NUREG/CR-1381
SAND80-7028
Unlimited Release

A Methodology for Evaluating Safeguards Capabilities for Licensed Nuclear Facilities — Final Report

Harold A. Bennett, M. Teresa Olascoaga, Sandia National Lal ratories
Alan Sicherman, Gary Smith, Woodward-Clyde Consultants

Printed March 1980

Sandia National Laboratories

SF 2900-Q(3-80)

Prepared for

U. S. NUCLEAR REGULATORY COMMISSION

NOTICE

Available from

GPO Sales Program

Division of Technical Information and Document Control
U.S. Nuclear Regulatory Commission
Washington, D.C. 20555

and

National Technical Information Service
Springfield, Virginia 22161

NUREG/CR-1381
SAND80-7`28
Unlimited Release

A METHODOLOGY FOR EVALUATING SAFEGUARDS CAPABILITIES FOR LICENSED NUCLEAR FACILITIES - FINAL REPORT

H. A. Bennett, M. T. Olascoaga
Sandia National Laboratories
Albuquerque, NM 87185

A. Sicherman, G. Smith
Woodward-Clyde Consultants
San Francisco, CA 94111

Date Published: March 1980

Submitted by
Woodward-Clyde Consultants
San Francisco, CA 94111
under
Contract Document No. 13-2215
to
Sandia National Laboratories
Albuquerque, New Mexico 87185
operated by
Sandia Corporation
for the
U.S. Department of Energy

Prepared for
Division of Safeguards
Office of Nuclear Material Safety and Safeguards
U.S. Nuclear Regulatory Commission
Washington, DC 20555
Under Interagency Agreement DOE 40-550-75
NRC FIN No. A1153

ABSTRACT

This report describes work performed by Woodward-Clyde Consultants under contract to Sandia Laboratories for assistance in the development and implementation of an evaluation methodology. This methodology was developed to aid the NRC in its evaluation of fixed-site physical protection system performance relative to the Physical Protection Upgrade Rule, 10 CFR Part 73.45

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 EXECUTIVE SUMMARY

Sandia Laboratories contracted with Woodward-Clyde Consultants (WCC) to assist in developing a methodology for evaluating safeguards capabilities at licensed nuclear facilities. The total effort was divided into two phases. Phase I was concerned primarily with the development of a preliminary evaluation algorithm. The results of this effort were described in [1]. Phase II which is reported here was devoted to completing and refining the algorithm. The evaluation methodology is to aid in the implementation of new NRC regulations (the Physical Protection Upgrade Rule), which are designed to upgrade the physical security of fuel cycle facilities. The methodology could also be used to provide guidance to licensees in meeting the safeguard system capability requirements.

Five performance capabilities[1] of physical protection systems were specified by the NRC in the Physical Protection Upgrade Rule, 10 CFR Part 73.45, paragraphs (b) - (f):

1. Prevent unauthorized access of persons and materials into Material Access Areas (MAAs) and Vital Areas (VAs).
2. Permit only authorized activities and conditions within Protected Areas (PAs), MAAs, and VAs.

[1]Throughout this report, certain terms are used that have a particular meaning in the context of the evaluation procedures. To avoid confusion, these terms are defined in Section 6.0 (glossary).

3. Permit only authorized placement and movement of Strategic Special Nuclear Material (SSNM) within MAAs.
4. Permit removal of only authorized and confirmed forms and amounts of SSNM from MAAs.
5. Provide for authorized access and assure detection of and response to unauthorized penetration of the PA.

To evaluate these five capabilities in a logical manner, the disaggregation structure shown in Figures 1-1 (1) through 1-1 (5) was developed by Sandia Laboratories with the cooperation of the NRC. WCC was to assist in formulating an algorithm by which individual safeguard system component (equipment and/or procedure) assessments could be combined into a meaningful score or series of scores indicating the total adequacy of safeguards at a particular facility.

1.1 TASKS

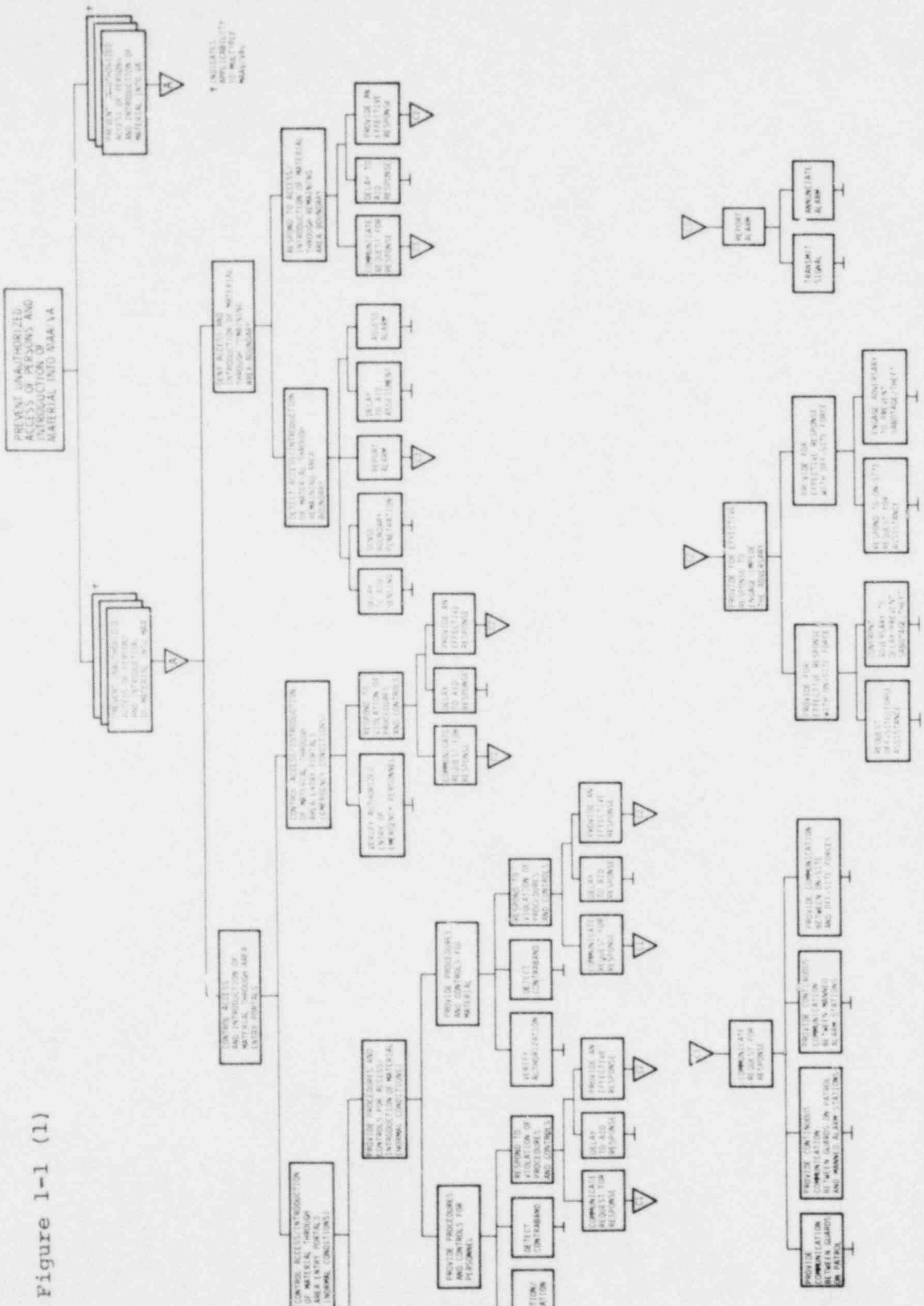
Four tasks were specified by Sandia Laboratories for Phase II.

Task 1: Assist Sandia personnel in defining the remaining performance characteristics not defined in Phase I.

Task 2: Assist Sandia in developing the remaining effectiveness test questionnaires (for assessing particular components) not defined in Phase I.

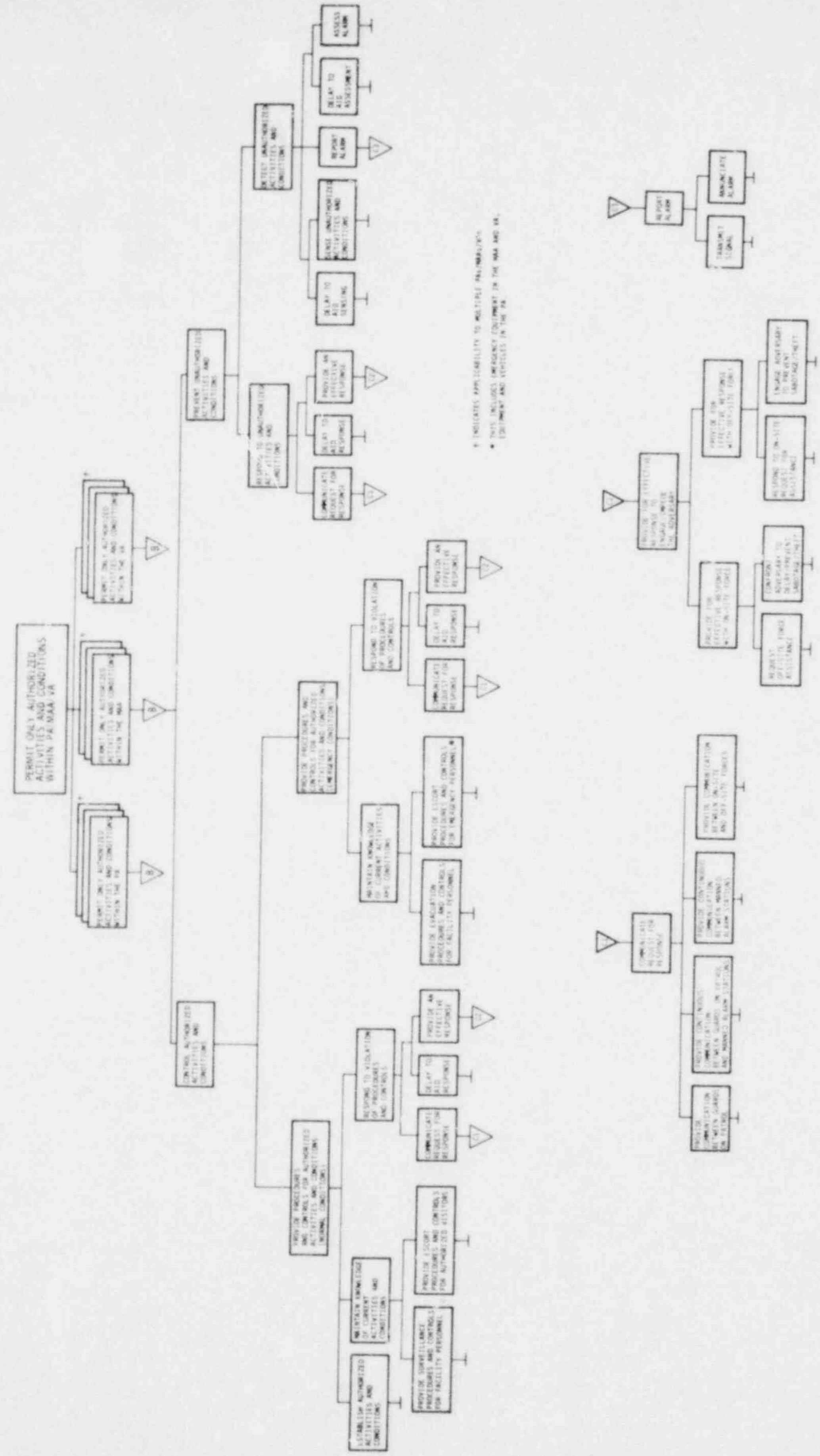
Task 3: Assist Sandia personnel in developing the remaining component combination scoring rules not developed in Phase I for jobs identified in single performance characteristics.

Figure 1-1 (1)



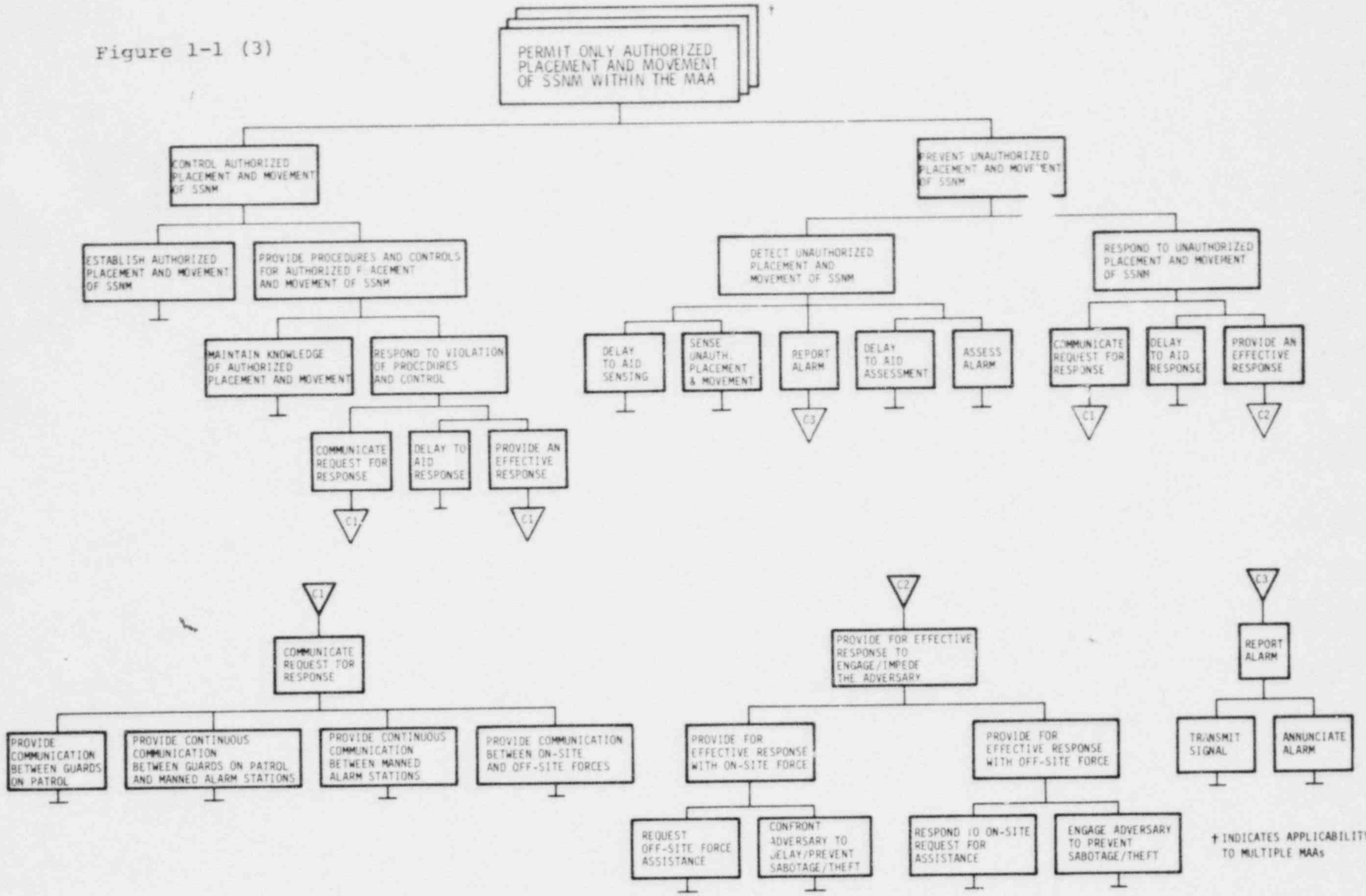
A FUNCTIONAL HIERARCHY FOR PROPOSED RULE PART 73.45 (D)

Figure 1-1 (2)



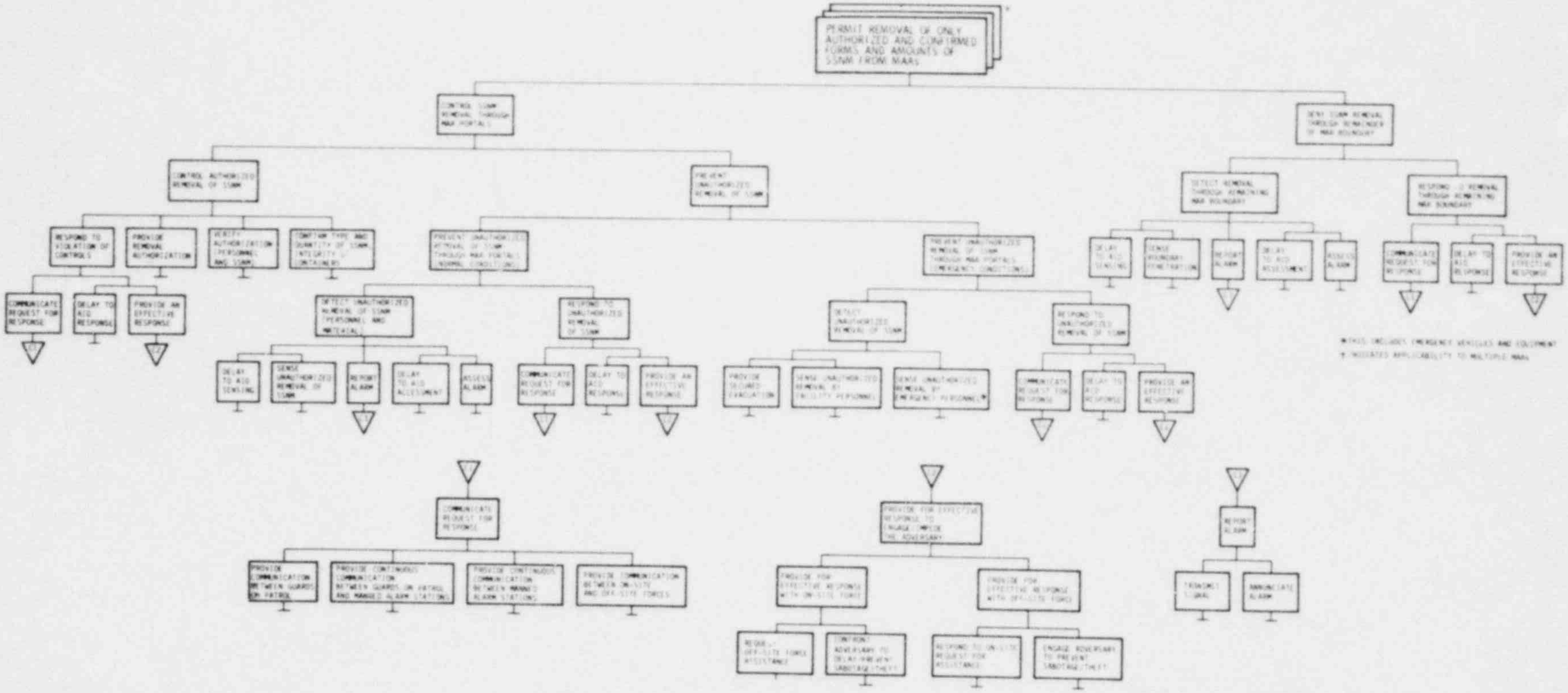
A FUNCTIONAL HIERARCHY FOR PROPOSED RULE PART 73.45 (c)

