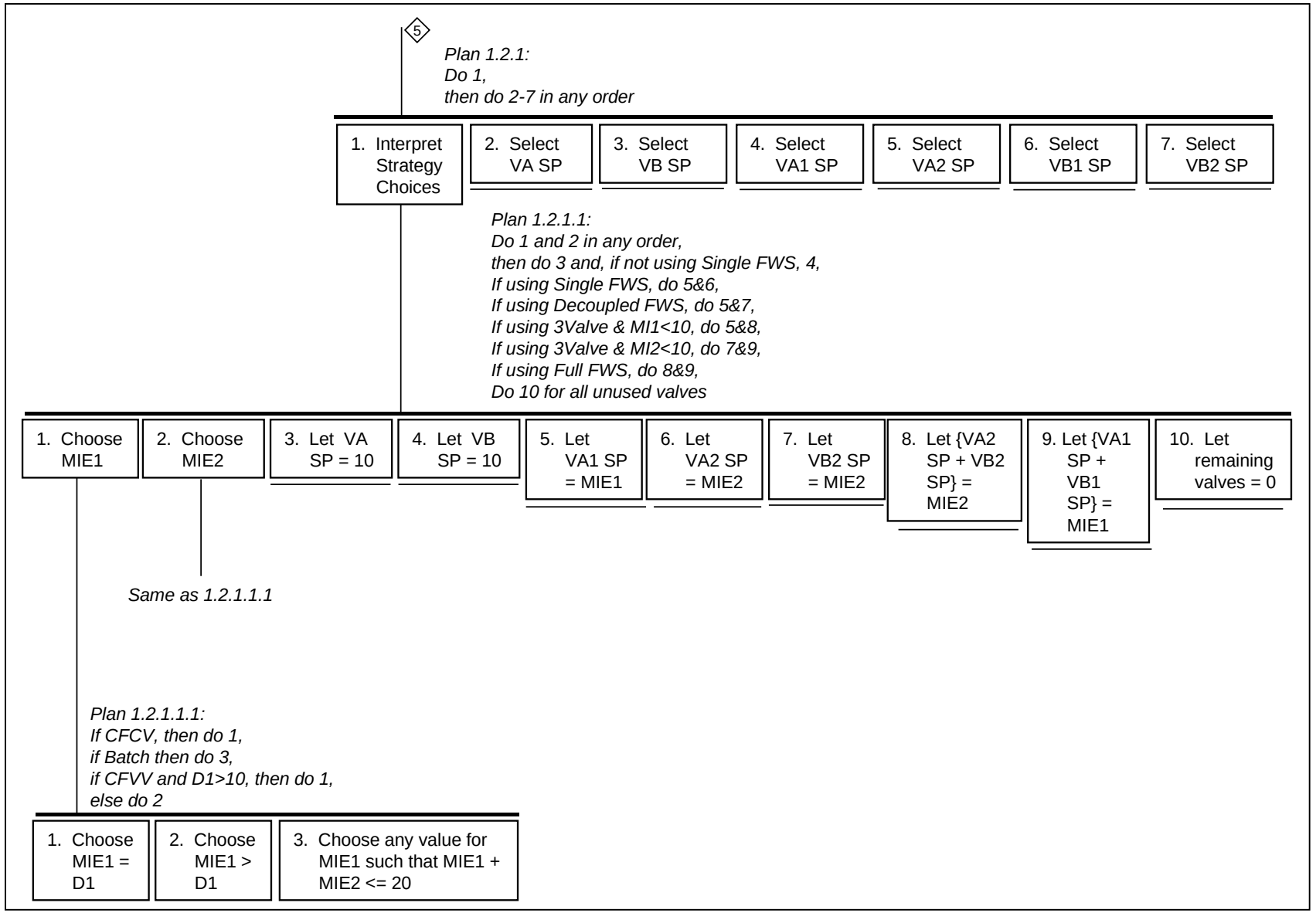

1
Plan 1.1
Do 1 and 2 in order,
then 3-4 in any order
1. Obtain Demand Goals
2. Select Rsvr Flow Strategy
3. Select Input Strategy
4. Select Rsvr Heating Strategy
2
3
Plan 1.1.4:
Do 1,
then do 2 and 3 in any order
1. Interpret Situation
2. Select Rsvr 1 Strategy
3. Select Rsvr 2 Strategy
Plan 1.1.4.1:
Do 1-4 in any order
Plan 1.1.4.2
Choose one of 1-4 with the following caveats:
if speed to output is critical, don't do 1 or 2,
If workload is very high or error cost is very high, don't do 3 or 4
If P+F interface not available, don't do 5,
Generally avoid 1 and 4 on start up.
Plan 1.1.4.3
Identical to plan 1.1.4.2
1. Evaluate importance of speed to output
2. Evaluate cost and likelihood of errors
3. Evaluate potential workload
4. Ascertain whether P+F interface available
1. Select Proportional Feedback Control
2. Select Model-based Open Loop Control
3. Select Derivative Feedback Contrl
4. Select Multi-variate Control
5. Select Bang-Bang Control
1. Select Proportional Feedback Control
2. Select Model-based Open Loop Control
3. Select Derivative Feedback Contrl
4. Select Multi-variate Control
5. Select Bang-Bang Control