Figure 1-1 (3)



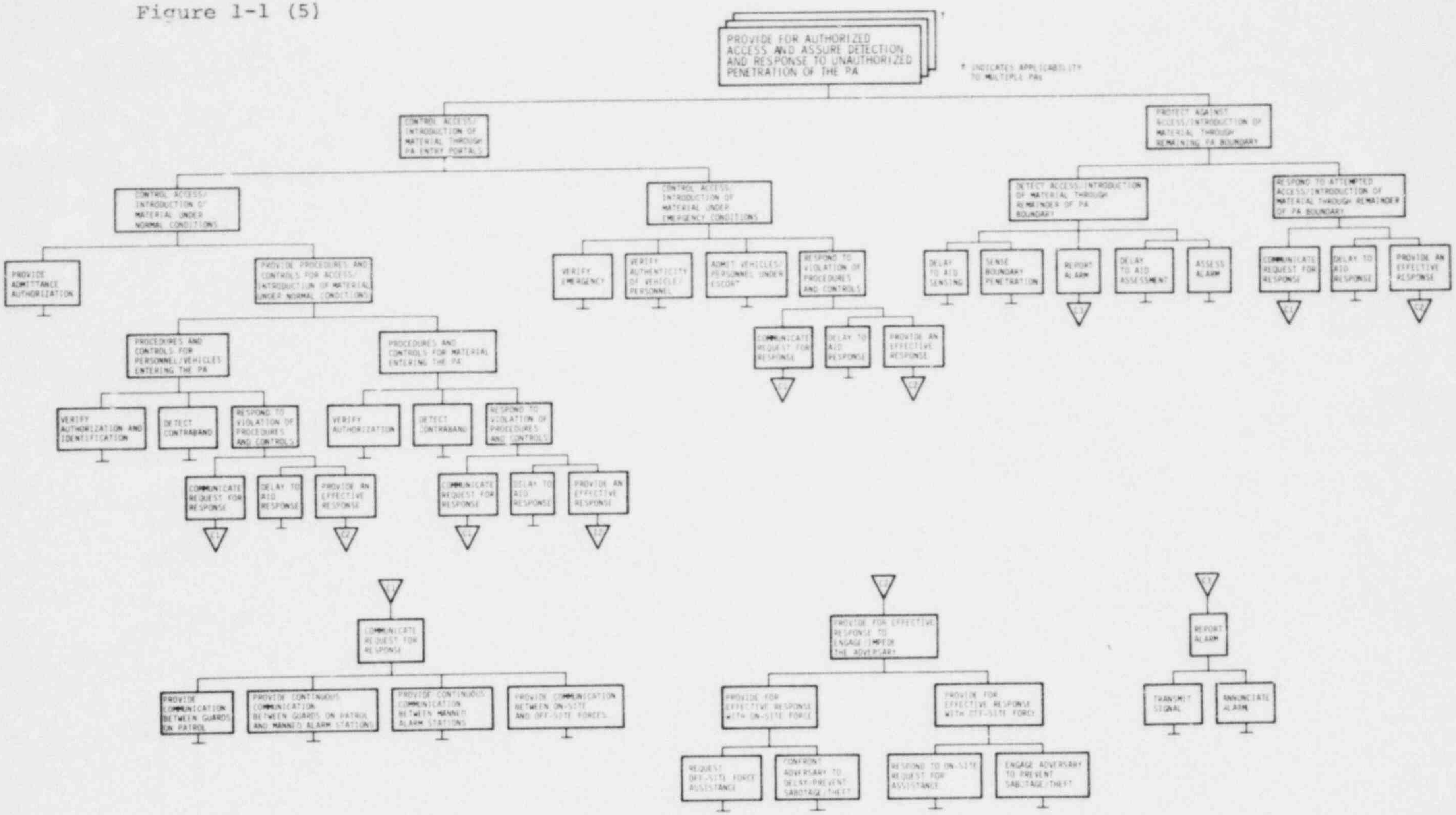
A FUNCTIONAL HIERARCHY FOR PROPOSED RULE PART 73.45 (d)

Figure 1-1 (4)



A FUNCTIONAL HIERARCHY FOR PROPOSED RULE PART 73.45 (e)

Figure 1-1 (5)



A FUNCTIONAL HIERARCHY FOR PROPOSED RULE PART 73.45 (f)

Task 4: Assist Sandia personnel in developing the aggregation algorithm in detail and provide an illustration of its implementation using hypothetical data provided by Sandia.

An ANSI standard FORTRAN computer program which would exercise the aggregation algorithm was to be provided.

WCC was to furnish a comprehensive document describing the algorithm and its development. Detailed guidelines for implementing the algorithm were to be provided along with documentation and instructions on the use of the computer program.

1.2 SUMMARY

The work done by WCC focused on the following areas:

- Providing an algorithm for combining component questionnaire responses to obtain an overall component effectiveness score.
- Providing an algorithm for combining component effectiveness scores to obtain an overall score for the performance characteristic for which the components were selected.
- Providing an algorithm for combining delay/response type scores.
- Providing an algorithm for combining the higher levels of the hierarchy for any particular performance capability.
- Providing a computer program for synthesizing these algorithms to evaluate portions of or an entire performance capability.

- Providing implementation guidelines for the entire methodology.

- Providing recommendations for future work.

Each of these areas is now briefly summarized.

1. Aggregating Component Questionnaire Responses

A methodology for aggregating questionnaire responses has been developed (primarily in Phase I) that is logical and defensible and yet practical to implement. The basic aggregation formula has a theoretical basis in both utility and probability theory, thus aiding defensibility. The evaluator's task is simplified by only requiring responses to multiple choice questions that concern the description of the component. The methodology allows for individual weighting of questions to reflect their relative importance and also for varying types of interaction (e.g., non-additive, additive) among question responses to yield an overall score for any individual component. The basic formula can also be used at higher levels of the evaluation hierarchy.

2. Aggregating Components to Evaluate Performance Characteristics

A methodology that indicates how several components coordinate with each other in addressing a performance characteristic has been developed. An evaluator answers multiple choice questions that determines how the component scores should be combined. The evaluator's responses can reflect facility dependent implementation of components.

3. Aggregating Delay/Response Type Scores

A methodology that allows for the direct comparison of delay times and response, monitoring, or assessment times has been developed to allow meaningful evaluations of delay/response type elements of a safeguard system.

4. Aggregating Higher Levels of the Hierarchy

The same methodology elements developed for the lower levels of the hierarchy are used to complete the evaluation of the hierarchy. Such aspects as multiple access points are addressed in these aggregations.

5. Computer Program Implementation

A computer program that performs the safeguards evaluation computations has been developed. The program takes as input questionnaire and hierarchy formats and the evaluator's responses to the multiple choice questionnaires. The program computes the scores for all components, performance characteristics and higher level elements of a capability hierarchy. It provides for sensitivity analysis on questionnaire weights and responses, and on the interaction of hierarchy elements. The program is interactive and has hierarchy display features.

6. Methodology Implementation Guidelines

Guidelines for developing component questionnaires, and assessing scoring rules and weights have been developed. Data requirements for the algorithm are specified and ways of interpreting evaluation and sensitivity analysis results are described.

7. Recommendations for Future Work

Suggestions for improving the implementation of the algorithm are presented. Ways of extending the algorithm to combine capabilities across facility area boundaries (e.g., PA and MAA) to evaluate an overall safeguards capability are discussed. Limitations of the methodology and general approach are reviewed.

All of these points are discussed in detail in the various sections and appendices of the report.

1.3 OUTLINE OF REPORT

Section 2 of this report contains a technical summary of the general methodology. Section 3 contains a discussion of how the methodology is implemented. Section 4 describes the computer program that performs the computations specified by the algorithm. A single capability for a safeguards facility using hypothetical data is evaluated to provide an illustration of how the algorithm and computer program work. Section 5 contains a critical summary of the methodology and recommendations for future work. The appendices provide supplementary technical information and data used in the computer example.

2.0 TECHNICAL SUMMARY OF EVALUATION METHODOLOGY

2.1 INTRODUCTION

In order to address the specifics of the Physical Protection Upgrade Rule the capabilities hierarchy (or disaggregation structure) shown in Figures 1-1(1) through 1-1(5) was developed by the NRC and Sandia Laboratories. Each of the five performance capabilities is treated as a separate objective, with its own independent hierarchy or disaggregation structure.

The upper level of each separate structure is the performance capability, as specified by the NRC Upgrade Rule. These are followed by system functions, which a system must perform in order to meet the specified capability. The major functions are further broken down into system subfunctions, which identify specific tasks to be performed by the system.

Each system subfunction is disaggregated into specific low-level system tasks which form the lowest level of the hierarchy. Performance characteristics relate these low-level system tasks to components. These performance characteristics correspond to the rows of the Component Selection Matrices (Figure 2.1), while the columns of the matrix represent specific components (equipment and procedures). Thus the Component Selection Matrices describe what approaches (equipment and procedures) are acceptable to perform specific tasks (performance characteristics).

Figure 2-1. COMPONENT SELECTION MATRIX EXAMPLE

DELAY		EQUIPMENT AND/OR DESIGN FEATURES																PROCEDURES
	PERFORMANCE CHARACTERISTICS	FENCE SYSTEMS	GATES AND ASSOCIATED HARDWARE	SALLY PORTS, PEDESTRIAN	SALLY PORTS, VEHICLE	VAULTS	DOORS AND ASSOCIATED HARDWARE	WINDOWS AND ASSOCIATED HARDWARE	WALLS	ROOFS	FLOORS	EMERGENCY EXITS	AIR & UTILITY INLET BARRIERS	ISOLATION ZONES	LOCKS	SNM CONTAINERS	SNM HOLDING/STORAGE AREAS	GUARD INTERVENTION
PROTECTED AREA	DELAY ADVERSARY TO AID DETECTION	●	●	●	●				●	●				●	●			
	DELAY ADVERSARY TO AID ASSESSMENT	●	●	●	●				●	●				●	●			
	DELAY ADVERSARY TO AID RESPONSE	●	●	●	●				●	●				●	●			●
MATERIAL ACCESS AREA	DELAY ADVERSARY TO AID DETECTION			●		●	●	●	●	●	●	●	●		●	●	●	
	DELAY ADVERSARY TO AID ASSESSMENT			●		●	●	●	●	●	●	●	●		●	●	●	
	DELAY ADVERSARY TO AID RESPONSE			●		●	●	●	●	●	●	●	●		●	●	●	●
VITAL AREA	DELAY ADVERSARY TO AID DETECTION			●			●	●	●	●	●	●	●		●			
	DELAY ADVERSARY TO AID ASSESSMENT			●			●	●	●	●	●	●	●		●			
	DELAY ADVERSARY TO AID RESPONSE			●			●	●	●	●	●	●	●		●			●

WCC's contract required that a set of aggregation schemes or algorithms be developed with which one could work back up the hierarchy to arrive at a single score for each capability. To do this, individual components (the lowest hierarchy level) must be assigned an overall score based on evaluators' responses to questionnaires. In the case of equipment, this single score must reflect such factors as general performance, installation, maintenance, reliability, and vulnerability. For procedures, the score should encompass aspects of training, vulnerability, and contingencies. All of these issues are addressed in the component questionnaires.

Once each component has received a score, scores for those components that address an individual performance characteristic must be aggregated to arrive at a single score for the appropriate low-level system task via the performance characteristic. Continuing up the hierarchy, scores on low-level system tasks are combined into system subfunction scores, which are then aggregated into system function scores, and finally, into an overall score for each performance capability.

General Aggregation Concepts

To be practical in terms of input requirements, an algorithm needs to operate with input consisting of subjective responses (using descriptive multiple choice response scales) to a large number of questions. Specifically, there are questionnaires for each component or procedure and system questionnaires addressing the interaction between low-level tasks and between components.

Care must be taken when using the responses to these questions. A practical evaluation algorithm cannot require information that is unrealistic to obtain from these questionnaire responses. Because of the large number of questions and their potential modifications, it also should not require a lengthy calibration procedure for each question.

In addition, the range of possible responses to these questions is not uniform, and the questions can differ in relative importance.

Within these practical constraints, an algorithm must still produce meaningful results. The computational rules and assumptions must not be arbitrary. The algorithm should be capable of providing the correct answer where inputs and their interactions are precisely known, since this is a primary means of checking the reasonableness of the algorithm.

Because of the need for meaningful and defensible results, it is desirable to use aggregation schemes based on well-developed methodologies. Two such methodologies are multiattribute utility analysis (decision analysis), and probability analysis. The former provides mechanisms for aggregating multiple criteria into a single overall score, and is particularly useful when subjective considerations are involved. Probability analysis also specifies how to derive a measure for a system in terms of its components and their interactions.

In the safeguards problem, both decision analysis and probability analysis appear desirable. The latter addresses the probability that the system will perform adequately in the event of specific types of adversary actions. This must be the underlying concern when evaluating capabilities and when characterizing interactions between system elements. Individual component successes or failures have different impacts on the probability of total system success or failure depending on system interactions. An algorithm should produce meaningful results _if_ probabilities and system interactions (e.g., fault trees) could be specified. On the other hand, the requirements for practical inputs using subjective questionnaires and responses make it too restrictive to assume that such probabilities can be derived from the input data. The input may be related to probabilities but a direct quantitative relationship cannot be assumed. An alternate approach is to combine subjective scales using decision analysis preference functions. These do not need to assume a one-to-one correspondence between response scales and probabilities, but rather reflect judgments as to the relative comparison of alternative components and systems. Still, it is desirable that if the probabilities were actually known, the scheme used to combine scores would give the correct results.

The algorithms to be presented in the following paragraphs are based on aggregation rules specified by decision analysis under certain assumptions. The assumptions are made in order to make the implementation procedures as practical as possible, hopefully without omitting any important features of the problem. These assumptions could be relaxed, but more calibration effort would be required. The aggregation models used allow for differential weightings of elements, and can reflect different interactions among elements. The results of the aggregation can be used as part of a logical and consistent judgmental comparison of alternative systems. In addition, the algorithms to be presented yield the correct results when certain assumptions and actual probabilities are used in the computation. Thus the aggregation logic can be interpreted using probability notions as well as preference concepts. The use of both analytical approaches helps to give the algorithm its practical and defensible characteristics.

The algorithms for the specific types of aggregations required to evaluate a capability hierarchy will now be individually discussed. For each case, the algorithm is described in terms of what features can be reflected by the computation. Implementation guidelines, discussed in Section 3 of this report, include further discussion of how certain parameters are assessed and interpreted.

2.2 COMPONENT EVALUATION

The algorithm for evaluating components was developed in Phase I and is discussed in detail in the Phase I report [1]. The discussion here reviews the main assumptions and results.

The response scale for each individual question on a component questionnaire is considered to be an attribute or measure. These measures have been developed with an orientation towards aspects of a component that can hinder its performance. The highest response on a question connotes that the factor being considered will not be compromised because of the particular component. A lower response connotes that the factor will have a certain degree of compromise depending upon the range of the response scale. The methodology allows a scaling parameter or weight to be applied to each question. This weight essentially normalizes response scales whose ranges cause them to differ in relative importance. For implementation practicality, relative preferences over individual response scales are assumed to be linear. We now define the following notation for question "i":

X_i = unadjusted question response normalized to go between 0 and 1.
W_i = weight asigned to reflect total range of the response scale.
S_i = adjusted question response or question score.

The formula connecting these three quantities is:

$$S_i = 1-W_i(1-X_i)$$

where all three quantities are restricted to the range between 0 and 1 as a scaling convention, with 1 being the best.[1] Note that if X_i=1

---

[1]Note that if any other range is used (say 1 to 5) it is trivial to normalize this to the range 0, 1. This also allows direct comparison between questions with differing numbers of possible responses.

then S_i=1 and if X_i=0, then S_i=1-W_i. Thus, only if W_i=1, can S_i=0. W_i=.2, for example, implies a minimum possible score of S_i equal to .8. Thus, a question with little importance with respect to a particular factor can have only a small effect on the score.

The scores for the questions can be thought of as simplified single-attribute utility functions that have been normalized so that they have equal importance. For a particular group of N attributes, we will make the assumption that they are mutually utility independent.[2] The results of multiattribute utility theory [2] allows us (if certain assumptions are made) to define an overall group score S normalized between 0 and 1 as:

$$S = C\sum_{i=1}^{N} S_i + KC^2 \sum_{i=1}^{N} \sum_{j>i} S_i S_j + \ldots + K^{N-1} C^N \prod_{i=1}^{N} S_i \quad (1)$$

$$\text{where } 1 + K = (1 + K\,C)^N \quad (2)$$

If we define V = KC then

$$S = \frac{1}{K}\left[V \sum_{i=1}^{N} S_i + V^2 \sum_{i=1}^{N} \sum_{j>i} S_i S_j + \ldots + V^N \prod_{i=1}^{N} S_i \right] \quad (3)$$

$$\text{where } K = (1 + V)^N - 1 \quad (4)$$

This leaves us with only one scaling constant (V) to evaluate.

---

[2]See, for example, R.L. Keeney and H. Raiffa, Decisions with Multiple Objectives; Preferences and Value Tradeoffs. New York: Wiley, 1976. Also see Appendix A1.

Before proceeding further, however, we briefly review the results to this point. The necessity of handling large numbers of questions and questionnaires imposes a practicality constraint on the complexity of the aggregation algorithm that can be realistically used. The assumptions leading to the above formula essentially allow us to decompose the problem into considering individual questions somewhat in isolation and then to combine the results. With all the simplified assumptions, the algorithm still reflects the key aspects of the importance range of the questions via the W_i and the way in which factors interact via V. Decision analysis enables the use of the formula above to compare different sets of responses. A set with a higher score is preferred to one with a lower score. The consequences of assigning different values of V with respect to both preference and probability interpretations will now be examined. In the discussion to follow, the entire set of questions for a particular component will be referred to as belonging to one group with a single interaction coefficient V. A general case of considering each component questionnaire as a "mini hierarchy" in itself will be discussed in Section 3 under implementation guidelines.

A. V=0. If we take limits as $V \to 0$ in (3) we get

$$S = \sum_{i=1}^{N} S_i / N .$$

Thus the overall score S is the mean of the individual scores. This is appropriate if an individual score makes the same incremental contribution to component quality regardless of the fixed levels of the other scores. It is also appropriate if the component behaves in a way such that one factor is chosen at random, and the component as a whole succeeds or fails on the basis of the one factor.

B. V=-1. If we substitute V=-1 into (3) we get

$$S = 1 - \prod_{i=1}^{N} (1 - S_i) .$$

This is appropriate if each factor can substitute completely for another factor in order for the component to work. A factor contributes incrementally more when other factors are low (e.g., this one is needed) than when other factors are high (this one is not really needed).

In fault tree theory, this is interpreted as the computation for an OR gate. If S_i is the probability that the factor associated with question i works, and the overall component works if any of the individual factors works, then the probability of overall sucess is given by S.

C. V→∞. If we take limits as V→∞, we get

$$S = \prod_{i=1}^{N} S_i .$$

This is appropriate if a factor contributes incrementally more when other factors are high rather than low, and does not contribute at all if any of the other factor scores is zero.

In fault tree theory, this is as the computation for an AND gate. In this case, every factor must succeed for the overall component to succeed.

D. V=1. If we substitute V=1 into (3) we get

$$S = \frac{1}{2^N - 1} \left[\sum S_i + \sum_{i=1}^{N} \sum_{j>i} S_i S_j + \dots + \prod_{i=1}^{N} S_i \right] \quad (5)$$

$$= \frac{1}{2^N - 1} \left[\prod_{i=1}^{N} (S_i + 1) - 1 \right] \quad (6)$$

In examining (5) (if the S_i are assumed to be appropriate probabilities, e.g., factors are mutually probabilistically independent or the S_i are appropriate conditional probabilities) it can be seen that this is the average of the probabilities that each factor succeeds in a particular subset of the S_i taken over all possible subsets. Thus S can be interpreted as the probability that all factors will succeed in a subset of the S_i chosen at random. This is appropriate if the factors are related in a way that requires success on each, but it is possible that not all will be relevant in a given situation. This situation represents an intermediate case between the AND gate ($V = \infty$) where all factors are always relevant and the average ($V = 0$) where exactly one factor (chosen at random) determines the outcome. We will call this case a "soft AND" gate.

E. $V=-1/2$. If we substitute $V=-1/2$ into (3) we get

$$S = \frac{1}{2^N-1}\left[\sum_{i=1}^{N} (1-(1-S_i)) + \sum_{\substack{i=1,n \\ j>i}} (1-(1-S_i)(1-S_j)) + \ldots + (1-\prod_{i=1}^{N}(1-S_i))\right] \tag{7}$$

$$= \frac{2^N}{2^N-1}\left(1-\prod_{i=1}^{N}(1-1/2S_i)\right) \tag{8}$$

In examining (7) it can be seen that this is the average of the probabilities that at least one factor succeeds over all possible subsets of the set $\{S_i | i = 1, \ldots, N\}$. The value S can be interpreted as the probability of at least one success in a subset chosen at random. This is an appropriate rule if the factors can substitute for each other but it is possible that not all factors will be relevant in a given situation. This represents an intermediate case between the strict OR calculation ($V = -1$), where all factors are relevant and the average ($V = 0$), which can be interpreted as only one factor chosen at random being relevant. We will call this case a "soft OR" gate.

Thus as the constant V ranges from -1 to infinity, the scoring formula (3) covers a complete range of possible factor interactions. The most severe is when questions must all have high scores for an overall high score ($V = \infty$). The least severe is when a high score on any question gives a high overall score ($V = -1$). For $-1 < V < \infty$ there is a complete range of intermediate interactions including the mean when $V = 0$. These possible interactions have the property that if $V > V'$, then $S < S'$. As the constant V increases, the score from its related formula decreases. Thus V can be interpreted as a "strength of interrelation" coefficient that ranges from complete redundancy ("parallel circuitry") $V = -1$ to complete interdependence ("series circuitry") at $V=\infty$.

The set of algorithms using the 5 values of V just described is the basis for the aggregation scheme that is used to evaluate component questionnaires and different levels of the capability hierarchy. In principle, the parameters involved in calibrating the evaluation formula can be assessed rigorously, both by multiattribute utility theory methods and/or probability modeling. In practice, less involved calibration methods can be employed as are discussed in Section 3.

In summary, the important features to note about the component evaluation algorithm are:

1) The formula can be derived axiomatically from a set of clearly stated assumptions and is defensible in being theoretically sound from a preference function viewpoint.

2) The formula reflects the key concepts of weighting questions based on their ranges as to importance and of allowing for a variety of interaction between factors. Although simplified

for practical implementation, the formula still provides much flexibility with respect to modeling safeguard features.

3) The formula can yield the correct results when exact probabilities are known and substituted into the computation formula, given that the fault-tree like gates are assumed appropriate from a probabilistic viewpoint.

Table 2-1 summarizes the algorithm for evaluating components.

Table 2-1. COMPONENT SCORING FORMULAS

$S_i = 1-W_i(1-X_i)$

N = number of questions

V	Formula	Interpretation
-1	$S = 1 - \prod_{i=1}^{N} (1 - S_i)$	OR Gate
-1/2	$S = (\prod_{i=1}^{N} (1 - 1/2\ S_i) - 1)/(2^N/(2^N - 1))$	Soft OR Gate
0	$S = \sum_{i=1}^{N} (S_i)/N$	Average
1	$S = (\prod_{i=1}^{N} (S_i + 1) - 1)/(2^N - 1)$	Soft AND Gate
∞	$S = \prod_{i=1}^{N} (S_i)$	AND Gate

2.3 LOW LEVEL SYSTEM TASK (PERFORMANCE CHARACTERISTIC) EVALUATION

Because of the flexibility of the set of aggregation rules described in Section 2.2, it is possible to use these same rules to aggregate scores all the way up to the top of the hierarchy. A performance characteristic evaluation consists of taking the scores of several components and aggregating them to get an overall score. The components themselves may interact with each other in a fashion analogous to the way factors interact as described in Section 2.2. In general, it is not possible to select an interaction rule independently of the specific components and facility features involved. It is also desirable that an evaluator or designer should not have to select an interaction coefficient but rather, as with components, provide responses to some multiple choice questions. "System effectiveness test questionnaires" are used to decide what type of inter ction is appropriate for a particular combination of components. This concept of system questionnaire will now be discussed in more detail.

Just as there are factors that affect how a component performs, so there are system factors that affect the way a combination of components performs. Questions can be developed that address each particular system factor. Since they are multiple choice, they are given weights. An interaction rule that is generic to system factors can be selected (e.g. soft AND) and an overall "compatibility" score can be computed for any set of components. This score can be interpreted in a consistent preference function viewpoint manner to determine an appropriate interaction rule with which to combine component effectiveness scores. Appendix A1 discusses this interpretation in more detail. Thus, evaluators can use system test questionnaires to provide input to the algorithm as to which interaction rule is most appropriate for combining component effectiveness scores to produce an overall score for a low-level system task. (Section 3 will discuss the implementation of system questionnaires in more detail.) An example of several components performing a low-level system task can be multiple sensors to detect boundary penetrations. If each sensor is independent and they act as substitutes for each other (high redundancy or "parallel circuitry") a system questionnaire would indicate that their component scores should be ORed together to produce an overall score for the low-level system task. (Of course, certain performance characteristics may not require system questionnaires if it appears that a fixed interaction rule can be assigned; e.g., multiple components always soft OR for a particular characteristic.)

The system effectiveness test questionnaire is also a possible format for factoring in supplementary information about a facility where that is necessary for evaluation. Such information can include features not addressed by any particular component questionnaire but still necessary to properly evaluate a performance characteristic. In this case, the system questionnaire is analogous to a component questionnaire where some of the questions involve using calculated scores for other components. The system questionnaire can also contain questions pertaining to delay/response type scores and multiple access points, when it is natural to do so in terms of the subject matter involved in the questionnaire. These issues are the subject of the following paragraphs.

2.4 HIGHER LEVEL AGGREGATION, DELAY/RESPONSE AND MULTIPLE ACCESS POINTS

At higher levels in the hierarchy, the aggregation should proceed in a straightforward manner. Since the hierarchy is fixed in advance, the appropriate interaction for combining low level task scores into subfunction scores etc., into an overall capability score can be specified before a evaluation is made. Since these hierarchy elements are more generic in nature, the interaction rule for combining elements should not require system questionnaires.

There are two aggregation issues, however that need further discussion. These are multiple access point type concerns and delay-response type evaluations. Multiple access points refers to the fact that different parts of the facility can each be viewed individually with respect to certain elements of the generic hierarchy. These parts then need to be aggregated together. For example, in the "Prevent Unauthorized Access of Persons and Introduction of Material into the MAA" capability hierarchy, one segment refers to denying access through the remaining (non-portal) area boundary. If a building forms part of the boundary, possible access points could include windows, walls, floors, roofs, vents, etc. Each of these access points may have its own sensor system. Conceptually, the algorithm can treat each access point individually and then use an interaction rule to combine the scores of all access points. Thus, the hierarchy segment on denying access through the area boundary could multiply into denying access through walls, floors, vents, etc. In practice (see Section 3) it may be possible to avoid subdividing the boundary into many smaller units and thus avoid proliferating hierarchy elements. These same concepts just discussed can apply to multiple portals, or multiple MAAs, etc.

The second issue requiring discussion is the evaluation of delay-response, delay-detect, delay-assess relationships of a facilty. (Hereafter, delay-response will be used as an example. The treatment is analogous for all three). For delay-response components or procedures, two types of information are provided in the component questionnaires. The first type is subjective or qualitative information that refers to evaluating the generic implementation of delay or response components. This information is somewhat component independent. For example, in implementing a barrier for delay, there are a set of questions that help evaluate how well the barrier has been installed. These questions need not refer to the material of which the barrier is made or how much of a delay the barrier provides. The second type of information is quantitative. It consists of a mean and range of delay times depending upon the type of barrier and type of adversary tools used. Delay-response type components differ from others in this quantitative aspect. With other components and with the "implementation" portion of delay-response components, the basic evaluation concept is one of whether the component "works or doesn't work". With delay and response, however, a "good response" really depends on how much delay is available. Thus a mechanism is needed whereby delay and response times can be compared before an evaluation of a delay-response hierarchy element can be done.

The algorithm provides for such a comparison during the evaluation in the following manner. First, the "qualitative" portion of each delay-response component is scored in the usual manner. Then, at the part of the hierarchy requiring an evaluation of the delay and response, the evaluator is asked to compare the quantitative delay times and response times provided by the system. In theory, this comparison can be done in a variety of ways ranging from using detailed probability models to a subjective preference judgement (Section 3 discusses one way of implementing this comparison using a multiple choice question). The result of the comparison is a score reflecting how well the basic design of the delay-response system works. If actual probabilities could be obtained, this score could be the probability that the delay time is greater than or equal to the response time. From a preference viewpoint, a score between 0 and 1 would be an indication of the utility of the delay-response system relative to some best and worst systems.

The design score or rating described in the last paragraph is then combined with the implementation scores of the delay-response components using one of the possible interaction rules. For example, if the "hard AND" rule were selected, one interpretation could be that if either the implementation of delay or response fails, (e.g. faulty implementation causes the barrier to be essentially ineffective), or the basic design fails because the response forces would not arrive in time, the entire delay-response element fails. A "soft AND" rule could be interpreted as providing for such possibilities as an adversary not realizing that a barrier could be negated by a particular tool or the deterrence value of a response force that may cause an adversary to flee rather than attempt access even though the access might be successful.

For the cases of multiple barriers or response forces, a preference interpretation would allow the combination of implementation scores for the like multiple components via a system questionnaire that would specify an aggregation rule depending upon how well the multiple components coordinated with each other. Sums of delay times could then be compared to response times to arrive at a system design score and the computation would proceed as above. A detailed probability interpretation or computation for multiple barriers becomes fairly complex. This issue is discusssed further in Section 3 of this report.

In summary, the algorithm provides mechanisms for dealing with the delay-response evaluation in a manner that allows for explicit comparison of delay times with response times.

2.5 CAPABILITIES EVALUATION

The previous discussion described the methodology for developing a score for a single capability hierarchy. The disaggregation structure with its five capabilities or objectives directly corresponds to the NRC's Physical Protection Upgrade Rule. As the hierarchies stand, each capability is evaluated independently of the others. Several issues that pertain to evaluating overall safeguards capabilities with respect to this disagregation structure are discussed in the following paragraphs.

Multiple Objectives

In one sense, the structure implicity overdesigns for physical security by evaluating each capability independently of the others. For example, in preventing theft, the only capability that must be achieved in an absolute sense is, "Permit removal of only authorized and confirmed forms and amounts of SSNM from MAAs." Thus, theoretically it might not matter if the facility fails to "prevent unauthorized access of persons and materials into MAAs and VAs" as long as no unauthorized SSNM is permitted to leave the MAA. However, separate evaluation of capabilities implies that the ability to prevent unauthorized access is important independent of the ability to prevent theft. Thus, the five capabilities represent multiple objectives for a safeguards system rather than a single prevent theft (or sabotage) objective. Multiple objectives can address important policy issues. For instance, in order to decrease the amount of risk as perceived by the public, a facility can be evaluated on how well it prevents unauthorized access to the PA and MAA--not only on how well it ultimately prevents theft and sabotage.

The algorithm that has been developed currently provides separate scores only for each of the five capabilities. The particular disaggregation structure was provided as a "given" with the requirement for independent evaluation. Section 5 of this report discusses recommendations for combining capabilities.

Types of Adversary

An adversary may be an outsider or an insider with respect to certain areas of the facility, where "outsider" implies that either stealth or force would have to be used in order to gain access to that area. For example, a guard at the PA boundary would be an insider with respect to the PA, but an outsider with respect to the MAA (i.e., access not permitted). The nature of the adversary will determine which safeguards can first be expected to be needed. For example, an insider with respect to the PA (but not the MAA) whose purpose is theft will have access to the entire protected area. Thus procedures and equipment associated with entry through the PA boundary will be ineffective in stopping an insider with respect to the PA from gaining access to the PA. However, boundary controls at the MAA should be effective, as should procedures and equipment for preventing removal of SSNM from the PA boundary. The importance of the type of adversary and its effect on detection and the timeliness of a response is shown in the adversary path diagram of Figure 2-2.

An aggregation scheme for evaluating overall safeguards can be complicated by consideration of different types of adversaries. For example, to evaluate how well a facility prevents theft by an outsider would seem to require knowing how well it prevented unauthorized access to the PA and MAA. The current disaggregation structure does not make an explicit formal connection between capabilities. Rather, the implication is that of an adversary attempt originating (for all practical purposes) in the particular area addressed by the particular hierarchy with the requirement of preventing the attempt before it succeeds in involving another capability. This condition appears to be reasonable for, say, an MAA insider with respect to moving SSNM. But it becomes awkward to consider the "permit only authorized placement and movement of SSNM within MAAs" hierarchy for the case of an outsider who has arrived there by force. Since the hierarchy treats the attempt as beginning inside the MAA, it is much more natural to view the attempt as coming from an apparent insider and thus have questionnaires tailored to that situation.

The previous discussion serves to describe some of the complexities of safeguards evaluation. It may be desirable to decompose the problem by evaluating separate independent capabilities. This separation may have connotations concerning "overdesigning security"

Figure 2-2. ADVERSARY PATH

Type of Threat: Theft

Physical Path of Adversary	Possible Steps Performed by Adversary
PA boundary	A. Unauthorized access to PA
PA	B. Unauthorized activities in PA
MAA boundary	C. Unauthorized access to MAA
MAA	D. Unauthorized activities (non-SSNM) in MAA
SSNM storage (e.g. vault or working area)	E. Unauthorized placement and movement of SSNM in MAA
	F. Unauthorized removal of SSNM from MAA and facility

Type of Adversary	Points of Detection
1. Outsider	A - F
2. Insider with respect to PA only	B - F
3. Insider with respect to PA and MAA	D - F
4. Insider with respect to PA, MAA, and SSNM	E - F

and "typical threats". While overall safeguards evaluation seems to require combining capabilities, the possible consideration of multiple adversary "modes of attack" (stealth, force, deceit and combinations there of), insider-outsider combinations, adversary tools and pathways can make this a difficult task. Section 5 discusses what types of information can be reasonably expected from a "disaggregation structure-questionnaire" evaluation method in view of the practical constraints on the method's implementation and the complexity of the evaluation.

3.0
IMPLEMENTATION GUIDELINES

3.1 INTRODUCTION

The previous section described the methodology in terms of its capabilities for aggregating safeguard system elements. The algorithms that were developed are flexible enough in terms of the parameters provided to reflect a variety of element interrelationships. This section describes guidelines for the implementation of the methodology in terms of techniques for setting the required parameters. The discussion does not focus on what specific material should be contained in component questionnaires or capability hierarchies. Rather, guidelines are given concerning the general nature of questionnaires and hierarchies and how information in a particular format can be used in a practical manner to assess algorithm parameters. The previous section described the algorithm in a somewhat bottom to top order reflecting the actual computation that would be done. In "starting from scratch," a capabilities hierarchy would be formulated first and probably revised after the lower levels had been defined in a first-cut manner.

3.2 DEVELOPMENT OF HIERARCHIES

The objectives of a safeguard system are defined and the objectives or capability hierarchies developed to indicate how well any particular system achieves those objectives. This kind of evaluation structure is typical of multiobjective evaluation problems. There are some general guidelines for developing these hierarchy structures and these are now discussed.

Typically, hierarchy development proceeds by subdividing a major objective into subobjectives and continuing this process until a fine enough subdivision has occurred so one can evaluate in a specific fashion the lowest level subobjective. The safeguards hierarchies exhibit a natural subdivision of objectives corresponding to different portions of a nuclear facility that continues down until the component level is reached.

Some desirable properties for such hierarchies are the following:

<u>Completeness</u>: All of the essential features of the system are addressed.

<u>Non-Redundancy</u>: The hierarchy should not double-count by aggregating more than once how a system achieves the <u>same</u> subobjective.

<u>Reasonable Size</u>: The hierarchy should not proliferate both vertically and laterally to where features are being examined that could more usefully be lumped together or ignored.

<u>Operational</u>: The structure should have a logical flow from bottom to top so that knowing the bottom levels enables one to proceed easily to evaluate higher levels.

There is not necessarily any unique way of developing a structure for a particular problem. The structure to be used can depend on the evaluation orientation that is desired. In the safeguards evaluation problem, there are at least two orientations that seem useful. One is a fault-tree like orientation from either a facility or an adversary viewpoint. This orientation focuses on sequences that must take place for the safeguard system to succeed or fail in achieving a particular objective. The strength of this approach is in the strong direction it provides in terms of how elements should be aggregated. However, it must be recognized that the nature of the evaluation information may not enable a fault-tree type analysis to be conducted. Analogs to probabilities and conditional probabilities may not be readily obtainable from very qualitative data or it may be too difficult to begin modeling a complex system using probability related computations. A second orientation is one of a checklist of all the features recognized as important to a safeguard system. The strength of this approach is its focus on completeness. Aggregations, however, may need to be done on a more subjective basis since the grouping of elements may not be done with a strong aggregation emphasis in mind. In practice, a hierarchy may contain a blend of both orientations.

3.3 DEVELOPMENT OF QUESTIONNAIRES

A questionnaire can be viewed as a mini-hierarchy that relates fairly specific features that can be assessed by an evaluator to an overall objective of having a component or procedure work as well as possible. As a hierarchy, the same guidelines that were discussed in Section 3.2 apply to questionnaires as well. However, because questionnaires are at the most specific level of the hierarchy, more specific guidelines can be discussed.

Each question on a questionnaire addresses a specific factor related to how well a component works. To make evaluations practical and consistent, each question has a response scale consisting of a set of multiple choices. The questions should be complete in addressing all important factors, non-redundant in not double-counting the same factor effect on the component and minimal in number so that very minor or irrelevant questions are weeded out.

Each specific question response scale should have the following desirable properties:

<u>Comprehensiveness</u>: The score on the scale should adequately reflect the component performance relative to the factor in question. The scale should be applicable in most situations and for most adversary actions.

<u>Operational</u>: The scales should minimize ambiguity by providing a) a sufficient number of possible responses to discriminate between most situations, b) meaningful scale point definitions that include examples for each point on the scale and use specific quantitative units where possible.