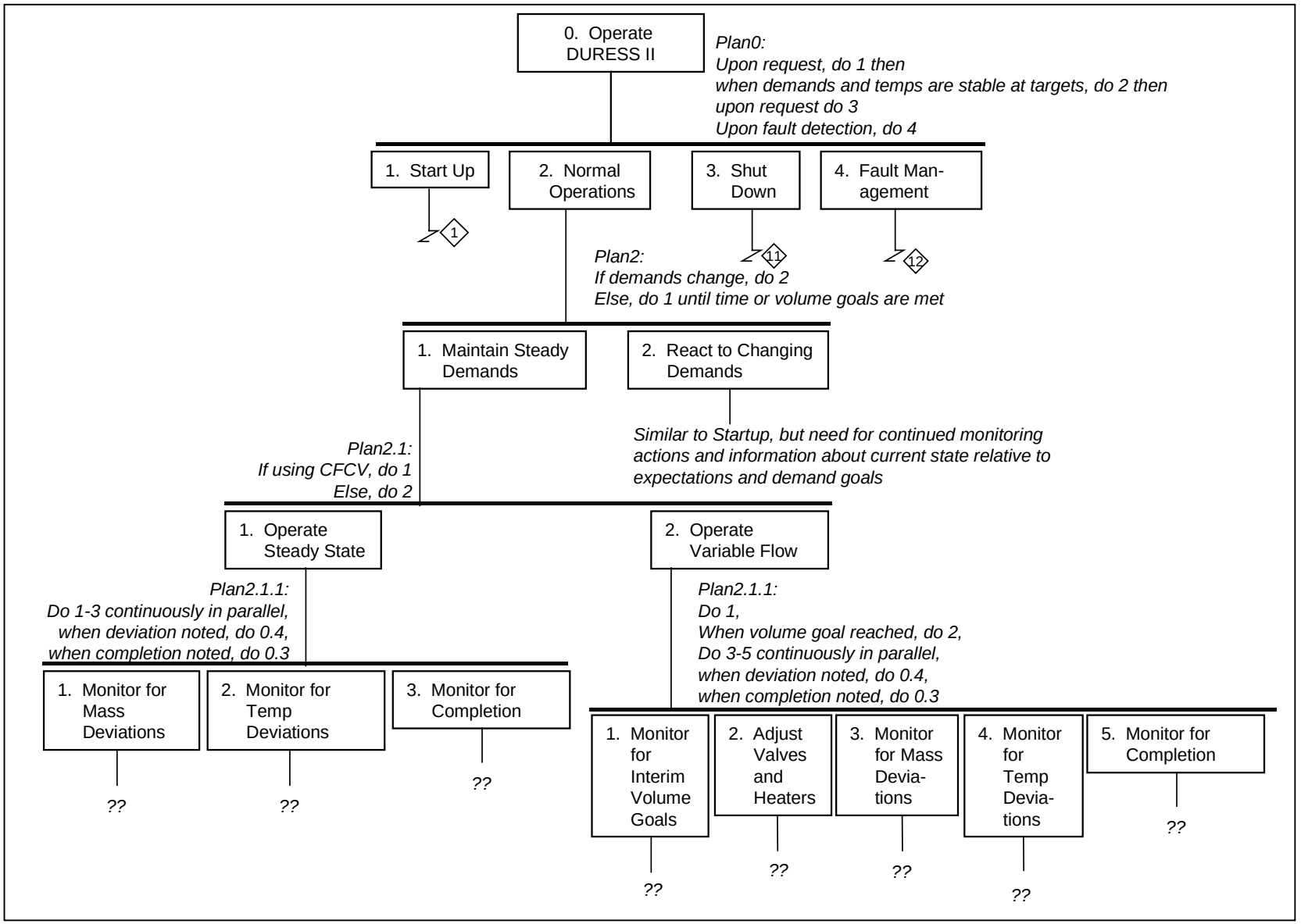

1
Plan1:
Do 1 and 2 in order,
Do 3 and 4 parallel,
if results are unacceptable (e.g., >20% from expectations), do 5 then repeat from 2
else do 6, then 7 & 8 in parallel
if results are unacceptable, repeat from 6
1. Define Task
2. Formulate Initial Procedure
3. Execute Procedure
4. Observe Results
5. Abort
6. Formulate Stabilize Procedure
7. Execute Stabilize Procedure
8. Observe Results
1
6
7
7
8
8
8
Plan 1.2:
Do 1 and 2 in any order,
then do 3
1. Formulate Initial Valve Settings
2. Formulate Initial Heater Settings
3. Formulate Monitoring Plan
Plan 1.2.1:
Do 1,
then do 2-7 in any order
Plan 1.2.2:
Do 1,
then do 2&3
in any order
Plan 1.2.3:
Do 1,
then do 2-5
in any order
1. Interpret Strategy Choices
2. Select VA SP
3. Select VB SP
4. Select VA1 SP
5. Select VA2 SP
6. Select VB1 SP
7. Select VB2 SP
9
1. Interpret Strategy Choices
2. Select HTR 1 SP
3. Select HTR 2 SP
?? Same style as 1.2.1.1
1. Interpret Strategy & SP Choices
2. Formulate Expected Values
Plan 1.2.3.2:
Do 1,
then do 2-4
in any order
3. Select Rsvr1 goal level
4. Select Rsvr2 goal level
?? Same style as 1.2.1.1
1. Review System Status
2. Formulate Valve Flow Expectations
3. Formulate Heater Flow Expectations
4. Formulate Pump Flow Expectations
??
??
??
??

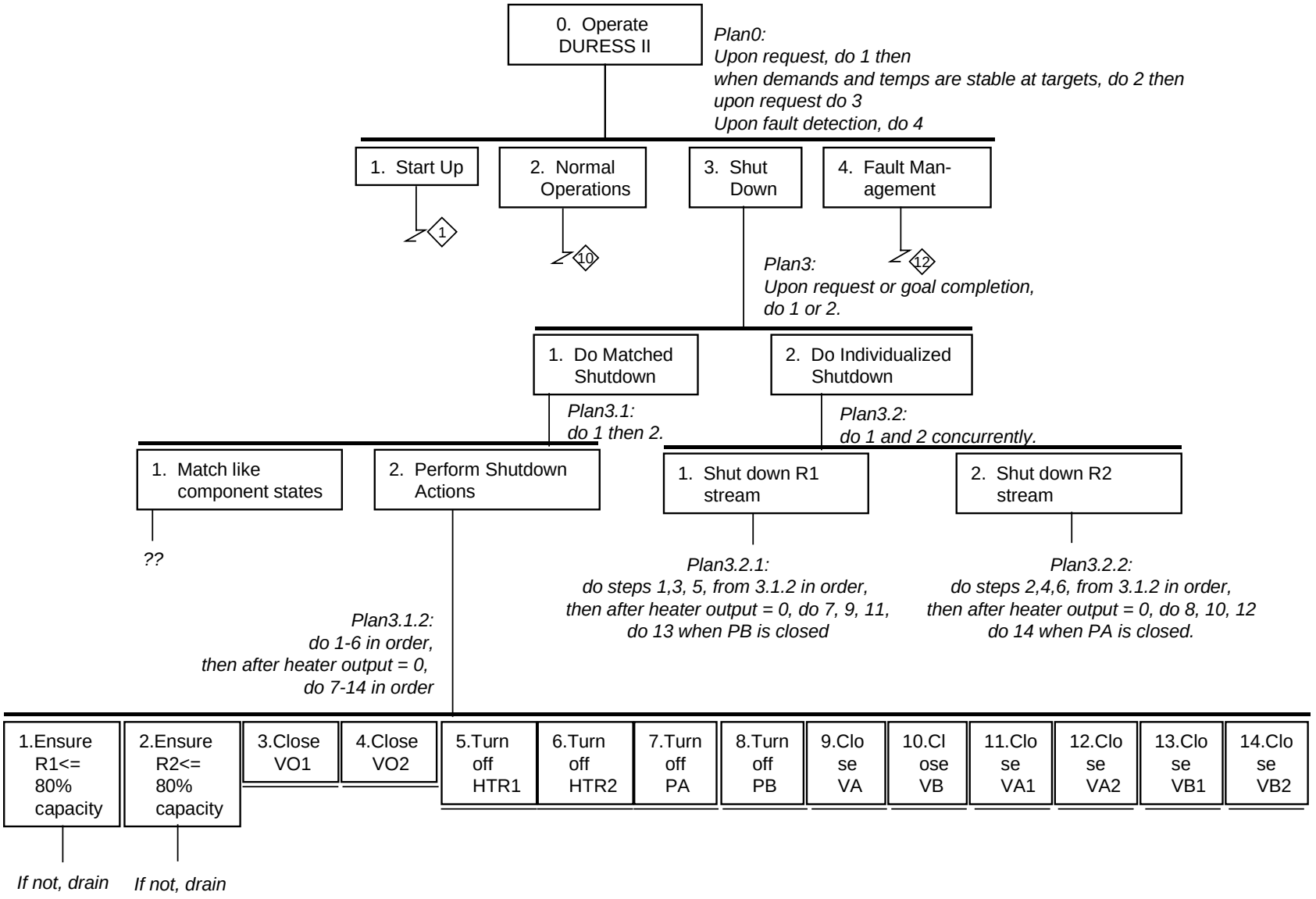

1
Plan1:
Do 1 and 2 in order,
Do 3 and 4 parallel,
if results are unacceptable (e.g., >20% from expectations), do 5 then repeat from 2
else do 6, then 7 & 8 in parallel
if results are unacceptable, repeat from 6
1. Define Task
2. Formulate Initial Procedure
3. Execute Procedure
4. Observe Results
5. Abort
6. Formulate Stabilize Procedure
7. Execute Stabilize Procedure
8. Observe Results
1
5
7
7
8
8
8
Plan1.3:
Do 1-4 in order
1. Configure Valves
2. Activate Pumps
3. Activate Heaters
4. Activate Demand Valves
Plan1.3.2:
Do in any order for each pump with downstream flow
Plan1.3.3:
Once reservoir contains water,
Do in any order for each reservoir in use.
Plan1.3.4:
Once reservoir contains water at demand temp,
Do in any order for each reservoir in use.
1. Activate Pump A
2. Activate Pump B
1. Place Heater 1 at SP
2. Place Heater 2 at SP
1. Place VO1 at SP
2. Place VO2 at SP
Plan1.3.1:
Do 1-6 in any order
1. Place VA at SP
2. Place VB at SP
3. Place VA1 at SP
4. Place VA2 at SP
5. Place VB1 at SP
6. Place VB2 at SP

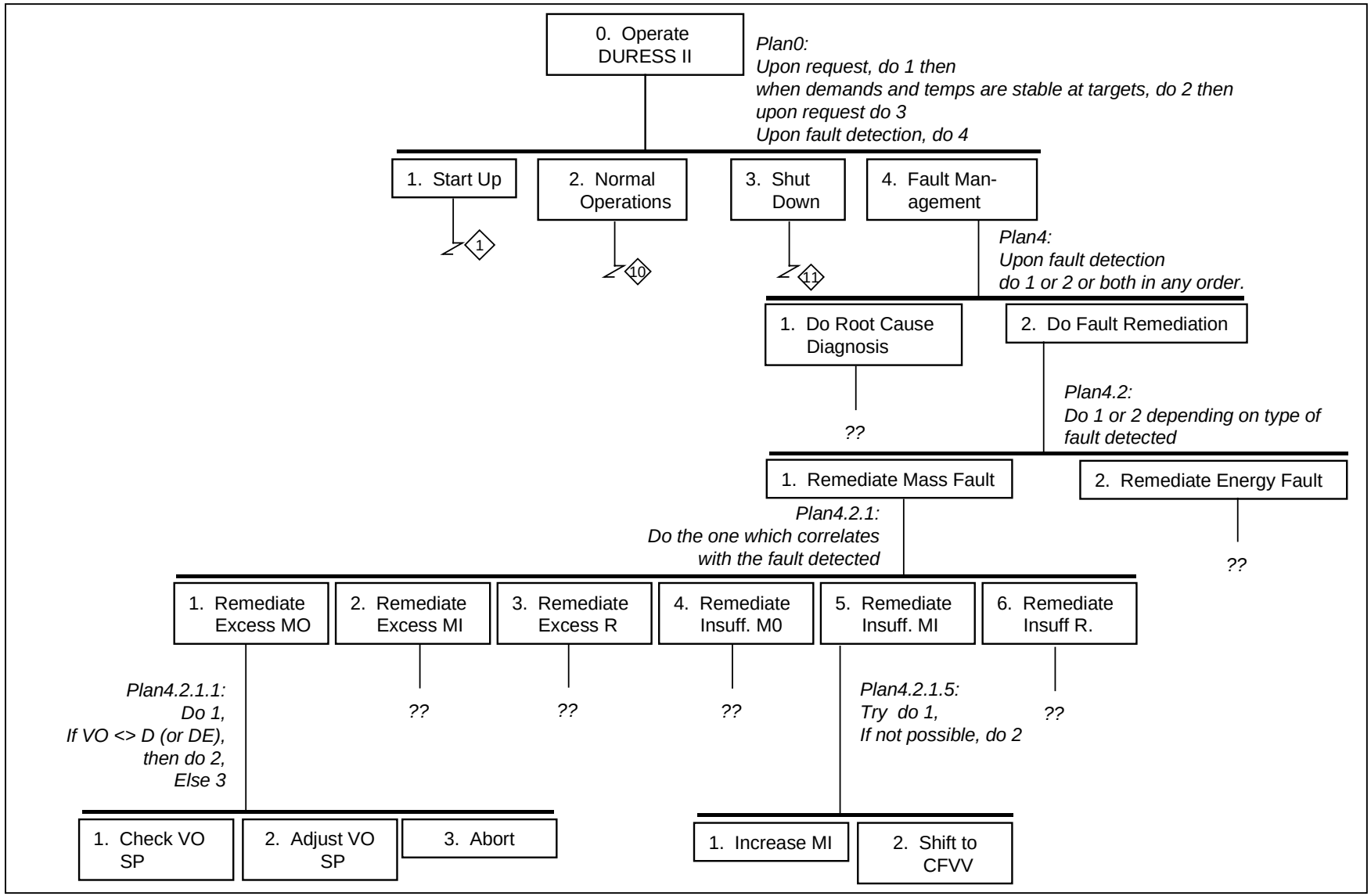

1
Plan1:
Do 1 and 2 in order,
Do 3 and 4 parallel,
if results are unacceptable (e.g., >20% from expectations), do 5 then repeat from 2
else do 6, then 7 & 8 in parallel
if results are unacceptable, repeat from 6
1. Define Task
2. Formulate Initial Procedure
3. Execute Procedure
4. Observe Results
5. Abort
6. Formulate Stabilize Procedure
7. Execute Stabilize Procedure
8. Observe Results
1
5
6
8
8
8
Plan1.4:
Using expectation values generated in 1.2.3.2,
Do 1-4 in any order,
1. Compare Pump Flow to expect. in init. Proc.
2. Compare Valve Flow to expect. in init. Proc.
3. Compare Heat Flow to expect. in init. Proc.
4. Compare Demand Flow to expect. in init. Proc.
Plan1.5:
Do 1 and 2 in order
If either rsvr level >50%, do 3
do 4 and 5 in order
1. Turn off pumps
2. Turn off Heaters
3. Drain Rsrvrs to 50%
4. Close demand valves
5. Reset Flow Valves