The scales may be objective such as "inches of clearance" or subjective such as a series of examples of different features that may be absent or present. Proxy scales may also be used such as "number of drills held per year" as a proxy for response force training.

For algorithm purposes, it is also desirable that response scales be defined so that the following are reasonable approximations:

<u>Linearity</u> of preferences over the scale responses. If two responses are almost equally desirable they should be put as alternatives for the same scale point. Extreme responses that connote an unacceptable facility should not be on a scale but rather should be noted separately for mandatory remedial action.

<u>Utility and Preferential Independence Assumptions Hold (see Appendix A1)</u>. These are assumptions underlying the algorithm computations. In general, if the response scales do not include extreme points, these assumptions are more reasonable.

The previous discussion on hierarchies and questionnaires provide suggestions for developing structures that are amenable to formal evaluation by the methodology. The following paragraphs discuss techniques for calibrating the parameters of the aggregation algorithms.

3.4 ASSESSING WEIGHTS AND AGGREGATION RULES: INTRODUCTION

The aggregation of any group of "elements", in general, requires two types of parameters to be set. For each element a weight (W_i) can be assigned to indicate its relative importance. Then an "interaction coefficient" (V) is assigned to the group as a whole. Ideally, these parameters would be assessed using formal techniques of utility theory or subjective probability. However, due to the large number of assessments to be made, some simplified procedures are described both in the remainder of this section and in Appendix A2.

The general procedure for implementing the algorithm is as follows:

1. Divide elements into groups such that the elements in any particular group can be aggregated using one rule.

2. Assign a weight (this can be a relative weight) to each element.

3. Assign aggregation rules for each group of elements and calibrate each group's evaluation function.

Note that in general, the term "element" above could range from being a question on a questionnaire to a higher level hierarchy box. Techniques for assessing weights and aggregation rules for different elements of the hierarchy will now be discussed.

3.5 ASSESSING WEIGHTS AND AGGREGATION RULES FOR QUESTIONS ON COMPONENT QUESTIONNAIRES

Divide the Questions into Groups.

Sections 3.2 and 3.3 have already discussed how a questionnaire can be viewed as a mini-hierarchy. Conceptually, an objectives structure can be developed using the notions of "checklists" and/or "fault-trees." The groupings of questions for assigning an interaction rule need not exactly correspond to the groupings of questions for checklists, although maintaining two sets of mini-hierarchies can be confusing. The Phase I report [1] contains an example of a fault-tree for a hypothetical component questionnaire. In practice, due to the large number of questionnaires and the prospect of their revision after some trial implementations, all questions can be grouped together in one single group for assigning an interaction coefficient. If the ranges on response scales are not too extreme, this approximation can be a reasonable one.

Assign Weights and an Aggregation Rule.

Conceptually, from the multiattribute utility point of view, all weights and the interaction rule are assessed relative to the best and worst levels of each of the questions on a questionnaire. There are techniques for assessing relative weights for different questions as well as an overall "interaction" constant. [2] The use of typical techniques, however, becomes difficult because of the size of the safeguards assessment problem. There are, on the average, about fifteen questions per questionnaire and on the order of one hundred questionnaires. The questionnaires can conceivably be modified further as the algorithm is tested (see Section 5). Also, because component questionnaires are used as input to higher levels of the hierarchy, keeping in mind the ranges of a myriad of questions when assessing upper level parameters becomes very complex.

To assess parameters in a practical yet systematic manner, techniques to be described shortly have been formulated. They allow relative weights to be assigned quickly and require, one assessment question per questionnaire to consistently link the interaction coefficient with the determination of absolute weights. (When all questions on such questionnaires are assumed to have the same relative weight, this is the only question that needs to be asked to calibrate the entire questionnaire evaluation function.)

The assessment techniques recognize that there are two natural performance level "ranges" to be considered when assessing parameters. The first is the range between the highest and lowest possible responses to a question. The weight W_i must reflect this range for a meaningful parameter calibration. The second is the implicit range that is natural when considering assigning an interaction rule. In this case, one cannot easily keep in mind all the varying question ranges. However, the endpoints of "component is not compromised" (best point) and "component is ineffective" (worst point) provide a somewhat "standardized" range that can be considered in assigning an interaction rule. The assessment techniques to follow connect these two ranges in a logical manner. Question weights are assigned to reflect the relative importance of questions based on their response scale ranges. An interaction rule is assigned based on the best-worst endpoints and is tied into determining the absolute weights for each question.

Assign Relative Weights.

As was discussed earlier, rigorous utility theory techniques for assessing relative weights are not practical for the safeguards problem.

A practical procedure for assigning relative weights to individual questions emphasizes distinguishing between relatively important and unimportant questions. The procedure is as follows:

For each question the following heuristic is posed: What is the maximum possible degradation of component quality that can occur as a result of changing from a maximum to a minimum response to the question?

1. A severe degradation in quality could occur, rendering the component ineffective in performing its function.

2. A moderate degradation in quality could occur, resulting in a likelihood that the component would be ineffective.

3. Only a minor degradation in quality could occur, with the component still likely to function properly.

4. A very minor degradation in quality could occur, with only a minimal effect on component quality.

Note that the weight is assigned on the basis of the range between the highest and lowest possible responses to the question. For example if a question has five possible responses (a-e) the question should be assigned a weight on the basis of the relative desirability of a component with a response of "a" versus one with a response of "e."

The content of this question should be such that influence from other questions or components is ignored. For example, if answers to other questions can aggravate an effect, they should be thought of as being at their best levels. If answers to other questions can mitigate an effect, they should be thought of as being at their worst values.

It remains to assign the relative W_i to be associated with the response to the above heuristic. Reasonable values might be (1) $W_i = 1$, (2) $W_i = .5$, (3) $W_i = .25$, and (4) $W_i = .1$. Note that technically, at this stage, these W_i are relative and could all get multiplied by a constant depending on the interaction rule assessment.

In practice, one might choose to treat all the questions as being of roughly equal importance. This is more reasonable after relatively minor questions have been deleted. In this case, one can proceed directly to the next step, which is assessing the interaction rule and absolute weight.

Assign Interaction Coefficient and Absolute Weights.

A question for determining the interaction coefficient for a group of questions is as follows:

In general, how do the factors interact with one another?

1. They interact in a "strongly interdependent" manner, with a weakness on any one factor negating the strength of the other factors. Or, factors interact destructively, tending to degrade each other's performance.

2. They interact in an "interdependent manner," with weaknesses accumulating to degrade the overall effectiveness of the other factors.

3. They interact in a neutral manner, with the contribution of each individual factor being unaffected by the contributions of others.

4. They interact in a redundant manner, with factors acting as layers of depth of defense, or making up each other's deficiencies.

5. They interact in a strongly redundant manner such that if any one question gets a high score, the group score should also be high.

These five levels have natural interpretations in terms of the constant V. Specifically, we can assign (5) V = -1 (OR gate), (4) V = -1/2 (soft OR gate), (3) V = 0 (average), (2) V = 1 (soft AND gate), and (1) V = + B (AND gate).

Note that in answering this question, there can be a tendency to have a concept of "weakness" in a factor that does not necessarily correspond to the range between best and worst on some of the questions. That is to say, there is an implicit worst point for each factor that makes it intuitively easier to assign an interaction coefficient than to explicitly consider all the best-worst points of each question. To logically connect the explicit and implicit ranges, the following question can be asked:

Given a set of N questions, with n of them having equal W_j, receiving their worst scores, and the other N - n their best, what must n be for the following situations to be about equally preferred:

A		B	
(N - n questions best response, n questions worst response)	<	.5*	All questions best response
		.5	"Ineffective component"

[*This probability can be changed and in general can be set to P where $0 \leq P \leq 1$.]

Here choice B represents a hypothetical 50-50 gamble between getting a component that scores the best possible on all questions and one which is "ineffective." An "ineffective" component is one which receives a score of 0 using the scoring formula, and the best possible component receives a score of 1.

Appendix A2 shows that the absolute weight W (corresponding to W_j) is computed as follows:

$$W = ((1 + V)/V)[1 - (P + (1 - P)(1 + V)^{-N})^{1/n}] \qquad (9)$$

$$-1 < V < \infty, \; V \neq 0$$

For $V = 0$, $W = (1 - P)N/n$; for $V \to \infty$, $W \to 1 - P^{1/n}$; for $V = -1$,

$n = N$ and $W = (1 - P)^{1/N}$

In essence, the calibration above measures how a component is perceived relative to an "ineffective" component given that it scores the worst responses on a certain number of questions. This computation links together the concept of factors making a component ineffective which is used to assign an interaction rule, and the weights W_i that indicate what range of factor "compromise" the questions actually span.

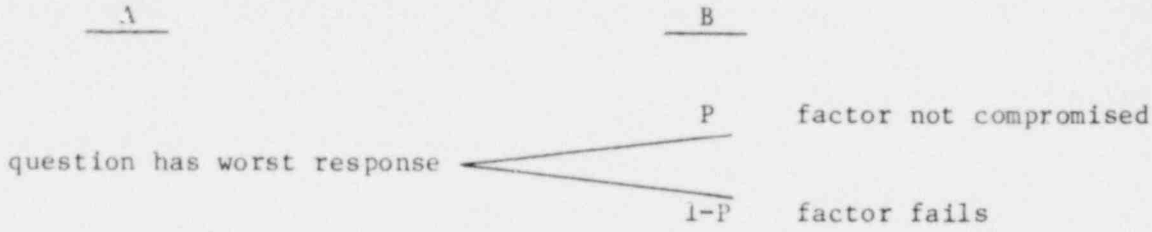
The calibration procedure for W provides a consistency adjustment that connects the original assignment of weights W_i, to the overall interaction coefficient V, so that they are assessed consistently. As an illustration of a typical case of V = 1 (soft AND gate), several absolute W's are shown as a function of n. Note that for N > 5, the formula for W is well approximated by the following:

$$W = 2[1 - .5^{1/n}] \quad \text{when } P = .5$$

n = 1	W = 1	n = 5	W = .26
n = 2	W = .59	n = 7	W = .19
n = 3	W = .41	n = 9	W = .15
n = 4	W = .32	n = 13	W = .10

3.6 INTERPRETATION OF COMPONENT SCORES AND AGGREGATION PARAMETERS

Interpretations for the interaction coefficient have already been presented in Section 2. The interpretations concerning the weights involve interpretations of what is meant by a factor "failing." Having a factor "fail" would ideally be defined by a specific scale point. However, for many factors, such a point is awkward to express in terms of what a facility may be expected to provide. Technically, a W_i represents the probability for which one is indifferent between the following:



Because factor "fails" is hard to explicitly define (though easy to work with implicitly when assigning an interaction rule) and because assessing this lottery is difficult enough without doing it fifteen times per questionnaire, the heuristic mentioned earlier referring to overall component quality is used to assess relative W_i's and an additional assessment is made to assess the absolute W_i's.

The component score that is finally computed is the "indifference" probability that the component is "not compromised" in its performance. The terms in quotations refer to the fact that the evaluation is considered as a judgemental subjective preference assessment.

A perceived "risk" need not be the same as a calculated one, and yet still have validity in terms of its impact on decisions. Similarly, an "ineffective" component is a judgmental term that need not imply the component fails with probability equal to 1, or that the component is effectively non-existent. It simply connotes a perception of ineffectiveness. If one does equate ineffectiveness with worthlessness and indifference probabilities with actual probabilities, then the component score could represent the conditional probability that the component is effective, given the responses to the questions on the questionnaire.

3.7 ASSESSING AGGREGATION RULES FOR HIGHER LEVEL ELEMENTS OF THE HIERARCHY

Weighting Elements

For higher level elements of the hierarchy, the same basic concepts apply as they did for the component questionnaires. For these higher level elements, however, there is currently no provision for assigning an absolute weight other than 1. One reason for assuming a weight equal to 1 is the fact that an implicit normalization has already taken place in deriving the score for upper level boxes. That is, a score of 0 already connotes an ineffective box. To explicitly weight the box again would be difficult because no easy description could be given as to what the new zero point might mean. Implicit weighting actually occurs when boxes are grouped. As an analogy, consider the case where one component (say tamper protection) feeds into another component (say a sensor). All questions have weights equal to 1 and we soft AND to compute the scores for both components. If the sensor component has one response at its worst level and all others at their best, it would receive a score of about .5 (when N>5). However, if one tamper protection response was at its worst level, the tamper protection element would receive a score of .5, but the sensor would receive a score of .75 if all other responses besides tamper protection were at their best. Thus a tamper protection question implicitly has less effect on the sensor score than a sensor question: This illustrates how the grouping of elements can implicitly give less weight to those factors that affect only a relatively narrow segment of the overall hierarchy. In essence, relative weighting is automatically taking place via grouping of the elements.

Assigning Interaction Rules - System Questionnaires

To assign the interaction coefficient for a group, the same heuristic discussed in section 3.5 can be used. Another approach to assigning an interaction coefficient is that of the system questionnaire discussed in section 2.3. An example of this concept is shown in Figure 3-1. Instead of using a very general heuristic such as that in section 3.5, an evaluator answers more specific questions such as those shown in Figure 3-1. The system questionnaire in Figure 3-1 is treated just like a component questionnaire. Weights can be assigned and a rule specified for combining question scores. As section 2.3 and Appendix A1 explain, the overall score is used to specify the rule for combining multiple sensors. A high score means that the multiple sensors are redundant and provide defense in depth. A low score means that one poor sensor can negate the strengths of the others.

Multiple Access Points and Delay-Response

Section 2.4 discussed the issue of multiple access points. For implementation purposes, it is simplest to aggregate such points at the lowest possible level of the hierarchy.* For example, if there are several possible entry points, each presenting a barrier to adversary penetration, it is simplest to aggregate all the barriers using some rule rather than to analyze each barrier-access point individually up to the highest level of the hierarchy and then combine access points at the highest level. Aside from proliferating hierarchy elements, this latter approach tries to define the entire pathway that an adversary might take in trying to gain access to a facility. The algorithm and questionnaires were not intended for such detailed system modeling.

---

*NOTE: It is the Sandia authors' view that aggregating each low-level task over all access points and subsequently combining the resultant measures at each level of the hierarchy fails to reflect the essential sequence of events for detection and response functions and fails to identify the locations where these functions are of concern. A test program should help resolve the question of correctness vs. practicality.

Figure 3-1 System Questionnaire Example: Multiple Sensors

1. Will each sensor type be selected to minimize the susceptibility of any two or more sensor types to the same local environmental (natural or man-made) source of nuisance alarms?

2. Will each sensor type be selected to minimize the likelihood that two or more sensor types will be affected by the simultaneous occurrence of environmental (natural or man-made) sources of nuisance alarms, e.g., wind and rain?

3. What provisions will be made to minimize the likelihood of false or nuisance alarms?

4. Will collocated sensors be installed to provide mutual tamper protection for the sensors and processors?

5. Will collocated sensors be selected to provide coverage over a wide range of intrusion methods, (e.g., microwave to sense surface intrusion and buried cable to sense tunneling or crawling under the microwave beam or balanced magnetic switch to sense door opening and breakwire system to sense cutting through the door)?

6. Will collocated sensors be selected to minimize operational performance incompatabilities?

Figure 3-2 Example Question to Aid in Assessing a Delay-Response Type Score

Direct or Indirect Monitoring

Using data from the questionnaires pertaining to the barrier(s) and the type of monitoring that will be used, how will the adversary boundary penetration and/or introduction of materials time compare with time between monitoring observations?

a. Adversary penetration and/or introduction of material time will exceed twice the time between observations
b. Adversary penetration and/or introduction of material time will be less than twice but greater than the time between observations
c. Adversary penetration and/or introduction of material time will be equal to th time between observations
d. Adversary penetration and/or introduction of material time will be less than the time between observations

Similarly, it is complex to do separate delay-response type analyses of every pathway an adversary might pursue. A more subjective evaluation of delay-response is possible as described in section 2.4. For implementation purposes, a multiple-choice question such as the one shown in Figure 3-2 can aid in assessing a delay-response score that considers multiple barriers/access points. (This question can have a weight.) Alternatively, the delay-response score can be directly assigned using whatever means is appropriate. (See those suggested in section 2.3.)

Questionnaires Input to Other Questionnaires

The issue of questionnaires being used as input to other questionnaires has already been discussed in terms of component questionnaires (e.g. tamper protection input to sensors) and system questionnaires (components combined with other information, such as described in section 2.3). Two effects occur when this is done. First, as was described in the beginning of this section, the questions on questionnaires feeding into others generally have implicitly much less weight, Second, a questionnaire is always the lowest level the hierarchy algorithm can consider. A systems questionnaire must be completed manually with other component scores being input manually (after a first pass computation). If sensitivity analysis is desired on input-type questionnaire responses, a manual update must be done before the algorithm can be run. Furthermore, in order to input one questionnaire into another, the score of the input questionnaire must be discretized to be in the proper multiple choice format. Because of these effects, we recommend the following:

- Eliminate as much as possible all input-type questionnaire situations. For example, tamper protection has a small influence on the scores of components to which it is input. One question with many multiple choice responses should be used to replace the entire tamper protection questionnaire, or else it should be treated as a component that does not feed into another questionnaire.*

- Compose system questionnaires so that they only determine the way in which components combine to perform a function. When other information is mixed in on a system questionnaire, it makes the analysis complex and also "cuts off" the algorithm from tracing results down to components below the system questionnaire level.

Sections 3.6 and 3.7 have described techniques for calibrating the parameters of the algorithm. The following subsections describe the basic steps for exercising the algorithm, interpreting the evaluation results and performing sensitivity analysis.

3.8 INSPECTION (DESIGN) STEPS - DATA REQUIREMENTS

The steps performed in exercising the algorithm are now summarized. The assumptions are that a capability hierarchy is defined and all component and system questionnaires required for the analysis have been developed. In addition, all the fixed interaction coefficients for hierarchy boxes and questionnaires and the weights for questions are assumed to be assigned.

Step 1. Identify all the components to be used for accomplishing low-level system tasks (performance characteristics) using the component selection matrices.

---

*NOTE: The opposing view taken by the Sandia authors is that to evaluate such components as tamper protection, emergency power supplies, etc. independently and then combine results with those from components such as sensors, CCTV, etc. effectively places equal importance on tamper protection as on the combined effect of all the other sensor performance factors.

Step 2. Answer all of the questionnaires for the components above. (Questionnaires that feed into others need to be scored first before this can be done. See Step 5.)

Step 3. Using the component selection matrices, identify those components that are used to perform the same low level system task. Answer any system questionnaire required to indicate how well these components coordinate with each other.

Step 4. Compose computer input. (See Section 4 for details.) Select questionnaire input for those components involved and compose File 1. Organize questionnaire responses in File 2. Indicate which questionnaires feed into which hierarchy elements in File 3.

Step 5. Use the algorithm to <u>score</u> all questionnaires.

Step 6. Assign scores to delay-response hierarchy boxes if they do not have questionnaires determining their scores. (Note whether the same barriers are being used both to help detect and/or assess as well as delay. If so, the delay-response score should take into account the fact that the barrier is doing "double or triple" duty.)

Step 7. Score capability hierarchy.

Step 8. Evaluate results and perform sensitivity analysis.