1
Plan1:
Do 1 and 2 in order,
Do 3 and 4 parallel,
if results are unacceptable (e.g., >20% from expectations), do 5 then repeat from 2
else do 6, then 7 & 8 in parallel
if results are unacceptable, repeat from 6
1. Define Task
2. Formulate Initial Procedure
3. Execute Procedure
4. Observe Results
5. Abort
6. Formulate Stabilize Procedure
7. Execute Stabilize Procedure
8. Observe Results
1
5
6
7
7
Plan1.6:
As Rsvr nears volume and/or temp goals
Do step for that rsvr.
Plan1.7:
Do 1 before 3,
and 2 before 4
Otherwise in any order
Plan1.8:
Do 1-3 in any order
1. Formulate Rsvr 1 procedure
2. Formulate Rsvr 2 procedure
1. Stabilize Flow Rsvr1
2. Stabilize Flow Rsvr2
3. Stabilize Temp Rsvr1
4. Stabilize Temp Rsvr2
Plan1.6.1:
Do in any order
Same as for 1
Plan1.7.1:
Do 1 and/or 2
as needed
Same as for 1
Same as for 3
1. Determine Flow adjust-ments
2. Determine Heat adjust-ments
1. Adjust InFlow Valve SPs
2. Adjust Demand Valve SP
1. Adjust Heater SP
??
??
1. Compare Rsvr levels
2. Compare Temp levels
3. Compare Demand Flows


5
Plan 1.2.1:
Do 1,
then do 2-7 in any order
1. Interpret Strategy Choices
2. Select VA SP
3. Select VB SP
4. Select VA1 SP
5. Select VA2 SP
6. Select VB1 SP
7. Select VB2 SP
Plan 1.2.1.1:
Do 1 and 2 in any order,
then do 3 and, if not using Single FWS, 4,
If using Single FWS, do 5&6,
If using Decoupled FWS, do 5&7,
If using 3Valve & MI1<10, do 5&8,
If using 3Valve & MI2<10, do 7&9,
If using Full FWS, do 8&9,
Do 10 for all unused valves
1. Choose MIE1
2. Choose MIE2
3. Let VA SP = 10
4. Let VB SP = 10
5. Let VA1 SP = MIE1
6. Let VA2 SP = MIE2
7. Let VB2 SP = MIE2
8. Let {VA2 SP + VB2 SP} = MIE2
9. Let {VA1 SP + VB1 SP} = MIE1
10. Let remaining valves = 0
Same as 1.2.1.1.1
Plan 1.2.1.1.1:
If CFCV, then do 1,
if Batch then do 3,
if CFVV and D1>10, then do 1,
else do 2
1. Choose MIE1 = D1
2. Choose MIE1 > D1
3. Choose any value for MIE1 such that MIE1 + MIE2 <= 20


0. Operate DURESS II
Plan0:
Upon request, do 1 then
when demands and temps are stable at targets, do 2 then
upon request do 3
Upon fault detection, do 4
1. Start Up
2. Normal Operations
3. Shut Down
4. Fault Management
1
11
12
Plan2:
If demands change, do 2
Else, do 1 until time or volume goals are met
1. Maintain Steady Demands
2. React to Changing Demands
Similar to Startup, but need for continued monitoring actions and information about current state relative to expectations and demand goals
Plan2.1:
If using CFCV, do 1
Else, do 2
1. Operate Steady State
2. Operate Variable Flow
Plan2.1.1:
Do 1-3 continuously in parallel,
when deviation noted, do 0.4,
when completion noted, do 0.3
1. Monitor for Mass Deviations
2. Monitor for Temp Deviations
3. Monitor for Completion
??
??
??
Plan2.1.1:
Do 1,
When volume goal reached, do 2,
Do 3-5 continuously in parallel,
when deviation noted, do 0.4,
when completion noted, do 0.3
1. Monitor for Interim Volume Goals
2. Adjust Valves and Heaters
3. Monitor for Mass Deviations
4. Monitor for Temp Deviations
5. Monitor for Completion
??
??
??
??
??


0. Operate DURESS II
Plan0:
Upon request, do 1 then
when demands and temps are stable at targets, do 2 then
upon request do 3
Upon fault detection, do 4
1. Start Up
2. Normal Operations
3. Shut Down
4. Fault Management
1
10
12
Plan3:
Upon request or goal completion,
do 1 or 2.
1. Do Matched Shutdown
2. Do Individualized Shutdown
Plan3.1:
do 1 then 2.
Plan3.2:
do 1 and 2 concurrently.
1. Match like component states
2. Perform Shutdown Actions
1. Shut down R1 stream
2. Shut down R2 stream
??
Plan3.2.1:
do steps 1,3, 5, from 3.1.2 in order,
then after heater output = 0, do 7, 9, 11,
do 13 when PB is closed
Plan3.2.2:
do steps 2,4,6, from 3.1.2 in order,
then after heater output = 0, do 8, 10, 12
do 14 when PA is closed.
Plan3.1.2:
do 1-6 in order,
then after heater output = 0,
do 7-14 in order
1.Ensure R1<= 80% capacity
2.Ensure R2<= 80% capacity
3.Close VO1
4.Close VO2
5.Turn off HTR1
6.Turn off HTR2
7.Turn off PA
8.Turn off PB
9.Close VA
10.Close VB
11.Close VA1
12.Close VA2
13.Close VB1
14.Close VB2
If not, drain
If not, drain


0. Operate DURESS II
Plan0:
Upon request, do 1 then
when demands and temps are stable at targets, do 2 then
upon request do 3
Upon fault detection, do 4
1. Start Up
2. Normal Operations
3. Shut Down
4. Fault Man-agement
1
10
11
Plan4:
Upon fault detection
do 1 or 2 or both in any order.
1. Do Root Cause Diagnosis
2. Do Fault Remediation
??
Plan4.2:
Do 1 or 2 depending on type of fault detected
1. Remediate Mass Fault
2. Remediate Energy Fault
??
Plan4.2.1:
Do the one which correlates with the fault detected
1. Remediate Excess MO
2. Remediate Excess MI
3. Remediate Excess R
4. Remediate Insuff. M0
5. Remediate Insuff. MI
6. Remediate Insuff R.
Plan4.2.1.1:
Do 1,
If VO <> D (or DE),
then do 2,
Else 3
??
??
??
Plan4.2.1.5:
Try do 1,
If not possible, do 2
??
1. Check VO SP
2. Adjust VO SP
3. Abort
1. Increase MI
2. Shift to CFVV

12. Appendix B

12.1 Abbreviations

Table 3 presents and defines the abbreviations used in the Information Requirements portion of the HTA included in this section:

Table 3. Abbreviations used to describe the information requirements identified in the HTA.

D1	Demand flow from R1; hence this is usually also the expected or goal flow
D2	Demand flow from R2; hence this is usually also the expected or goal flow
EI1	Aggregate energy flowing into R1
EI1E	Expected or goal aggregate energy into R1
EI2	Aggregate energy flowing into R2
EI2E	Expected or goal aggregate energy into R2
EO1	Aggregate energy flowing out of R1
EO1E	Expected or goal aggregate energy out of R1
EO2	Aggregate energy flowing out of R2
EO2E	Expected or goal aggregate energy out of R2
FA	Actual flow through Valve A
FB	Actual flow through Valve B
FA1	Actual flow through Valve A1
FA2	Actual flow through Valve A2
FB1	Actual flow through Valve B1
FB2	Actual flow through Valve B2
FH1	Actual flow of energy from HTR1
FH2	Actual flow of energy from HTR2
FO1	Actual flow through VO1
FO2	Actual flow through VO2
MI1	Aggregate mass flowing into R1
MI1E	Expected or goal aggregate mass into R1
MI2	Aggregate mass flowing into R2
MI2E	Expected or goal aggregate mass into R2
MO1	Aggregate mass flowing out of R1; should equal VO1
MO1E	Expected or goal aggregate mass flowing out of R1
MO2	Aggregate mass flowing out of R2; should equal VO2
MO2E	Expected or goal aggregate mass flowing out of R2
HTR1	Setting of Heater 1
HTR1E	Expected or goal output of heater 1
HTR2	Setting of Heater 2
HTR2E	Expected or goal output of heater 2
PA	Pump A setting
PAE	Expected or goal setting of Pump A
PB	Pump B setting
PBE	Expected or goal setting of Pump B
R1	Level (= mass) of water in reservoir 1
R1E	Expected or goal level of water in reservoir 1
R2	Level (= mass) of water in reservoir 2
R2E	Expected or goal level of water in reservoir 2
T0	Input temperature for water into system and, thus (presumably), into R1 and R2
T1	Demand temperature for output from T1; hence usually the goal temp
T2	Demand temperature for output from T2; hence usually the goal temp
TR1	Actual temperature of R1
TR1E	Expected or goal temperature of water in R1
TR2	Actual temperature of R2

TR2E	Expected or goal temperature of water in R2
VA,	Set point for Valve A
VAE	Expected or goal flow through Valve A
VB,	Set point for Valve B
VBE	Expected or goal flow through Valve B
VA1,	Set point for Valve A1
VA1E	Expected or goal flow through Valve A1
VA2,	Set point for Valve A2
VA2E	Expected or goal flow through Valve A2
VB1,	Set point for Valve B1
VB1E	Expected or goal flow through Valve B1
VB2,	Set point for Valve B2
VB2E	Expected or goal flow through Valve B2
VO1	Setting of Output valve on R1, during normal operations, should be set at D1
VO2	Setting of Output valve on R2, during normal operations, should be set at D2

12.2 HTA in Tabular form

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
0.	Operate DURESS II			
	Plan 0: *Upon request, do 1 then* *When output stabilizes at targets, do 2 then* *Upon request, do 3.* *Upon fault detection, do 4*	DURESS trials frequently were defined as 5 minutes at demands, plus start up and shut down times—hence <10 minutes overall		
	1. Start Up	Once per run, 5-10 min		Implies these IRs could be presented sequentially and not overlapped
	2. Normal Operations	5 min continuous		
	3. Shut Down	2 min		
	4. Fault Management	Occasional/variable	Implies you've always got to be monitoring for faults	Implies you always need quick access to IRs for this task
1.	Start Up			
	Plan 1: *Do 1.1 and 1.2 in order, then do 1.3 and 1.4 in parallel, if results are not acceptable (> 20% of expectations), do 1.5 then repeat from 2, else do 1.6 then 1.7 and 1.8 in parallel. If results are unacceptable, repeat from 1.6*			
	1. Define Task	Nearly intuitive for skilled operators, much slower for unskilled.		
	2. Formulate Initial Procedure	~1-2 min		
	3. Execute procedure	1 min		
	4. Observe results	1-2 min		
	5. Abort	1 min		
	6. Formulate Stabilize Procedure	1 min		
	7. Execute Stabilize Procedure	1 min		
	8. Observe Results	1-2 min		
1.1	Define Task			
	Plan 1.1 *Do 1 and 2 in order,* *then 3-4 in any order*			
	1. Obtain Demand Goals			T1, T2 D1, D2 Time period or Volume goal (either collectively or by output valve)
	2. Select Reservoir flow Strategy			

sequential

parallel

occ

parallel

sequential

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	3. Select Water Input Strategy			
	4. Select Heat Input Strategy			
1.1.2	*Plan 1.1.2:* *Do 1, then* *Do 2 and 3 in any order*			
	1. Interpret situation	sequential xor		D1, D2
	2. Select Rsvr 1 strategy			
	3. Select Rsvr 2 strategy			
1.1.2.1	*Plan 1.1.2.1:* *Do 1 & 2 in any order*			
	1. Sum Demands			{D1+D2}
	2. eval import of speed and consistency		Don't have to be in interface, but do have to be available somewhere	Relative importance of speed and consistency
1.1.2.2	*Plan 1.1.2.2:* *Choose any 1 with the following caveats: If constant flow and/or speed to completion is important, don't do 1, if sum of demands >20, and other rsvr strategy is CFCV, don't do 2*			
	1. Select Batch Flow	xor		∑D, speed & consistency concerns, selected R1 flow strategy
	2. Select CFCV			
	3. Select CFVV			
1.1.2.3	*Plan 1.1.2.3:* *Choose any 1 with the following caveats: If constant flow and/or speed to completion is important, don't do 1, if sum of demands >20, and other rsvr strategy is CFCV, don't do 2*			
	1. Select Batch Flow	xor		∑D, speed & consistency concerns, selected R1 flow strategy
	3. Select CFCV			
	3. Select CFVV			
1.1.3	Select Input Strategy *Plan 1.1.3:* *Do 1, then choose one of 2-5 with the following caveats: if sum of demands > 10, don't do 2, if either demand >10, don't do 3, if a demand change (to >10 for either demand or both combined) is likely, then don't do 2, if fault and demand change likely, then try to avoid 3 or 4*		The valve reduction strategy has been built into the execute task (1.3), thus it will always be done by following that plan	