A capability is scored for a single set of evaluator responses. (Multiple evaluators are not currently handled by the algorithm in terms of such possibilities as averaging evaluator scores or consistency checking them.) The following paragraphs discuss Step 8 in more detail.

3.9 EVALUATION OF RESULTS - SENSITIVITY ANALYSIS

Once a capability is evaluated, the results can be interpreted in a comparative sense. As an illustration of this, suppose the algorithm is used to evaluate systems A and B and system A scores higher than B. The implication is that system A is preferred to system B. If system A is considered a barely acceptable facility, then system B might be considered unacceptable. Whatever score system A received is then considered a threshhold score.

Another interpretation can be given in terms of an "ideal" system versus an "ineffective" system. Theoretically, a facility receiving a score of .5 on a capability is considered equally preferred to a 50-50 gamble between an ideal system getting perfect responses on all questionnaires and an "ineffective system" receiving an overall score of 0. (A system receiving the worst response to all questionnaires will probably have a score very close to 0). A score below .5 indicates that one is willing to take such a gamble. This does not indicate much faith in the facility to perform the capability. On the other hand, a score of above .5 indicates that at least one would rather stay with such a facility than take an even chance at "perfecting it" versus losing its effectiveness. (This same interpretation can be applied to individual components and lower level boxes.)

The computer program described in section 4 provides the routines for tracing the capability score computation down through lower level boxes all the way to the component level and to questionnaire responses. It provides displays that help to pinpoint particularly low scores, or situations where many elements are "ANDing" together to produce a lower score than may be desired. The designer or inspector can then see whether improving a component, adding a component or improving the way components coordinate can help improve the facility score.

Sensitivity Analysis

The computer program enables the user to change the following parameters to see how the results are affected:

- questionnaire weights and responses
- all aggregation rules
- scores for any box or questionnaire

With these features, a user can examine what "improvements" will cause one facility to be at least as preferred as another. Changing weights and aggregation rules also allows the evaluation to span a range of relatively more conservative assumptions (e.g. larger weights and more AND operators) to relatively less conservative assumptions (e.g. smaller weights and fewer AND operators). In this way, the algorithm is a useful tool that can help analyze what set of assumptions cause a facility to be evaluated as relatively acceptable or not.

In summary, section 3 has discussed how the algorithm can be implemented and used. Section 4 describes the computer program that facilitates this implementation. (Hand calculations and programmable calculators can be used to compute scores for individual elements since the scoring algorithms are straightforward. But the computer program greatly facilitates the handling of hierarchies and large numbers of questionnaires). An example presented in section 4.3 illustrates the algorithm implementation.

4.0 SAFEGUARDS EVALUATION COMPUTER PROGRAM

4.1 INTRODUCTION

To implement the algorithm described in previous sections, an evaluation computer program has been developed. This program is designed to automate the scoring of evaluation questionnaires and hierarchy elements and to provide maximum flexibility to the user for sensitivity analysis and other changes.

The program uses two basic types of input. The first type provides the structure of the questionnaires and hierarchies, including the number of questions (or inputs to a hierarchy element) weights and the scoring rules to be used. This data is independent of any particular evaluation and can be developed and stored on the computer before an evaluation is done. The second type of input is the responses to the questionnaires. These are the answers to the questionnaires filled out by an evaluator.

To compute the score for a hierarchy, the program first looks at questionnaires. The questionnaire structure (number of questions, weights, lowest values for each question, etc.) is read from one disk file and responses are read off another. The routine then automatically computes and saves the questionnaire score. After the questionnaires have been scored the program can be switched into hierarchy mode. To score a hierarchy element (box) its name is entered. If the scores for all the boxes leading into the box have been computed it scores the box using the appropriate rule. If not, the program attempts to score lower level boxes, gradually working down in the hierarchy until a box whose score can be computed is found. The program then works back up the hierarchy until the original box's score can be computed. Low level boxes (with component questionnaires) are scored in the same way except that the program assumes that all questionnaires have been scored.

The rest of this chapter describes the structure and operation of the evaluation computer program in more detail. Section 4.2 describes the use of the program and the format of the associated input files. Section 4.3 provides an example of the program's use. Listings of the program are presented in Appendix A3.

4.2 USE OF THE EVALUATION PROGRAM

This section describes how the evaluation computer program can be used to evaluate questionnaires and hierarchies. First the data base is described in detail, then the operation of the program, including the various options available and the flow of the program, is described. The use of the program is demonstrated in section 4.3.

DATA BASE

The input to the program consists of four "files" (sets of data stored on cards or disk):

1. Questionnaire structures
2. Questionnaire responses
3. Hierarchy structure
4. Hierarchy initial scores

The content and format of these files is described in more detail below. The program is written in FORTRAN and FORTRAN formats are listed where appropriate.

Questionnaire Structures

This file contains information on the questionnaires to be evaluated. The data for each questionnaire includes:

- Name
- Number of questions
- Number of subgroupings (if any)
- Weight for each question
- Lowest possible response for each question
- Scoring rules for overall group and subgroups

The specific layout is as follows:

Card (record) 1: Number of questionnaires in file (1I2 format)

Card 2: First card number for each questionnaire (i.e. the number of the card where the questionnaire starts). (20I4 format) This card is repeated as necessary to specify the first record of all questionnaires in the file.

Card 3: Questionnaire title. Contains the questionnaire name (maximum of four characters) the number of questions (maximum of 40) and the number of subgroups (counting the overall questionnaire as 1). The inital implementation of the algorithm will not use subgoups, but the program has the ability to process them. Format (1A4,6X,1I2,8X,1I2).

Card 4: Worst Response. The letter corresponding to the worst response is given for each question. (The best response is always assumed to be "A".) (40(1X,1A1)).

Card 5: Group Information. Group number. (The group number for the overall questionnaire is always 50). Additional groups are numbered 51,52...etc. The number of questions (and subgroups) to be aggregated and the rule to be used is also given. The codes for rules are HA = hard AND, SA = soft AND, AV = average, SO = soft OR, OR = hard OR. Format (1I2,8X,1I2,8X,1A2).

Card 6: Group Inputs. The questions (or subgroups) to be aggregated as part of the group are given in (40I2) format.

Card 7: Question Weights. contains the weight (between 0 and 1) assigned to each question. Initially the questions will be equally weighted at 0.5, but the program can accept differential weights. (8F5.3) This card is repeated until a weight is specified for each question. A convenience option allows one weight for all questions to be set by specifying 2. as the first "weight" and the assigned weight for all as the second weight.

Cards 5 and 6 are repeated for each group.
Cards 3 through 7 are repeated for each questionnaire.

Questionnaire Reponses

This file contains the responses to the various questionnaires (the results of the evaluation). The format is as follows:

Card 1: The number of questionnaires evaluated. (1I2) format.

Card 2: The first card number for each questionnaire. (20I4) format. The questionnaires must be in the same order as in the questionnaire structures file. This card is repeated as many times as necessary to identify the first record for each questionnaire.

Card 3: The name of a questionnaire. (1A4) format

Card 4: The score for each question on the questionnaire. In alphabetic format (40(1X,1A1)).

Cards 3 and 4 are repeated for each questionnaire.

Hierarchy Structures

This file contains structural data about the organization and scoring of hierarchies. The format is as follows:

Card 1: The number of complete hierarchies in the file. Format (1I2), maximum value = 5. (Typically, there is only one hierarchy in a file)

Card 2: The first card number for each hierarchy (5I4) format.

Card 3: Box Data Card. This card includes the name of a box, the number of subelements to be aggregated and the scoring rule to be used. If the elements to be aggregated are questionnaires instead of boxes then 50 is added to the number of subelements. If a questionnaire is to be used to determine the scoring rule the questionnaire name also appears on the card. The order is: box name, number of elements, rule, questionnaire name (if any). The format is (1A6,4X,1I2,8X,1A2,8X,1A4).

Card 4: Input Box Data Card. This card contains the name of an input subelement (box or questionnaire) in (1A6) format.

Card 4 is repeated for each input subelement. Cards 3 and 4 are repeated for each hierarchy box having sub-elements. The only restriction of the ordering of the boxes is that a box name must not appear on a number 4 card after it has appeared on a number 3 card, (i.e, go from top to bottom).

Card 5: The last card for each hierarchy is a card with the word "NOMORE" in the first six columns.

Hierarchy Initial Scores

This file contains values for any initial scores to be set for hierarchy boxes. The file is structured as follows:

Card 1: The number of hierarchies in (1I2) format. (Typically, this number is 1)

Card 2: The initial card for each hierarchy in (5I4) format.

Card 3: The names of boxes to be set followed by the initial score. If the score is set at -1, the initial score is free. (Otherwise scores must be between 0 and 1). There are no restrictions on the order of the boxes. If a box does not appear its initial score is assumed to be -1. The format is (5(1A6, 1F6.3)).

Card 3 is repeated until all set scores have been read in.

Card 4: Questionnaire scores in (10F8.5) format. Ten cards (they may all be blank) are required for each hierarchy.

This file is used primarily by the program to store the results of an evaluation so that they need not all be computed each time the program is run. Initially, the file can be set with the first two cards specified as above, and 18 blank lines following the first two cards.

INTERACTIVE PROGRAM OPERATION

Questionnaires and hierarchy elements are evaluated using an interactive computer program. This program uses the data files described in the previous section as input and provides the user with a wide variety of evaluation and sensitivity analysis options. The following paragraphs describe the relationship of the program elements and data files and the options available to the user.

Input/Output Considerations

The evaluation program is designed to be used interactively at a timesharing terminal. In addition, four disk storage files are needed. These files were described previously. They interface with the program as shown in Table 4-1.

Program Operation: General Features

When the evaluation program is called, it first initializes the major variables and then prompts the user with the following question:

```
SELECT        1-HIERARCHIES     2-QUESTIONNAIRES     3-STOP --
```

Typing "1" in response to this question initiates the hierarchy manipulation portion of the program. A second list of options will be printed to allow the user to control the manipulation. These options are described shortly. Similarly, if the user responds with "2", a set of options relating to questionnaires is printed. Typing "3" stops the program. If the user is familiar with the program options described

Table 4-1. DATA BASE DEFINITIONS

File	Unit	Type	Record Length (Characters)	Maximum Number of Records
1. Questionnaire Structures	1	Random Access	80	300
2. Questionnaire Responses	2	Random Access	80	150
3. Hierarchy Structure	3	Random Access	80	200
4. Hierarchy Scores	4	Random Access	80	50

below, any valid option number can be typed and the program will branch directly to that option.

Program Operation: Questionnaire Manipulation

Selecting the questionnaire option causes the following table to be printed.

```
SELECT ONE:

21-COMPUTE SCORES              22-PRINT SCORES
23-SET SCORES                  24-REVISE WEIGHTS
25-REVISE RULES                26-REVISE RESPONSES
            29-NO MORE REVISIONS
WHICH?
```

The user simply types in the number corresponding to the desired option and the computer will initiate the option and ask additional questions to enable its completion. The options are described in more detail below.

Option 21 - Compute Scores. This option computes the score for a questionnaire. When the option is selected the prompt "ENTER QUESTIONNAIRE NAME -- " is given. If the name is valid, the questionnaire's information is retrieved from the questionnaire structure and response files and the score is printed and stored. If "ALL" is typed in response to the name prompt, all the currently stored questionnaires are scored and printed. The user is then asked to select another option.

Option 22 - Print Scores. This option prints the data associated with a questionnaire. A name is entered as in Option 21 and the computer prints a table of information for the questionnaire. The information includes the scoring rule and score and a diagram of the questionnaire structure. The structure shows the subgroups (if any) used in scoring the questionnaire, the scoring rules used for the subgroups and the individual questions included in each group along with their associated raw scores, weights and adjusted scores.

Option 23 - Set Scores. This option allows the user to directly specify a score for a questionnaire. In response to a prompt the user enters a questionnaire name. The prompt "SCORE =" is printed and the user may enter any value between 0 and 1.0. This score is saved until the score is recomputed or reset.

Option 24 - Revise Weights. The option allows the user to revise the weight assigned to a given question or questions. After the questionnaire name is entered, the prompt "NUMBER OF QUESTIONS TO BE REVISED =" is given. If the weight has been assigned using the brief form, the common weight assigned to all questions must be revised. For each question to be revised, the prompts "QUESTION NUMBER =" and "WEIGHT =" allow the new weight to be assigned to the appropriate question. After this option is completed the score is recomputed and printed.

Option 25 - Revise Rules. This option allows the user to revise the scoring rule used to score a questionnaire or subgroup. After the questionnaire name is entered, the computer asks for the "group number" to be changed. Group "50" corresponds to the overall questionnaire and 51, 52, etc., correspond to the subgroups (if any). Next the revised rule is requested, using the following abbreviations:

HA-Hard AND
SA-Soft AND
AV-AVERAGE
SO-Soft OR
OR- OR

The revised score is computed after the desired number of changes has been made.

Option 26 - Revise Responses. This option allows the user to revise the responses associated with particular questions. The procedure is similar to that for revising weights, in that the questionnaire name and number of questions to be revised initializes a loop for entering revised responses. For each question, a prompt asks for the question number and then the user is prompted "ENTER REVISED RESPONSE (A to WORST) --". WORST is the letter of the alphabet corresponding to the worst answer on the question. The user enters the letter of the alphabet corresponding to the revised response. After all desired changes have been completed the questionnaire score is recomputed and printed.

Option 29 - No More Revisions. This simply returns the program to the original hierarchy/questionnaire/stop choice. These options represent all of the interactive routines relating to questionnaires. Other changes (e.g., revisions to questionnaire structure) must be made using a text editor on the appropriate files.

Program Operation: Hierarchy Manipulation

When the hierarchy manipulation option of the program is first initiated, the computer requests "ENTER HIERARCHY NUMBER --". The user enters the number of the hierarchy to be manipulated in the current session. The computer then retrieves the data corresponding to that hierarchy from the disk files. Next the following table is printed:

SELECT ONE:

41-COMPUTE SCORES	[illegible]-PRING DATA
43-ASSIGN SCORES	44-REVISE DELAY/RESP
45-REVISE RULES	46-SELECT NEW HIERARCHY
47-PRINT BOX NAMES	48-FILE HIERARCHY DATA
49-CHANGE BOX NAMES	50-PRINT HIERARCHY

51-NO MORE REVISIONS

WHICH?

Typing the number corresponding to an option initiates the option. As described below, the computer asks additional questions as necessary to allow completion of the option.

Option 41 - Compute Scores. This option allows the user to compute the score for a hierarchy box. Of course, if the top box of the hierarchy is scored, the overall score will be computed. After the box name is requested and entered, the computer automatically searches as far down in the hierarchy as is necessary (up to a maximum of five levels) to identify boxes which can be scored, (i.e., boxes for which scores are available for each lower level box or questionnaire). Then the computer works back up the hierarchy, scoring higher level boxes until it is possible to compute the score for the requested box. This score is printed. (Note: The scores for all higher level boxes are reinitialized to -1 when a lower level score has been changed.)

Option 42 - Print Data. This option allows the user to obtain a simplified diagram of the hierarchy structure beneath a specified box. Up to four levels of boxes are printed. The information for each box includes the box name, its score, (-1 is shown if the score has not been computed) and scoring rule and scoring questionnaire (if any). The table is printed in outline style, with lower level boxes being indented beneath higher level boxes.

Option 43 - Assign Scores. This option allows the user to assign a score to a specified box. The computer first prompts for the box name and then requests the score, which must be between 0.0 and 1.0. The scores for all higher level boxes are reinitialized to show that a lower level score has been changed.

Option 44 - Revise Delay/Response. This option is not used at the current time.

Option 45 - Revise Scoring Rule. This option allows the user to change the scoring rule associated with a box. The computer first requests the box name and then the rule. The rule is entered using the following abbreviations.

HA-Hard AND
SA-Soft AND
AV-Average
SO-Soft OR
OR-OR
Q -Scoring rule determined by questionnaire

If Q (Questionnaire Scoring) is entered, the computer will prompt for the questionnaire name.

Option 46 - Select New Hierarchy. This option reinitializes the program by allowing the user to reenter the data for the current hierarchy or another stored in Disk File 3. The only prompt is "ENTER HIERARCHY NUMBER".

Option 47 - Print Box Names. This option causes a list of the current box names to be printed.

Option 48 - File Hierarchy Data. This option saves all revisions and scores for the hierarchy (including questionnaire scores) during the current session on Disk File 3 and 4 respectively. The original data is overwritten. This option is done automatically at the termination of a session if Options 45 or 49 have been used.

Option 49 - Change Box Name. The computer first prompts for the original box name and then for a revised name. Names are allowed to be a maximum of 6 characters long.

Option 50 - Print Hierarchy. This option is similar to option 42 except that the structure is printed in a more easy to follow graphical form. One to five hierarchy levels are printed starting with a box name entered with the computer prompt. Warning: If you are in a hurry or conserving paper it is best to use Option 42 for viewing hierarchy data.

Option 5_ - No More Revisions. This option reverts the program back to the ori;inal questionnaire/hierarchy/stop choice.

4.3 E:AMPLE OF ALGORITHM AND COMPUTER PROGRAM USE

To illustrate the algorithm and computer program, hypothetical data wa: provided by Sandia concerning the capability of "prevent unauthorized access of persons and materials into the MAA" (see Figure 1-1(1)). A set of component questionnaires was filled out corresponding to several, but not all of the low-level system tasks. The weight on each question in every component questionnaire was set equal to .5. Each component's score was computed by using a soft AND operator on the question scores. (Some questionnaires that were input to others were scored in a "first pass".) A total of 38 questionnaires appear directly in the computer program input data which can be found in Appendix A4.

Interaction coefficients were assigned for all hierarchy boxes requiring them. All boxes without appropriate component questionnaire input were assigned arbitrary scores (e.g., a 1 in most cases).

The example is developed in terms of the commands that would be issued b a user to the computer and the resulting output. Appendix A4 contains computer input and additional computer output for the entire capability. In this example, however, the focus is on that segment of the capability hierarchy concerned with "detect access/ introduction of material through remainder of area boundary." (This is the only segment of the example with reasonably complete questionnaire input data.)

Each questionnaire was given an identifying number and each hierarchy box was given a mnemonic identifier. Appendix A4 lists the component corresponding to each questionnaire number. The mnemonics for this example are shown in Figure 4-1.

In exercising the algorithm, the first command to the computer should be to read in the hierarchy structure (and any initializing information). This step is shown in Figure 4-2. The next step is to score all the questionnaires (if such scores have not been computed and stored previously). The appropriate option for this is 21. Figure 4-3 shows how all the questionnaire scores can be computed. Figure 4-4 shows an example of how a more detailed printout for the annunciator systems component questionnaire can be displayed.

The next step is to evaluate the capability hierarchy. (First, boxes requiring assigned scores should be given these scores using option 43.) The computer is capable of evaluating up to five hierarchy levels down. Therefore, lower level boxes must be evaluated first if the top box has more than five levels beneath it. In this example, the right side of Figure 1-1(1), "Deny Access" (DENACC) can be evaluated directly using option 41. One form of computer display of the hierarchy results is shown in Figure 4-5. By using a series of display commands, the user can focus on different segments of the hierarchy. For example, in Figure 4-5, the DETACC display shows a trace of the scoring down to the questionnaire level for some boxes. The second page of Figure 4-5 illustrates how one can obtain a more detailed look at lower level boxes such as INDM1.