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	1. Interpret Situation	sequential		
	2. Select single FWS	xor		
	3. Select Decoupled FWS			
	4. Select 3 Valve FWS			
	5. Select Full FWS			
1.1.3.1	Interpret Situation *Plan 1.1.3.1* *Do 1-3 in any order*			
	1. Evaluate likelihood of demand change		Again, these don't have to be in UI	Info about likelihood of demand change
	2. Evaluate likelihood of faults			Info about likelihood of faults
	3. Sum demands			{D1+D2}
1.1.4	Select Rsvr Heating Strategy *Plan 1.1.4:* *Do 1, then do 2 and 3 in any order*			
	1. Interpret situation	sequential		
	2. Select Rsvr 1 Strategy			
	3. Select Rsvr 2 Strategy			
1.1.4.1	Interpret Situation *Plan 1.1.4.1:* *Do 1-4 in any order*			
	1. Evaluate importance of speed to output		Again, these don't have to be in UI	Info about importance of speed to output
	2. Evaluate cost and likelihood of errors			Info about likelihood and costs of errors
	3. Evaluate potential workload			Info about potential workload (and contributing factors: concurrent and/or near future tasks, changes in demand, noise in input and control, etc)
	4. Ascertain whether P+F interface available		Most importantly, is there a graphical mass and energy balance presentation	Interface appearance?
1.1.4.2	Select Rsvr 1 Strategy *Plan 1.1.4.2:* *Choose one of 1-4 with the following caveats: If speed to output is critical, don't do 1 or 2; if workload is very high or error cost is very high, don't do 3 or 4; if P+F interface not available, don't do 5, generally, avoid 1 and 4 on startup.*			
	1. Select Proportional Feedback Control	xor		
	2. Select model-based Open Loop Control			∑social/organizational factors described above and selected heat strategy for R1
	3. Select Derivative			

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	Feedback Control			
	4. Select Multivariate Control			
	5. Select Bang-Bang control			
1.1.4.3	Select Rsvr 2 Strategy *Plan 1.1.4.3:* *Choose one of 1-4 with the following caveats: If speed to output is critical, don't do 1 or 2; if workload is very high or error cost is very high, don't do 3 or 4; if P+F interface not available, don't do 5, generally, avoid 1 and 4 on startup.*			
	1. Select Proportional Feedback Control	xor		∑social/organizational factors described above and selected heat strategy for R2
	2. Select model-based Open Loop Control			
	3. Select Derivative Feedback Control			
	4. Select Multivariate Control			
	5. Select Bang-Bang control			
1.2	Formulate Initial Procedure *Plan 1.2:* *Do 1&2 in any order, then do 3* *1. Formulate Initial Valve Settings* *2. Formulate Heater Settings* 3. *Formulate Monitoring Plan*	sequential	Translating the general strategies selected above into actual settings	
1.2.1	Formulate Initial Valve Settings *Plan 1.2.1:* *Do 1, then do 2-7 in any order* *1. Interpret Strategy Choices* *2. Select VA SP* *3. Select VB SP* *4. Select VA1 SP* *5. Select VA2 SP* *6. Select VB 1 SP* 7. *Select VB2 SP*	sequential	I'm making a somewhat arbitrary distinction here between deciding what you want the valve SPs to be (done in 1.2.1.1.1-10) and then deciding what needs to be done to get them there (done here)	VA, VAE VB, VBE VA1, VA1E VA2, VA2E VB1, VB1E VB2, VB2E
1.2.1.1	*Interpret Strategy Choices* *Plan 1.2.1.1:* *Do 1 & 2 in any order,* *Do 3 and, if not using Single FWS strategy, do 4,* *If using Single FWS, do 5&6,* *If using Decoupled FWS, do 5&7,* *If using 3Valve & MI1<10, do*		Following this rule/plan inevitably implements the valve complexity reduction strategy, and always prefers to use the VA stream first. Other approaches and	

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	5&8, *If using 3Valve & MI2<10, do 7&9,* *If using Full FWS, do 8&9,* *Do 10 for all unused valves*		combinations are, of course, possible.	
	1. Choose MIE1	sequential		See below
	2. Choose MIE2			See below
	3. Let VA SP = 10			VAE
	4. Let VB SP = 10			VBE
	5. Let VA1 SP = MIE1	occasional		VA1E, MIE1
	6. Let VA2 SP = MIE2			VA2E, MIE2
	7. Let VB2 SP = MIE2			VB2E, MIE2
	8. Let {VA2 SP + VB2 SP} = MIE2			VA2E, VB2E, MIE2
	9. Let {VA1 SP + VB1 SP} = MIE1			VA1E, VB1E, MIE1
	10. Let unused valve SPs = 0			Remaining, unused valve SPs and expectation values
1.2.1.1.1	Choose MIE1 *Plan 1.2.1.1.1:* *If CFCV, then do 1, if Batch then do 3, if CFVV and D1>10, then do 1, else do 2*			
	1. Choose MIE1 = D1	xor		D1
	2. Choose MIE1 > D1		Input capacity is determined through current settings of all valves—but it could be presented in an aggregated form . . .	D1 (and perhaps input capacity)
	3. Choose any MI1 such that MIE1 + MIE2 <= 20			D1, D2, MIE2 (and perhaps input capacity)
1.2.1.1.2	Choose MI2 *Plan 1.2.1.1.2:* *If CFCV, then do 1, if Batch then do 3, if CFVV and D1>10, then do 1, else do 2*			
	1. Choose MIE2 = D2	xor		D2
	2. Choose MIE2 > D2			D2(and perhaps input capacity)
	3. Choose any MI2 such that MIE1 + MIE2 <= 20			D1, D2, MIE1 (and perhaps input capacity)
1.2.2	Formulate Initial Heater Settings *Plan 1.2.2:* *Do 1, then do 2&3 in any order* *1. Interpret Strategy Choices* *2. Select Heater 1 SP* *3. Select Heater 2 SP*	sequential		T1, T2 (and heat input range capacities) T0 EI1, EI1E, EI2, EI2E HTR1, HTR1E HTR2, HTR2E (R1, R1E, R2, R2E?) (rate of input water to each rsvr?)
1.2.2.1	Interpret Strategy Choices			