A second way of displaying hierarchy data is shown in Figure 4-6. This type of display does not contain questionnaire scores explicitly, but does give a pictorial view of hierarchy relationships. Figure

Figure 4-1. MNEMONICS FOR ALGORITHM EXAMPLE

DETACC - Detect Access

SENSE - Sense Attempt

REPALR - Report Alarm

ASSESS - Assess Alarm

MULTS - Multiple Components for Sensing Attempts

INDIM1 - Indirect Monitoring to Sense Attempts

TSIG - Transmit Signal

ANALRM - Anunciate Alarm

MULTA - Single or Multiple Componer s for Assessing Alarms

INDA1 - Indirect Assessment of Al s

CASSAS - Central and Secondary Alarm Stations Score

DDS - Delay/Detection Score (based on time comparisons)

GP - Guard Patrol Score

BARR - Barriers (multiple access points)

DDA - Delay/Assessment Score (based on time comparisons)

ALAS - Alarm Assessment System Questionnaire (not actually used in capability evaluation)

PNSS - Multiple Sensor Penetration Sensing System Questionnaire (not actually used in capability evaluation).

Figure 4-2. READING IN THE HIERARCHY STRUCTURE

```
EXECUTION:
 SELECT 1- HIERARCHIES, 2- QUESTIONAIRES, 3- STOP -- 1
 ENTER HIERARCHY NUMBER -- 1

 SELECT ONE:

 41- COMPUTE SCORES        42- PRINT DATA
 43- ASSIGN SCORES         44- REVISE DELAY/RESP
 45- REVISE RULES          46- SELECT NEW HIERARCHY
 47- PRINT BOX NAMES       48- FILE HIERARCHY DATA
 49- CHANGE BOX NAME       50- PRINT HIERARCHY
               51- NO MORE REVISIONS
```

Figure 4-3. COMPUTING QUESTIONNAIRE SCORES

```
SELECT 41-51 -- 21
ENTER QUESTIONAIRE NAME -- ALL
QUESTIONNAIRE 4   : THE SCORE = 0.755
QUESTIONNAIRE 6   : THE SCORE = 0.766
QUESTIONNAIRE 10  : THE SCORE = 0.670
QUESTIONNAIRE 47  : THE SCORE = 0.820
QUESTIONNAIRE 57  : THE SCORE = 0.579
QUESTIONNAIRE 1   : THE SCORE = 1.000
QUESTIONNAIRE 2   : THE SCORE = 0.917
QUESTIONNAIRE 3   : THE SCORE = 0.606
QUESTIONNAIRE 11  : THE SCORE = 0.820
QUESTIONNAIRE 14  : THE SCORE = 1.000
QUESTIONNAIRE 16  : THE SCORE = 1.000
QUESTIONNAIRE 21  : THE SCORE = 0.911
QUESTIONNAIRE 22  : THE SCORE = 0.237
QUESTIONNAIRE 25  : THE SCORE = 0.562
QUESTIONNAIRE 28  : THE SCORE = 0.516
QUESTIONNAIRE 32  : THE SCORE = 0.750
QUESTIONNAIRE 36  : THE SCORE = 0.875
QUESTIONNAIRE 38  : THE SCORE = 1.000
QUESTIONNAIRE 43  : THE SCORE = 1.000
QUESTIONNAIRE 51  : THE SCORE = 0.766
QUESTIONNAIRE 60  : THE SCORE = 0.516
QUESTIONNAIRE 63  : THE SCORE = 0.387
QUESTIONNAIRE 66  : THE SCORE = 1.000
QUESTIONNAIRE 68  : THE SCORE = 1.000
QUESTIONNAIRE 69  : THE SCORE = 0.548
QUESTIONNAIRE 74  : THE SCORE = 1.000
QUESTIONNAIRE 75  : THE SCORE = 1.000
QUESTIONNAIRE 83  : THE SCORE = 0.746
QUESTIONNAIRE 84  : THE SCORE = 0.637
QUESTIONNAIRE 87  : THE SCORE = 0.733
QUESTIONNAIRE 90  : THE SCORE = 0.667
QUESTIONNAIRE 95  : THE SCORE = 0.337
QUESTIONNAIRE 12  : THE SCORE = 0.338
QUESTIONNAIRE 33  : THE SCORE = 1.000
QUESTIONNAIRE ALAS: THE SCORE = 0.598
QUESTIONNAIRE PNSS: THE SCORE = 1.000
QUESTIONNAIRE 17  : THE SCORE = 1.000
QUESTIONNAIRE 18  : THE SCORE = 1.000
```

Figure 4-4. DISPLAYING QUESTIONNAIRES

```
SELECT 21-29 -- 22
ENTER QUESTIONAIRE NAME -- 4

                    QUESTIONAIRE DATA FOR
                      QUESTIONAIRE 4
      OVERALL SCORE =     0.75493      RULE : SA

BOX: 50  RULE: SA
      Q= 1  RESP= 1.000 W= 0.500 S= 1.000
      Q= 2  RESP= 0.667 W= 0.500 S= 0.833
      Q= 3  RESP= 1.000 W= 0.500 S= 1.000
      Q= 4  RESP= 1.000 W= 0.500 S= 1.000
      Q= 5  RESP= 1.000 W= 0.500 S= 1.000
      Q= 6  RESP= 1.000 W= 0.500 S= 1.000
      Q= 7  RESP= 1.000 W= 0.500 S= 1.000
      Q= 8  RESP= 1.000 W= 0.500 S= 1.000
      Q= 9  RESP= 1.000 W= 0.500 S= 1.000
      Q=10  RESP= 1.000 W= 0.500 S= 1.000
      Q=11  RESP= 1.000 W= 0.500 S= 1.000
      Q=12  RESP= 1.000 W= 0.500 S= 1.000
      Q=13  RESP= 0.833 W= 0.500 S= 0.917
      Q=14  RESP= 1.000 W= 0.500 S= 1.000
      Q=15  RESP= 0.667 W= 0.500 S= 0.833
      Q=16  RESP= 0.750 W= 0.500 S= 0.875
```

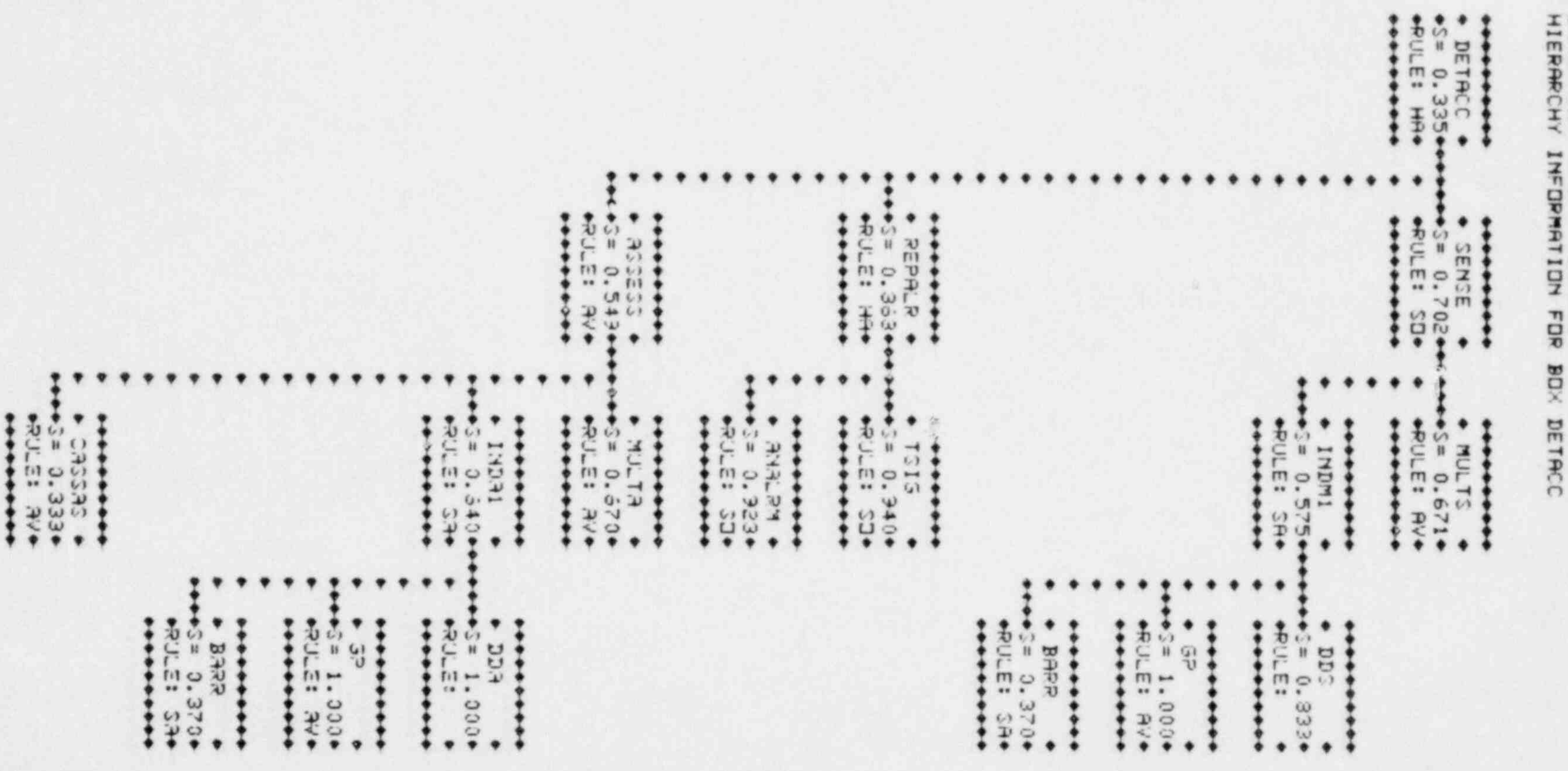
Figure 4-5. HIERARCHY DISPLAY

```
 WHICH? 42
 ENTER BOX NAME -- DENACC

 HIERARCHY DATA FOR BOX DENACC

BOX:DENACC  RULE:SA  SCORE: 0.436   Q:
      BOX:DETACC  RULE:HA  SCORE: 0.335   Q:
            BOX:SENSE   RULE:SD  SCORE: 0.702   Q:
                  BOX:MULTS   RULE:AV  SCORE: 0.671   Q:
                  BOX:INDM1   RULE:SA  SCORE: 0.575   Q:
            BOX:REPALR  RULE:HA  SCORE: 0.868   Q:
                  BOX:TSIG    RULE:SD  SCORE: 0.940   Q:
                  BOX:ANALRM  RULE:SD  SCORE: 0.923   Q:
            BOX:ASSESS  RULE:AV  SCORE: 0.549   Q:
                  BOX:MULTA   RULE:AV  SCORE: 0.670   Q:
                  BOX:INDA1   RULE:SA  SCORE: 0.640   Q:
                  BOX:CASSAS  RULE:AV  SCORE: 0.338   Q:
      BOX:RESACC  RULE:HA  SCORE: 0.730   Q:
            BOX:COMRSP  RULE:SA  SCORE: 1.000   Q:
                  BOX:BETGDS  RULE:    SCORE: 1.000   Q:
                  BOX:GDSSTN  RULE:HA  SCORE: 1.000   Q:
                  BOX:BETSTN  RULE:    SCORE: 1.000   Q:
                  BOX:ONOFF   RULE:    SCORE: 1.000   Q:
            BOX:RESP    RULE:SA  SCORE: 0.730   Q:
                  BOX:DELRSP  RULE:SD  SCORE: 0.790   Q:
                  BOX:EFFRSP  RULE:SA  SCORE: 0.706   Q:
                  BOX:DRRSP   RULE:    SCORE: 1.000   Q:
 SELECT 41-51 -- 42
 ENTER BOX NAME -- DETACC

 HIERARCHY DATA FOR BOX DETACC

BOX:DETACC  RULE:HA  SCORE: 0.335   Q:
      BOX:SENSE   RULE:SD  SCORE: 0.702   Q:
            BOX:MULTS   RULE:AV  SCORE: 0.671   Q:
                   QUESTIONNAIRE: 6     SCORE: 0.766
                   QUESTIONNAIRE: 57    SCORE: 0.579
                   QUESTIONNAIRE: 10    SCORE: 0.670
            BOX:INDM1   RULE:SA  SCORE: 0.575   Q:
                  BOX:DDS     RULE:    SCORE: 0.833   Q:
                  BOX:GP      RULE:AV  SCORE: 1.000   Q:
                  BOX:BARR    RULE:SA  SCORE: 0.370   Q:
      BOX:REPALR  RULE:HA  SCORE: 0.868   Q:
            BOX:TSIG    RULE:SD  SCORE: 0.940   Q:
                   QUESTIONNAIRE: 47    SCORE: 0.820
                   QUESTIONNAIRE: 18    SCORE: 1.000
            BOX:ANALRM  RULE:SD  SCORE: 0.923   Q:
                   QUESTIONNAIRE: 4     SCORE: 0.755
                   QUESTIONNAIRE: 51    SCORE: 0.766
                   QUESTIONNAIRE: 43    SCORE: 1.000
      BOX:ASSESS  RULE:AV  SCORE: 0.549   Q:
            BOX:MULTA   RULE:AV  SCORE: 0.670   Q:
                   QUESTIONNAIRE: 10    SCORE: 0.670
            BOX:INDA1   RULE:SA  SCORE: 0.640   Q:
                  BOX:DDA     RULE:    SCORE: 1.000   Q:
                  BOX:GP      RULE:AV  SCORE: 1.000   Q:
                  BOX:BARR    RULE:SA  SCORE: 0.370   Q:
            BOX:CASSAS  RULE:AV  SCORE: 0.338   Q:
                   QUESTIONNAIRE: 12    SCORE: 0.338
```

Figure 4-5. (Continued)

```
 SELECT 41-51 -- 42
 ENTER BOX NAME -- INDM1

 HIERARCHY DATA FOR BOX INDM1

BOX:INDM1   RULE:SA  SCORE: 0.575   Q:
      BOX:DDS      RULE:     SCORE: 0.833   Q:
      BOX:GP       RULE:AV  SCORE: 1.000   Q:
             QUESTIONNAIRE: 43      SCORE: 1.000
      BOX:BARR     RULE:SA  SCORE: 0.370   Q:
             QUESTIONNAIRE: 3       SCORE: 0.606
             QUESTIONNAIRE: 21      SCORE: 0.911
             QUESTIONNAIRE: 28      SCORE: 0.516
             QUESTIONNAIRE: 38      SCORE: 1.000
             QUESTIONNAIRE: 68      SCORE: 1.000
             QUESTIONNAIRE: 69      SCORE: 0.548
             QUESTIONNAIRE: 90      SCORE: 0.667
 SELECT 41-51 -- 42
 ENTER BOX NAME -- RESP

 HIERARCHY DATA FOR BOX RESP

BOX:RESP    RULE:SA  SCORE: 0.730   Q:
      BOX:DELRSP  RULE:SD  SCORE: 0.790   Q:
            BOX:BARR     RULE:SA  SCORE: 0.370   Q:
                   QUESTIONNAIRE: 3       SCORE: 0.606
                   QUESTIONNAIRE: 21      SCORE: 0.911
                   QUESTIONNAIRE: 28      SCORE: 0.516
                   QUESTIONNAIRE: 38      SCORE: 1.000
                   QUESTIONNAIRE: 68      SCORE: 1.000
                   QUESTIONNAIRE: 69      SCORE: 0.548
                   QUESTIONNAIRE: 90      SCORE: 0.667
            BOX:GP       RULE:AV  SCORE: 1.000   Q:
                   QUESTIONNAIRE: 43      SCORE: 1.000
      BOX:EFFRSP  RULE:SA  SCORE: 0.706   Q:
            BOX:ONSITE  RULE:SA  SCORE: 0.559   Q:
                  BOX:REQOFF  RULE:HA  SCORE: 0.338   Q:
                  BOX:CONADV  RULE:SA  SCORE: 1.000   Q:
            BOX:OFFSIT  RULE:SA  SCORE: 1.000   Q:
                  BOX:RSPREQ  RULE:HA  SCORE: 1.000   Q:
                  BOX:ENGADV  RULE:SA  SCORE: 1.000   Q:
      BOX:DRRSP   RULE:     SCORE: 1.000   Q:
 SELECT 41-51 -- 42
 ENTER BOX NAME -- EFFRSP

 HIERARCHY DATA FOR BOX EFFRSP

BOX:EFFRSP  RULE:SA  SCORE: 0.706   Q:
      BOX:ONSITE  RULE:SA  SCORE: 0.559   Q:
            BOX:REQOFF  RULE:HA  SCORE: 0.338   Q:
                   QUESTIONNAIRE: 12      SCORE: 0.338
                   QUESTIONNAIRE: 16      SCORE: 1.000
            BOX:CONADV  RULE:SA  SCORE: 1.000   Q:
                   QUESTIONNAIRE: 16      SCORE: 1.000
                   QUESTIONNAIRE: 43      SCORE: 1.000
      BOX:OFFSIT  RULE:SA  SCORE: 1.000   Q:
            BOX:RSPREQ  RULE:HA  SCORE: 1.000   Q:
                   QUESTIONNAIRE: 16      SCORE: 1.000
            BOX:ENGADV  RULE:SA  SCORE: 1.000   Q:
                   QUESTIONNAIRE: 16      SCORE: 1.000
```

Figure 4-6. HIERARCHY GRAPHICAL DISPLAY



4-7 (not produced by the computer) illustrates the level by level evaluation taking place for the entire capability.

The following discussion illustrates a hypothetical analysis of how the computer output might aid an evaluator or designer in recommending upgrade procedures for a facility. Let us assume that it is desired to upgrade the facility capability score from a .4 to a .5 by improving the "deny access" portion of the hierarchy. The upper portions of the hierarchy evaluation involve AND type interactions. From descriptions in Section 2, (and also from formal computations with the algorithm formula in Appendix A1), the greatest improvement to an overall score per unit improvement of scores directly beneath it comes from improving the worst score when an AND type operation is involved. In looking at Figure 4-7, we first look at improving the DETACC score (the smaller of the two scores immediately below DENACC). We then look at improving the SENSE and ASSESS boxes to at least the same level as the REPALR box.

Figure 4-5 shows that the ASSESS box is evaluated by averaging the scores of the direct and indirect assessment techniques with the alarm station score. The alarm station score (#12) appears quite low. In addition, the techniques only average together reflecting a not especially well coordinated group of elements in performing the assessment task.

The questionnaire for the alarm stations (#12), indicates that the responses to several of the questions are at their worst levels. If these were changed to their best levels, the component would score .802 instead of .338. If, in addition, the ASSESS elements were coordinated so that a soft OR were appropriate, the total ASSESS score would become .838.

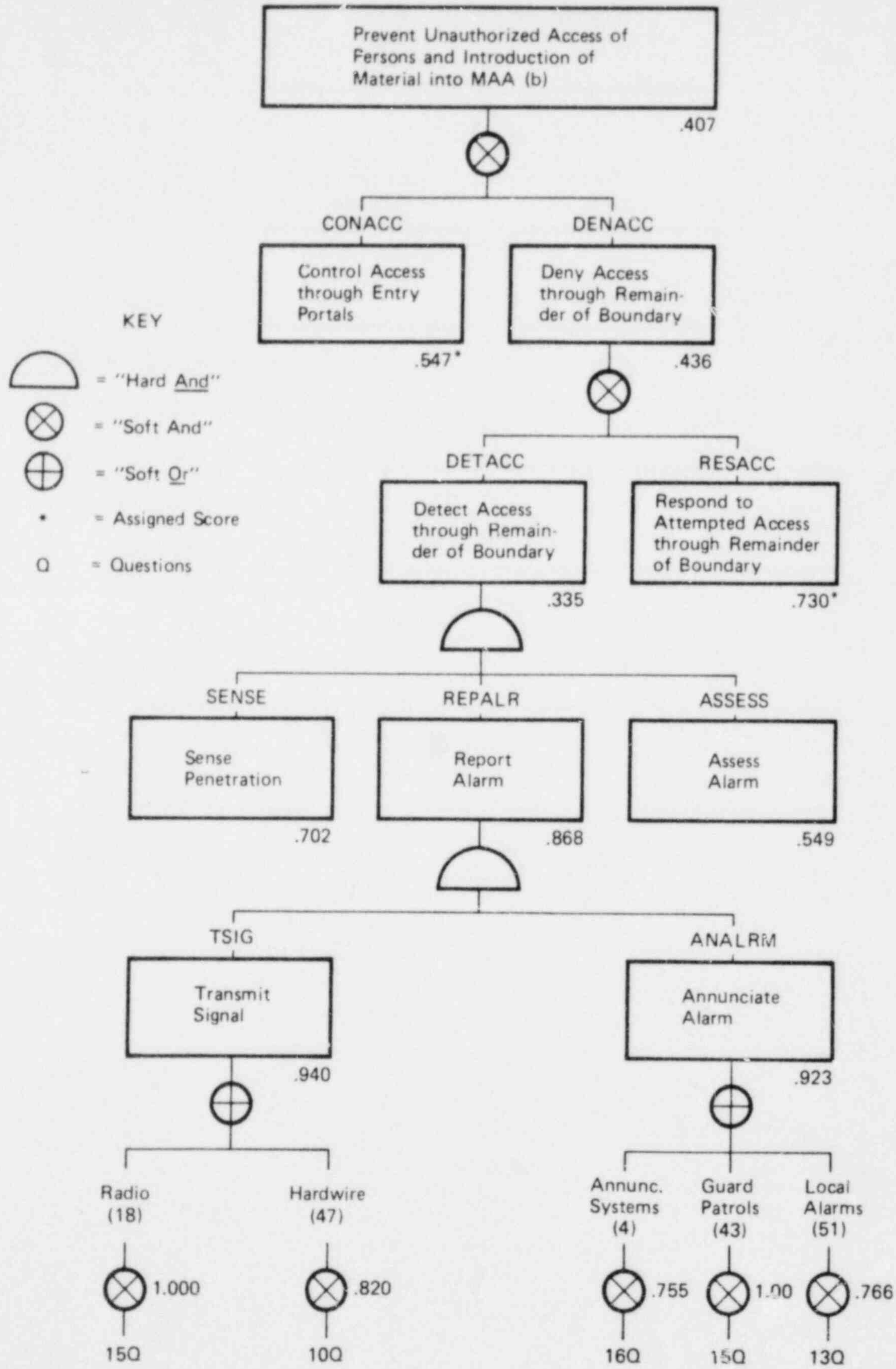


Figure 4-7. AGGREGATION EXAMPLE

In examining the SENSE box, we note two techniques are used. Since the SENSE box is evaluated using an OR rather than an AND, the most improvement is gained by improving the best technique beneath it rather than the worst. This refers to the box labeled MULTS. In Figure 4-5, MULTS is evaluated by averaging a group of components. This reflects the situation that the components do not especially provide defense in depth; e.g., when one component is active another is not. If these components coordinated together more closely, (e.g., all were always active providing some defense in the depth), so that a soft OR were appropriate, the score for MULTS would become .808. Finally, if the direct and indirect assessment techniques were made to provide completely independent, redundant systems so that a hard OR were appropriate, the total score for the SENSE box would be .918.

Working the improved scores up through the hierarchy yields improved scores for the following elements:

SENSE:	.918
ASSESS:	.838
DETACC:	.668
DENACC:	.629
Capability:	.506

The above discussion and example serves to illustrate how the algorithm is implemented using the computer program and how the resulting output can be analyzed to provide insight into and to explore different safeguard strategies.

5.0 SUMMARY AND RECOMMENDATIONS FOR FUTURE WORK

5.1 STRENGTHS OF THE METHODOLOGY

The evaluation methodology presented in this report was designed to have certain desirable properties as outlined in Section 2.1. These strengths of the methodology are now summarized.

- Defensibility of Computational Formulas: The computational rules that are used are not arbitrary. They can be derived formally from a set of assumptions that allow the problem to be decomposed into simpler parts and then logically connected together. Whether the assumptions that justify the use of the computational formulas are valid for a particular component questionnaire or capability hierarchy depends a great deal upon the nature and interpretation of all the elements and factors involved. Because the formulas can be related to specific assumptions, however, the motivation for choosing a particular aggregation rule in a given situation can be explained. The rules have a basis in both probability and utility theory. Both theories provide orientations that appear useful for the safeguards evaluation problem.

- Flexibility of Aggregation Rules: The algorithms are capable of modeling several types of interactions using relatively simple functional forms. The rules were developed to address all the elements of a capability hierarchy including component questionnaires, component combinations, delay-response elements and higher level hierarchy elements.

- Practicality: The methodology can be implemented in a practical fashion using a framework consisting of multiple choice questionnaires and a hierarchy structure. The effort required to specify the parameters of the algorithms is reasonable. Both interaction rules and weighting factors can be assigned in a practical and systematic manner with a provision for some consistency checking.

- Traceability: The computer program allows a user to trace an overall capability score all the way down to a score on an individual component or question. Thus, the reasons why a facility received a particular score can be specified. The computer program has interactive capabilities that especially facilitate such tracing displays.

- Aid to Evaluators: In addition to traceability, the computer program can aid evaluators via sensitivity analysis. By varying the algorithm parameters and/or element scores, evaluators can examine the range of assumptions under which a facility receives an overall score within a certain range, or for which it is less "preferred" than another facility. This type of analysis can provide useful insight into the critical elements affecting a facility's overall evaluation.

- Aid to Designers: Sensitivity analysis can also aid designers. By testing out combinations of potential components, or by adding or upgrading components, a designer can gain insight into what the strengths and weaknesses of a design might be and into what changes can most improve the design score.

In summary, the methodology has the potential for providing, in a practical manner, useful evaluations of safeguards facilities to both evaluators and designers.

5.2 LIMITATIONS OF THE METHODOLOGY

The methodology has limitations stemming from the complexity of safeguards evaluation. Limitations of the methodology and approach are now summarized.

- Multiple-Choice Questionnaire, Capability Hierarchy Framework: An approach selected to characterize a facility should be practical to implement and yet still reflect in a reasonable way essential features of the safeguards system. The questionnaire hierarchy framework is an approach that addresses the evaluation problem, not from the standpoint of providing a detailed model or simulation of a safeguards operation, but rather of providing a large "checklist" of aspects that should be examined in inspecting a design or facility. Information that can be obtained via questionnaires is very diverse in both subject matter and precision. This makes it difficult to handle such questionnaires in a systematic, quantitative fashion. This should be remembered when interpreting an evaluation score. While giving useful insight into a facility, such an evaluation should not be the sole input to a complete safeguards capability evaluation. It should not be expected that this particular approach could address all the complex features of a facility and potential adversary actions.

- Methodology Assumptions: In order to quantitatively evaluate a hierarchy capability, several simplifying assumptions were made in developing the aggregation algorithms. It must be recognized that these assumptions are approximate at best. Careful design of questionnaires and hierarchy structure can help make these assumptions more reasonable approximations. The methodology, like the questionnaire-hierarchy framework, has had to consider the compromise between practical implementation and the ability to address complex features of the problem.

- Methodology Implementation: The current capability hierarchies and questionnaires and the methodology itself have had very little testing in terms of sample or hypothetical facility information. The preliminary testing that has been done has served to point out areas for improvement, further implications of setting certain algorithm parameters and potential redundancies or double-counting in both questionnaires and hierarchies. This issue will be discussed further in the paragraphs to follow.

5.3 RECOMMENDATIONS FOR FUTURE WORK

While it is difficult to address all of the limitations discussed above, there are several areas where additional work could improve the framework and methodology presented in this report. These recommendations are discussed below.

- Design Methodology Testing: Further testing of the methodology is strongly recommended. Availability of the computer program will greatly facilitate this testing. The testing should include:

    1) Evaluating component questionnaires and deciding what question weights and interaction rule can most reasonably reflect the effectiveness of individual components.

    2) Evaluating component combinations and devising, if necessary, improved system questionnaires to deal with this issue.

    3) Evaluating higher level system tasks, whether the current hierarchy structures are appropriate and what interaction coefficients should be assigned to higher levels.

Formal assessments and consistency checking of algorithm parameters is recommended as part of this testing.

This evaluation and testing should be carried out under as realistic conditions as possible. Several existing facilities may provide useful data to help evaluate whether computed scores correspond to an intuitive evaluation of the quality of safeguards. After this testing procedure has been completed, an effort should be made to further simplify the use of the methodology for designers and evaluators.

- Extending the Methodology to Evaluate Overall Safeguards Capabilities: As was mentioned in Section 2.5, the current hierarchy structure does not reflect the concept of several safeguard capabilities working together to provide more complete safeguards than an individual capability considered in isolation could provide. There are possibilities for using the questionnaire information assessed for individual capabilities to evaluate overall safeguards. These may involve restructuring hierarchies especially for this purpose and also devising questionnaires that consider how capabilities coordinate with each other. Since some safeguard systems may be designed with an overall concept in mind, it would be useful to be able to evaluate overall safeguards in addition to individual capabilities.

6.0

GLOSSARY

Disaggregation Structure: The capabilities hierarchy that specifies required safeguard capabilities and defines the functions that a system must perform in order to meet these capabilities (see Figure 6-1).

Performance Capabilities: The five major objectives specified by the NRC that form the highest level of the disaggregation structure.

System Functions: The second level of the disaggregation structure. Those functions that contribute directly to a system capability.

System Subfunctions: All disaggregation levels below the system functions (excluding the lowest levels), which identify specific tasks to be performed by the system.

Low-level System Tasks: Those specific tasks or jobs which follow from the system subfunctions and which correspond to the Performance Characteristics in the Component Selection Matrix.

Performance Characteristics: Constrained low-level system tasks.

Component Selection Matrices: Matrices that indicate correlations between specific tasks via performance characteristics and feasible approaches (equipment and procedures).

SSNM: Strategic Special Nuclear Material.

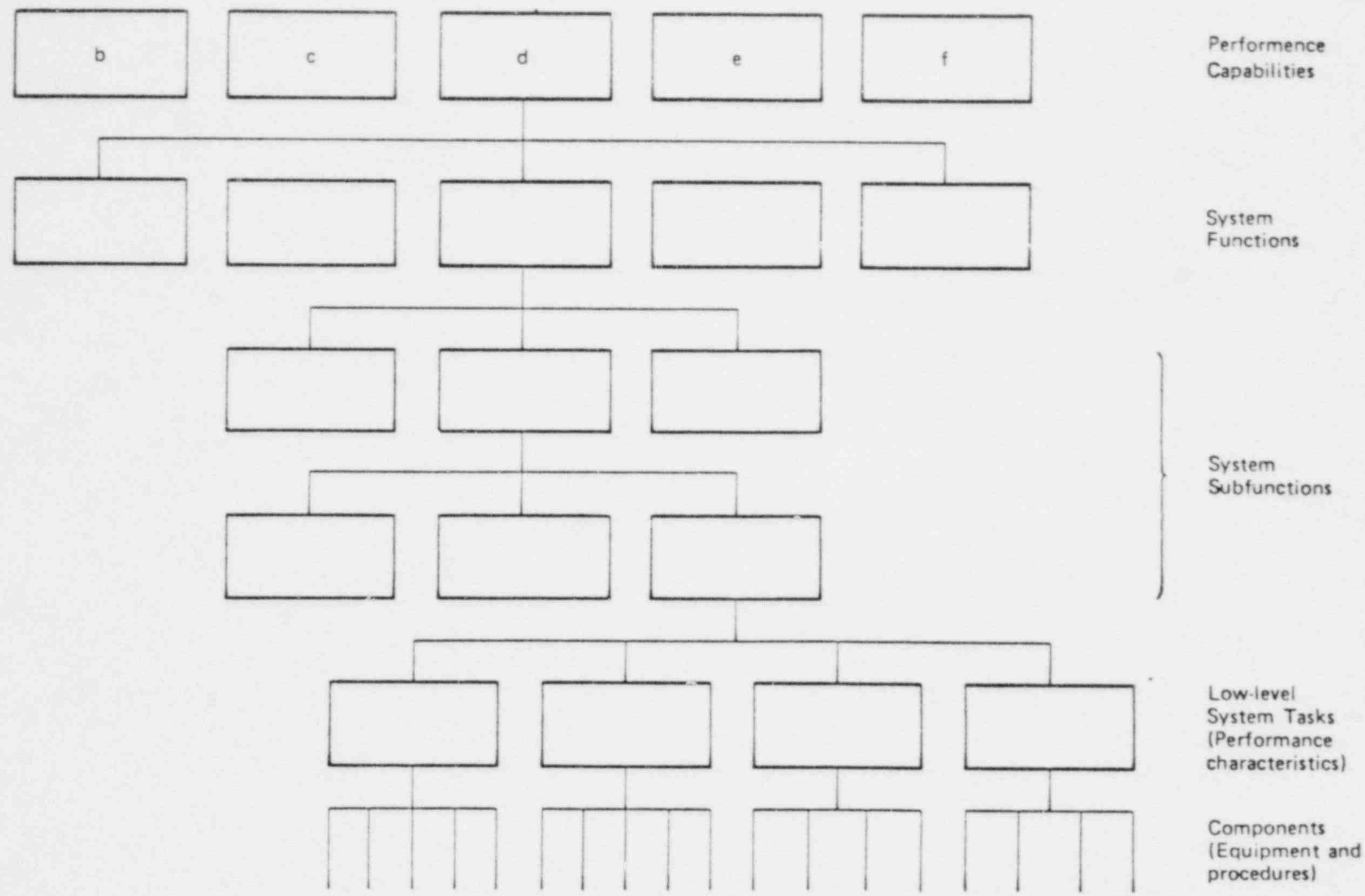


Figure 6-1. DISAGGREGATION STRUCTURE

MAA: Materials Access Area. Any area containing SSNM.

VA: Vital Area. Any area containing equipment that could be sabotaged.

PA: Protected Area. Area surrounding a VA and/or MAA, up to the perimeter of the facility.

ETQ: Effectiveness Test Questionnaire. A set of multiple choice questions that provides information used by the algorithm to evaluate an overall component effectiveness score.

IRS: Information Request Sheet. A more expansive description of a component or safeguard system that provides additional information to evaluators. The algorithm uses only the multiple choice responses and direct scoring information provided by the evaluator in its computations.

System Effectiveness Test Questionnaire: A set of multiple choice questions that provides information used by the algorithm to decide what type of computation is appropriate to combine certain component scores. The questionnaire may also provide the information necessary to combine delay-response type scores, treat multiple access points and factor in facility information not covered in any particlar component questionnaire.

Delay-Response Type Elements: Elements that require a synthesis of delay and response/assessment/monitoring capabilities by comparing delay and response/assessment/monitoring times.

REFERENCES

1. Woodward-Clyde Consultants. A Methodology for Evaluating Safeguards Capability for Licensed Nuclear Facilities. Prepared for Sandia Laboratories, December 1978

2. Keeney, R.L. and H. Raiffa, Decisions with Multiple Objectives. New York: Wiley, 1976

Appendix A1
FUNCTIONAL FORM FOR SAFEGUARDS EVALUATION ALGORITHM

In Section 2.2, the utility function structure used in safeguards evaluation computations was briefly presented. In this appendix, the underlying assumptions made in that structure are described. In addition, further properties of the evaluation function are discussed.

A1.1 Utility Theory

The axioms of decision analysis [1] define a formal logic for evaluating alternatives where the consequences of those alternatives may be uncertain. Specifically, the assumptions utilized in this study imply the existence of a utility function to model the preferences of the evaluators.

Before stating the axioms of utility theory, we define our notation. A <u>simple lottery</u>, written $L(x_1,p,x)$, is a probabilistic event characterized by two possible consequences, which will be designated by x_1 and x_2, and by their respective probabilities of occurrence, designated by p and 1-p. The symbols $>$, $\sim$, and $<$ will be read "is preferred to," "is indifferent to," and "is less preferred than," respectively.[1] Thus, $X_1 \sim L(x_2,p,x_3)$ says that x_1 is indifferent to the lottery which yields either x_2 with probability p or x_3 with probability 1-p.

[1]These designations are only for section A1.1.

The axioms stated here which imply the existence of a utility function are only slightly modified from the formulation of Pratt, Raiffa, and Schlaifer [1].

Axiom u1: Existence of Relative Preferences. For every pair of consequences x_1 and x_2, preferences exist such that either $x_1 \sim x_2$, $x_1 > x_2$, or $x_2 < x_1$.

Axiom u2: Transitivity. For any lotteries L_1, L_2, and L_3, the following hold:

i) $L_1 \sim L_2$ and $L_2 \sim L_3$ implies that $L_1 \sim L_3$

ii) $L_1 > L_2$ and $L_2 \sim L_3$ implies that $L_1 > L_3$, etc.

Since a consequence can be interpreted as a degenerate lottery (i.e., $p = 1$), axioms u1 and u2 together imply the existence of a ranking of the relative desirabilities of the various possible consequences. They do not say that an individual can articulate this, nor do they require that this ranking be stationary over time. Let us designate as x° a consequence which is not preferred to any of the other consequences for a problem and as x^* a consequence which is at least as preferred as each of the other consequences. Therefore, one possiblity is that x° and x^* designate the least and most preferred consequences, although they may represent hypothetical consequences such that $x^* > x$ and $x > x^\circ$ for all possible x.

Axiom u3. Comparison of Simple Lotteries. Given the preference order $x_1 > x_2$, then

i) $L_1(x_1,p_1,x_2) > L_2(x_1,p_2,x_2)$ if $p_1 > p_2$,

ii) $L_1(x_1,p_1,x_2) \sim L_2(x_1,p_2,x_2)$ if $p_1 = p_2$.

Axiom u4. Quantification of Preference. For each possible consequence x, the evaluator can specify a number $\pi(x)$, where

$0 \leq \pi(x) \leq 1$, such that $x \sim L(x^*, \pi(x), x^\circ)$.

Axioms u3 and u4 taken together establish a measure of the relative desirabilities of the various consequences to the evaluator. The $\pi(x)$ value - or indifference probability, as it is called - is that measure.

Clearly, since the standards x° and x^* for measuring $\pi(x)$ are somewhat arbitrary, different π functions may be assess or a specific individual in a particular situation. To be consistent with these axioms, however, all possible functions must be positive linear transformations of each other. Any positive linear transformation of π of the form

$$u(x) = a + b\pi(x),\ b > 0$$

is referred to as a utility function. The quantity $u(x)$ is said to be the utility of consequence x. If one accepts the above axioms, one should always prefer alternatives that maximize expected utility. There are no alternative procedures for making decisions consistent with these axioms.