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	Plan 1.2.2.1 *??*			
1.2.3	Formulate Monitoring Plan *Plan 1.2.3* *Do 1, then do 2-4 in any order* 1. *Interpret Strategy and SP choices* 2. *Formulate expectation values* 3. *Select Rsvr1 goal level* 4. *Select Rsvr 2 goal level*	sequential		R1E R2E TR1E TR2E
1.2.3.1	Interpret Strategy and SP choices *Plan 1.2.3.1:* *??*			All valve and heater SPs
1.2.3.2	Formulate expectation Values *Plan 1.2.3.2:* *Do 1,* *Then do 2-4 in any order* 1. *Review System status* 2. *Formulate valve flow expectations* 3. *Formulate Heater flow expectations* 4. *Formulate Pump flow expectations*	sequential		Equation type knowledge (though this could be reduced to heuristics or rule-knowledge).
1.3	Execute Procedure *Plan 1.3:* *Do 1-4 in order* 1. *Configure Valves* 2. *Activate Pumps* 3. *Activate heaters* 4. *Activate demand valves*	sequential	Opportunity to support sequential constraint via interface here.	
1.3.1	Configure Valves *Plan 1.3.1:* *Do 1-6 in any order* 1. *Place VA at SP* 2. *Place VB at SP* 3. *Place VA1 at SP* 4. *Place VA2 at SP* 5. *Place VB1 at SP* 6. *Place VB2 at SP*		Note that current steps only needed info about setting ranges and current settings, now we need to actually interact and control settings	VA, VAE VB, VBE VA1, VA1E VA2, VA2E VB1, VB1E VB2, VB2E And ability to control SPs
1.3.2	Activate Pumps *Plan 1.3.2:* *Do 1&2 in any order for each pump with downstream flow* 1. *Activate Pump A* 2. *Activate Pump B*			PA, PAE PB, PBE And ability to control pumps
1.3.3	Activate Heaters *Plan 1.3.3:* *Once reservoir contains water, do in any order for each reservoir in use* 1. *Place Heater 1 at SP*			R1, R2 HTR1, HTR1E HTR2, HTR2E And ability to control heaters

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	2. *Place Heater 2 at SP*			
1.3.4	Activate Demand Valves *Plan 1.3.4:* *Once reservoir contains water at demand temp, do 1&2 in any order for each reservoir in use.* *1. Place VO1 at SP* 2. *Place VO2 at SP.*		I've assumed that you don't want to feed bad product out. Different social-organizational priorities might change this procedure.	R1, R2 R1E, R2E, T1, T2, D1, D2 TR1, TR1E, TR2,. TR2E VO1, VO2 And ability to control demand valves
1.4	Observe Results *Plan 1.4:* *Using expectations generated in 1.2.3.2,* *Do 1-4 in any order*			
	1. Compare Pump Flow to expectation in initial procedure			PA, PAE PB, PBE
	2. Compare Valve Flow to expectation in initial procedure			VA, VAE, FA VB, VBE, FB VA1, VA1E, FA1 VA2, VA2E, FA2 VB1, VB1E, FB1 VB2, VB2E, FB2
	3. Compare Heat Flow to expectation in initial procedure			HTR1, HTR1E, FH1 HTR2, HTR2E, FH2
	4. Compare Demand Flow to expectation in initial procedure			VO1, D1, FO1 VO2, D2, FO2 TR1, T1, TR1E TR2, T2, TR2E
1.5	Abort *Plan 1.5:* *Do 1 and 2 in order,* *If either rsvr level >50%, do 3* *Then do 4 and 5 in order*		Not always done, could be an on-demand item in the interface	
	1. Turn off pumps			PA, PB, control over pumps
	2. Turn off heaters			HTR1, HTR2, control over heaters, FH1, FH2, HTR1E, HTR2E
	3. Drain rsvrs to <50%	occ		R1, R2, VO1, VO2, control over VO valves
	4. Close demand valves			VO1, VO2, D1, D2, control over VO valves
	5. Reset flow valves	sequential		VA, VAE, FA VB, VBE, FB VA1, VA1E, FA1 VA2, VA2E, FA2 VB1, VB1E, FB1 VB2, VB2E, FB2 Control over flow valves
1.6	Formulate Stabilize Procedure			

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	Plan 1.6: *As rsvr nears volume and/or temp goals, do step corresponding to that rsvr.*			
	1. Formulate Rsvr 1 procedure			
	2. Formulate Rsvr 2 procedure			
1.6.1	Formulate Rsvr 1 procedure *Plan 1.6.1:* *12.2.1.1.1.1 Do 1&2 in any order*			
	1. Determine Flow Adjustments			D1, T1, R1, R1E, Flows and settings and expectation values for valves feeding R1, MI1, MI1E, MO1, MO1E
	2. Determine Heat Adjustments			D1, TO, T1, R1, R1E, HTR1, HTR1E, FH1, FH1E, EI1, EI1E, EO1, EO1E
1.6.1.1	Determine Flow adjustments *Plan 1.6.1.1* *??*			Rolled up above
1.6.1.2	Determine Heat Adjustments *Plan 1.6.1.2* *??*			Rolled up above
1.6.2	Formulate Rsvr 2 procedure *Plan 1.6.2:* *12.2.1.1.1.2 Do 1&2 in any order*			
	1. Determine Flow Adjustments			D2, T2, R2, R2E, Flows and settings and expectation values for valves feeding R2, MI2, MI2E, MO2, MO2E
	2. Determine Heat Adjustments			D2, TO, T2, R2, R2E, HTR2, HTR2E, FH2, FH2E, EI2, EI2E, EO2, EO2E
1.6.2.1	Determine Flow adjustments *Plan 1.6.2.1* *??*			Rolled up above
1.6.2.2	Determine Heat Adjustments *Plan 1.6.2.2* *??*			Rolled up above
1.7	Execute Procedure *Plan 1.7:* *Do 1 before 3* *Do 2 before 4* *Otherwise, do in any order.*			
	1. Stabilize Flow thru Rsvr 1	sequential		
	2. Stabilize Flow thru Rsvr 2			
	3. Stabilize Temp in Rsvr 1			
	4. Stabilize Temp in Rsvr 2			
1.7.1	Stabilize Flow thru Rsvr 1 *Plan 1.7.1:* *Do 1 and/or 2 as needed*			

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	1. Adjust inflow valve setpoints			Settings, flows and expectations for valves feeding R1, R1, MI1, MO1, control over flow valves
	2. Adjust Demand valve SP			VO1, D1
1.7.2	Stabilize Flow thru Rsvr 2 *Plan 1.7.2:* *Do 1 and/or 2 as needed*			
	1. Adjust inflow valve setpoints			Settings, flows and expectations for valves feeding R2, R2, MI2, MO2, control over flow valves
	2. Adjust Demand valve SP			VO2, D2
1.7.3	Stabilize Temp in Rsvr1 *Plan 1.7.3* *Do 1*			
	1. Adjust Heater 1 SP			HTR1, HTR1E, FH1, T1, EI1, EO1
1.7.4	Stabilize Temp in Rsvr2 *Plan 1.7.4* *Do 1*			
	1. Adjust Heater 2 SP			HTR2, HTR2E, FH2, T2, EI2, EO2
1.8	Observe Results *Plan 1.8:* *Do 1-3 in any order*			
	1. Compare Rsvr levels to goal states			R1, R1E, MI1, MO1, MI1E, MI2E R2, R2E, MI2, MO2, D1, D2
	2. Compare Temp levels to goals states			TR1, TRIE, TR2, TR2E, EI1, EO1, EI2, EO2, EI!E, EI2E, T1, T2
	3. Compare Demand flows to goal states.			D1, VO1, D2, VO2 T1, TO1, T2, TO2
2	Normal Operations *Plan 2:* *If demands change, do 2* *Else, do 1 until time or volume goals are met*	Goal is usually 5 min continuous operations at demand states		
	1. Maintain Steady Demands			
	2. React to Changing Demands			
2.1	Maintain Steady Demands *Plan 2.1* *If using CFCV, do 1* *Else, do 2*			
	1. Operate Steady State	xor		
	2. Operate Variable Flow			
2.1.1	Operate Steady State *Plan 2.1.1* *Do 1-3 continuously in Parallel* *When deviation noted, do 0.4* *When completion noted, do 0.3*			
	1. Monitor for Mass Deviations	parallel		MI1, MI2, MO1, MO2, MI1E, MI2E, MO1E, MO2E
	2. Monitor for Temp			EI1, EI2, EO1, EO2, T1, T2, TR1, TR2,

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	Deviations			and expectations for all the above
	3. Monitor for Completion			Time and/or total flow indications
2.1.2	Operate Variable Flow *Plan 2.1.2* *Do 1,* *When volume goal reached, Do 2* *Do 3-5 continuously in parallel,* *When deviation noted, do 0.4,* *when completion noted, do 0.3*			
	1. Monitor for Interim Volume Goals	sequential		R1, R1E, R2, R2E
	2. Adjust Valves and Heaters		Similar to startup, but needs to include continued monitoring of current state	As for startup
	3. Monitor for Mass Deviations	parallel		As for 2.1.1
	4. Monitor for Temp deviations			
	5. Monitor for Completion			
2.2	React to Changing Demands		Similar to startup, but needs to include continued monitoring of current state relative to expectations and goals. Info about constraints on and interactions between actions and parameters important here.	As for start up
3.	Shut Down *Plan 3:* *Upon request or goal completion,* *Do 1 or 2*			Shut down request or completion indication
	1. Do Matched Shutdown	xor		
	2. Do Individualized Shutdown			
3.1	Do Matched Shutdown *Plan 3.1* *Do 1 then do 2*			
	1. Match like component states	sequential	Potential for use of dynamic	Paired state info: e.g., FA:FB, R1:R2, etc.

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
		sequential	organization of feed streams here	
	2. Perform shutdown actions			
3.1.2	Perform shutdown actions *Plan 3.1.2* *Do 1-6 in order,* *Then after heater output = 0,* *Do 7-14 in order*			
	1. Ensure R1<= 80% capacity	sequential		R1
	2. Ensure R2<= 80% capacity			R2
	3. Close VO1			VO1 & controls, FO1
	4. Close VO2			VO2 & controls, FO2
	5. Turn off HTR1			HTR1 & controls, FH1
	6. Turn off HTR2			HTR2 & controls, FH2
	7. Turn off PA			PA & controls, FA
	8. Turn off PB			PB & controls, FB
	9. Close VA			VA & controls, FA
	10. Close VB			VB & controls, FB
	11. Close VA1			VA1 & controls, FA1
	12. Close VA2			VA2 & controls, FA2
	13. Close VB1			VB1 & controls, FB1
	14. Close VB2			VB2 & controls, FB2
3.2	Do individualized Shutdown *Plan 3.2* *Do 1 and 2 concurrently*			
	1. Shutdown R1 Stream	parallel		
	2. Shutdown R2 Stream			
3.2.1	Shutdown R1 Stream *Plan 3.2.1* *Do 1-3 in order,* *Then after heater out put = 0,* *Do 4-6 in order* *Do 7 after PB is off*		Potential for use of dynamic organization of feed streams here	
	1. Ensure R1<= 80% capacity	sequential		As for appropriate parts of 3.1.2
	2. Close VO1			
	3. Turn off HTR1			
	4. Turn off PA			
	5. Close VA			
	6. Close VA1			
	7. Close VB1			
3.2.2	Shutdown R2 Stream *Plan 3.2.2* *Do 1-3 in order,* *Then after heater output =0,* *Do 4-6* *Do 7 after PA is off*			
	1. Ensure R2<= 80% capacity	sequential		As for appropriate parts of 3.1.2
	2. Close VO2			
	3. Turn off HTR2			
	4. Turn off PB			

Superordinate	Tasks: operations and plans	Freq./Duration/ Sequencing	Notes	Information Requirements
	5. Close VB			
	6. Close VB2			
	7. Close VA2			
4.	Fault Management *Plan 4:* *Upon fault detection,* *Do 1 or 2 or both in any order*		Means to detect faults are generally in previous steps (esp. monitoring steps)	
	1. Do Root Cause Diagnosis			?? Maybe physical form and location info in addition to most other things listed
	2. Do Fault Remediation			
4.2	Do Fault Remediation *Plan 4.2:* *Do 1 or 2 depending on type of fault detected*			
	1. Remediate Mass Fault			
	2. Remediate Energy Fault			EI1, EO1, EI1E, E1OE, HTR1, HTR1E, HTR2, HTR2E, TR1, TR2, TR1E, TR2E
4.2.1	Remediate Mass Fault *Plan 4.2.1* *Do the step which correlates with the fault detected*			
	1. Remediate Excess MO	xor		
	2. Remediate Excess MI			
	3. Remediate Excess R			
	4. Remediate Insuff. MO			
	5. Remediate Insuff. MI			
	6. Remediate Insuff. R			
4.2.1.1	Remediate Excess MO Plan 4.2.1.1 Do 1, If VO<>D (or DE), then do 2 Else, 3.			
	1. Check VO SP	sequential		VO1, VO2, FO1, FO2, D1, D2
	2. Adjust VO SP			Controls over VO1 and VO2
	3. Abort			As for 1.5
4.2.1.5	Remediate Insuff. MI *Plan 4.2.1.5* *Try to do 1, if impossible, do 2*			
	1. Increase MI	xor		Flow valves settings, controls, flows, pump settings
	2. Shift to CFVV			