Since maximizing expected utility is equivalent to maximizing the expected value of π, the arbitrary choice of x^* and x° has no influence on the actual decision. Utility provides a relative scale analogous to the temperature scales, and two scales which are positive linear transformations of each other are identical for decision-making purposes.

A1.2 INDEPENDENCE ASSUMPTIONS

The theory underlying multiattribute utility functions is presented in detail in [2]. Here we briefly present the theoretical results that underlie the utility functions used in the evaluation methodology presented this report. Suppose $\{X_1, X_2, \ldots, X_N\}$ are the attributes of an evaluation problem. For notational convenience let X_I be any specified subset of $\{X_1, X_2, \ldots, X_N\}$ and $\bar{X}_I$ be the complement of X_I. Then X_I is <u>utility independent</u> of $\bar{X}_I$ if preferences for risky choices (lotteries) over X_I with the value of $\bar{X}$ held fixed do not depend on the fixed value of $\bar{X}_I$. If X_I is utility independent of $\bar{X}_I$ for all X_I then $\{X_1, X_2, \ldots, X_N\}$ are <u>mutually utility independent</u>.

<u>Theorem</u> [2, Theorem 6.1]. If $\{X_1, X_2 \ldots, X_N\}$ are mutually utility independent then either

$$u(x_1, x_2, \ldots, x_N) = \sum_{n=1}^{N} k_n u_n(x_n) \qquad \text{(A1.1a)}$$

or

$$1 + Ku(x_1, x_2, \ldots, x_N) = \prod_{n=1}^{N} [1 + Kk_n u_n(x_n)] \qquad \text{(A1.1b)}$$

where x_n represents a specific value of X_n, u and the u_n's are utility functions scaled from zero to one, the k_n's are scaling constants with $0<k_n<1$, and $K>-1$ is a nonzero scaling constant which is the solution to

$$K + 1 = \prod_{n=1}^{N} (Kk_n + 1). \qquad \text{(A1.1c)}$$

Note that in the derivation of the utility function for the safeguards evaluation, some or all of the attributes may be vectors.

In Section 2.2, the expanded expression for the utility function is given with the following relationships:

$S = u(x_1, x_2, \ldots, x_n)$
$C = k_n$ for all N attributes
$\mu_n = S_n$ for all N attributes

Thus, in simplifying the expression for the aggregation algorithm,

$$S = \sum_{i=1}^{N} S_i/N \qquad \text{or} \tag{A1.2}$$

$$S = \frac{\prod_{i=1}^{N}(1 + VS_i) - 1}{(1 + V)^N - 1} \tag{A1.3}$$

A1.3 SYSTEM QUESTIONNAIRES FOR DETERMINING CONDITIONAL UTILITY FUNCTIONS

The parameter V in (A1.3) can be considered to be conditional upon other factors that are assumed fixed for a particular evaluation. If such factors change, it is possible that V might change. A system questionnaire provides the mechanism for assigning V conditional upon how well a group of elements coordinate with one another.

Specifically, in the Phase 1 report in Appendix A3 [3], it was shown how the interaction coefficient was related to the value of Y that would make a decision maker indifferent between the following:

A. Element 1, Element 2
$(S_1=1,\ S_2=0)$

B. Element 1, Element 2
$(S_1=Y,\ S_2=.5)$

It was shown there, that V=(1/Y)-2

A value of Y near 1 would indicate that S_1 and S_2 were redundant or substitutes for each other. A value of Y near 0 would show that a fatal flaw in either one would cause the other to be almost worthless. A system questionnaire provides a mechanism for computing more formally what value of Y a decision maker would feel to be appropriate for the above tradeoff. Specifically, Y is set equal to the score on the system questionnaire. To simplify the algorithm, the following correspondences between Y and V are used:

Y	V	
.8-1.	V=-1	(OR)
.6-.8	V=-1/2	(SOFT OR)
.4-.6	V=0	(AVERAGE)
.2-.4	V=1	(SOFT AND)
0-.2	V=∞	(AND)

In summary, V is conditional upon the score of the system questionnaire in the manner indcated above. In these instances, (A1.3) with the appropriate value of V is a conditional evaluation or utility function based on howe well the elements coordinate with each other as determined by the system questionnaire.

A1.4 SOME PROPERTIES OF THE FUNCTIONAL FORM

If we take the partial derivative of S with respect to S_i in (A1.2) and (A1.3) we obtain the following:

$$\frac{\partial S}{\partial S_i} = \frac{1}{N} \quad \text{for } V=0 \qquad \text{(A1.4)}$$

$$\frac{\partial S}{\partial S_i} = \left(\prod_{\substack{j=1 \\ j \neq i}}^{N} (1+V\, S_j) \right) * V/K \qquad \text{(A1.5)}$$

In (A1.5) the quality V/K is always positive. In examining (A1.5), when $-1 \leq V < 0$, the product term is largest when the S_i excluded is the largest. This implies that for an OR type aggregation, the greatest gain per unit increase in an S_i is achieved when the largest S_i element is selected. In other words, to upgrade a combination of components which OR together, it would pay to improve the best element further, if possible. When $0 < V < \infty$, the product term is largest when the S_i excluded is the smallest. Thus for an AND type aggregation, it would "pay" to improve the worst element further, if possible.[1]

Of course, there are other factors which would influence those elements to be selected for potential upgrading; e.g., cost, feasibility of improvement, etc., But the algorithm's functional form does provide some guidance on this issue.

---

[1] In this sense, the maximum and minimum element scores have the greatest influence respectively on OR and AND type aggregations. But, unlike using a MAX or MIN operator, their influence need not completely override the contribution of the other elements.

REFERENCES

1. Pratt, J.W., Raiffa, H., and Schlaifer., R. 1964. The Foundations of Decision under Uncertainty: an Elementary Exposition. Journal of the American Statistical Association 59:353-357.

2. Keeney, R.L. and H. Raiffa, Decisions with Multiple Objectives; Preferences and Value Tradeoffs. New York: Wiley, 1976

3. Woodward-Clyde Consultants, A Methodology for Evaluating Safeguards Capability for Licensed Nuclear Facilities, prepared for Sandia Laboratories, December 1978.

Appendix A2

ASSESSING ALGORITHM PARAMETERS

In Section 3.5, a technique was presented with which the parameters of the algorithm could be assessed in a consistent manner. The derivation of the formulas shown in that section are now presented.

To review, to calibrate the weight W for a group of equally weighted elements in a manner consistent with the selected aggregation coefficient, one should choose n and P for which the followign two situations are equally preferred:

<u>A</u>

(N-n elements at their best,
n elements at their worst)

<u>B</u>

P — All elements at their best (i.e., S=1)

1-P — S=0

In practice, P is sometimes set to .5, and n is then determined. The formulas for u in terms of N,n and P are now derived.

The basic equations used are the formulas (A1.2) and (A1.3). As appendix A1 explained, the expected utility or score for situations A and B above should be set equal to each other since they are equally preferred. In terms of formula (A1.2), the expected utility for situation A is equal to $(N - n + n(1-W))/N$. The expected utility for situation B is P. Equating these two scores, yields:

$$W = (1-P)N/n$$

There is a transformation of formula (A1.3) that is useful in deriving expressions for W. It is a noormalization scheme that has the scores going from -1 to 0 rather than 0 to 1. To make this transformation, the following quantities are defined:

$V' = V/(1+V) \rightarrow V = V'/(1-V')$

$S_i' = S_i - 1 \rightarrow S_i = 1 + S_i'$

$S' = S-1 \rightarrow S = 1 + S'$

In terms of the new quantities, (using formula (A1.3) and reducing),

$$S' = \frac{\prod_{i=1}^{N} (1 + V'S_i') - 1}{1-(1-V')^N} \qquad \text{(A2.1)}$$

Both formulas (A1.3) and (A2.1) will now be used to derive expressions for W.

The expected utility for B in the new variable system is P(0)+(1-P)(-1) or P-1. The expected utility for A using formula (A2.1) is simply:

$$\frac{(1-V'W)^n - 1}{1-(1-V')^N} \qquad \text{(A2.2)} \qquad -\infty < V' \leq 1,\ V' \neq 0$$

(S_i' for a perfect element is 0 in the new system). Equating (A2.2) with P-1 yields the following for W:

$$W = (1-(P+(1-P)(1-V')^N)^{1/n}/V' \qquad \text{(A2.3)}$$

The formula in the original variable system is shown in Section 3.5.

For $V \to \infty$, $V' \to 1$, and formula (A2.3) becomes $W = 1-P^{1/n}$.

For $V=1$, $V'=1/2$ (soft AND). For N large, the term containing $(1-V')^N$ is multiplied by $(1/2)^N$ and can be ignored in (A2.3) for $N>5$.

For $V=-1$ (hard OR), the use of formula (A1.3) yields an expected utility for A of $1-W^N$ for the case where all elements are at their worst. (Even one at its best will yield a utility of 1). Solving for W in this case yields $W=(1-P)^{1/N}$.

APPENDIX A3

COMPUTER PROGRAM DOCUMENTATION

A3.1 GENERAL OVERVIEW

Two versions of the safeguards evaluation computer program were written in FORTRAN IV. This appendix contains listings of both versions. An interactive program design was chosen because (1) it facilitates easy use of the program by providing easy to understand dialogue and (2) it greatly facilitates sensitivity analysis as well as fine tuning of algorithm parameters. The first version has more interactive options than the second, but may require more modifications to implement on non-IBM systems. This version is not strictly 1966 ANSI standard[1]. However, it has been recognized that the 1966 standard is quite limited in certain respects. Specifically, no random access input-output is allowed by the 1966 standards. A new standard (FORTRAN 77)[2] has been published, but as yet is not implemented on most systems.

The second version is less interactive and replaces all the random access input-output statements with sequential input-output statements. This second version is intended to comply with the 1966 standard and should be easy to implement most systems. It has the following limitations: Options 21 and 48 may only be done once in a run and options 22 through 29 are not available.

A3.2 LANGUAGE AND OPERATING SYSTEM CHARACTERISTICS

Both programs do not require large amounts of core, but do utilize four disk files for input and output as explained in Section 4 of the main report. Currently, there is room for 100 questionnaires with up to 40 questions each, and 40 "boxes" per capability. These dimensions can be changed if necessary. With the current dimensions, it is sometimes necessary to evaluate an entire capability in two pieces. This was done for the example discussed in Section 4 and presented in more detail in Appendix A4.

Some features that may require modification when implementing the program on different systems are now discussed. DOUBLE PRECISION statements are necessary to accommodate label variables of more than 4 characters on IBM machines. DOUBLE PRECISION statements should be removed for CDC machines. All random access input-output related statements are usually different for different computer systems. The current fully interactive version is IBM compatible only. The hierarchy graphical display routine utilizes certain carriage control characters that may be different or not present on some systems. The graphical display option 50 should not be used on systems that do not support a control character that allows the printer to remain on the same line with no carriage return and no line feed. Finally, CDC systems require a PROGRAM CARD to appear as the first card in the main program.

The mnemonic QUEAS (Quantitative Evaluation of safeguards) is given to the program files which contain the routines implementing the algorithm. The program routines and data files are described in Table A3-1. The program listings follows the table. The routines with a "B" appended to the file name replace their counterparts to compose version B of the program.

REFERENCES

1. American Standard FORTRAN, New York: American Standards Association, Inc., 1966.

2. Harry Katzan Jr., FORTRAN 77. New York: Van Nostrand Reinhold Company, 1978.

Table A3-1. PROGRAM ROUTINES AND DATA FILES

QUEASI (FORTRAN): Main Program File

QUEAS1 (FORTRAN): First Subroutine File

SUBROUTINE GETHN :	Interactively requests a hierarchy box name and returns the corresponding number
SUBROUTINE ATGET :	Accepts a hierarchy box name and returns the corresponding number
SUBROUTINE SCREH :	Accepts a box number and returns the box score
SUBROUTINE BOX :	Computes score for a box with questionnaires as input
SUBROUTINE MULTEV:	First level of recursive scoring routine. Tries to compute box score
SUBROUTINE GETQN :	Interactively requests a questionnaire name and returns the corresponding number
FUNCTION YESNO :	Accepts an interactive answer to a yes/no question and returns 0 for no, 1 for yes
FUNCTION GETNUM :	Accepts a number from within a specific range interactively
SUBROUTINE TXTURE:	Combines a group of numbers using a scoring rule
SUBROUTINE MULTE1:	Second level of recursive box scoring routine
SUBROUTINE MULTE2:	Third level of recursive box scoring routine
SUBROUTINE MULTE3:	Fourth level of recursive box scoring routine
SUBROUTINE MULTE4:	Fifth level of recursive box scoring routine
SUBROUTINE MULTE5:	Final level of recursive box scoring routine
SUBROUTINE GETR :	Reads a questionnaire's responses

QUEAS2 (FORTRAN): Second Subroutine File

SUBROUTINE PRINTH:	Prints short version of hierarchy structure starting with specified box
SUBROUTINE SETH :	Resets higher score values when a box score is set

Table A3-1. (Continued)

FUNCTION NQ : Returns the questionnaire number of a supplied name

QUEAS3 (FORTRAN): Third Subroutine File

FUNCTION SCOREQ : Returns the computed score of a questionnaire

SUBROUTINE GETQ : Retrieves questionnaire structure

SUBROUTINE PRINTQ: Prints detailed questionnaire data

QUEAS4 (FORTRAN): Fourth Subroutine File

SUBROUTINE HPR : Prints hierarchy diagram

SUBROUTINE PNAME : Prints names of hierarchy boxes

SUBROUTINE PQNAME: Prints names of questionnaires

FILE1 (DATA): Questionnaire structure data

FILE2 (DATA): Questionnaire scores

FILE3 (DATA): Hierarchy structure data

FILE4 (DATA): Initial hierarchy scores

```
C QUEASI -- A PROGRAM FOR EVALUATION OF SAFEGUARDS QUESTIONNAIRES
C AND HIERARCHIES
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      COMMON SCOREH(40),RULEH(40),IDEX(40,10),QDEX(40),QNAME(100)
      DOUBLE PRECISION BNAME(40),BLANK,NOMO,HNAME(40),ANAME
      INTEGER LOCHS(5),ISET(40),FLAG(10),LOCH(5),IRESP(40),IBEST(40)
      REAL SCORE(40),WEIGHT(40),RULE(10),TEXTS(7)
      DATA TEXTS/2HHA,2HSA,2HAV,2HSO,2HOR,1HQ,2HDR/
      DATA IAA/1HA/,NOMO/6HNOMORE/BLANK/6H      /
      DEFINE FILE 1 (300,80,E,I9)
      DEFINE FILE 3 (200,80,E,I9)
      DEFINE FILE 2 (150,80,E,I9)
      DEFINE FILE 4 (50,80,E,I9)
      AAA=FLOAT(IAA)
      DO 399 I=1,40
  399 SCOREH(I)=-1.
      J=2
C READ QUESTIONNAIRE LOCATIONS AND NAMES
      READ(1'1,9464) NUMQ
      READ(1'J,9465)(LOCQ(I1),I1=1,NUMQ)
      READ(2'J,9465)(LOCR(I2),I2=1,NUMQ)
      DO 30 I=1,NUMQ
      J=LOCQ(I)
   30 READ(1'J,9272) QNAME(I)
      DO 10 I=1,100
   10 QSCORE(I)=-1.
      DO 20 I=1,10
   20 FLAG(I)=0
C SELECT INITIAL OPTION
 1000 WRITE(6,9100)
 9100 FORMAT(54H? SELECT 1- HIERARCHIES, 2- QUESTIONAIRES, 3- STOP -- )
      IOP=GETNUM(1.,3.,2.)
 1001 IF (IOP.EQ.3) STOP
      GOTO (4000,2000) IOP
C REVIEW OPTION SELECTION AND BRANCH TO PROPER OPTION
      IOPT=IOP
 1100 CONTINUE
      IF (IOPT.GE.1.AND.IOPT.LE.3) GOTO 1101
      IF(IOPT.GE.21.AND.IOPT.LE.26) GOTO 1102
      IF (IOPT.GE.41.AND.IOPT.LE.50) GOTO 1102
      GOTO (1000,230,1000,402) IOP
 1101 IOP=IOPT
      GOTO 1001
 1102 IOP=INT(FLOAT(IOPT)/10.)
      IF(IOP*10.EQ.IOPT) GOTO 1001
      IOPT=IOPT-IOP*10
      GOTO (1000,201,1000,404) IOP
      GOTO 1000
C PRINT MENU AND GET QUESTIONNAIRE OPTION
 2000 IF (FLAG(2).EQ.1) GOTO 210
  230 WRITE(6,9200)
      WRITE(6,9201)
      WRITE(6,9202)
      WRITE(6,9203)
      WRITE(6,9204)
```

```
      WRITE(6,9205)
 9200 FORMAT(/,14H  SELECT ONE: ,/)
 9201 FORMAT(51H  21- COMPUTE SCORES        22- PRINT SCORES          )
 9202 FORMAT(51H  23- SET SCORES            24- REVISE WEIGHTS        )
 9203 FORMAT(51H  25- REVISE RULES          26- REVISE RESPONSES      )
 9207 FORMAT(51H  27- REVISE NAMES          28- PRINT NAMES           )
 9204 FORMAT(51H            29- NO MORE REVISIONS                     )
 9205 FORMAT(/,9H? WHICH?  )
      GOTO 220
  210 WRITE(6,9206)
 9206 FORMAT(18H? SELECT 21-29 -- )
  220 IOPT=GETNUM(21.,29.,2.)
      IF (IOPT.LT.21.OR.IOPT.GT.29) GOTO 1100
      IOPT=IOPT-20
  201 FLAG(2)=1
C BRANCH TO PROPER OPTION
      GOTO (2001,2002,2003,2004,2005,2006,2007,2008,1000) IOPT

C -- OPTION 22 TO PRINT DATA FOR A QUESTIONNAIRE --
 2002 GOTO 222
  223 DO 221 I=1,NUMQ
      CALL PRINTQ(I,QNAME)
  221 CONTINUE
      GOTO 2000
  222 CALL GETQN(ID,QNAME,NUMQU)
      IF(ID.EQ.-1) GOTO 223
      CALL PRINTQ(ID,QNAME)
      GOTO 2000

C -- OPTION 21 TO COMPUTE A QUESTIONNAIRE SCORE --
 2001 GOTO 213
  214 DO 211 I=1,NUMQ
      WRITE(6,9301) QNAME(I)
 9301 FORMAT(16H? QUESTIONNAIRE ,1A4,1H:)
  211 CALL SCOREQ(I)
      GOTO 2000
  213 CALL GETQN(ID,QNAME,NUMQU)
      IF(ID.EQ.-1) GOTO 214
      CALL SCOREQ(ID)
      GOTO 2000

C -- OPTION 23 TO SET A QUESTIONNAIRE SCORE --
 2003 CALL GETQN(ID,QNAME,NUMQU)
      WRITE(6,9300)
 9300 FORMAT(30H? ENTER QUESTIONAIRE SCORE -- )
      QSCORE(ID)=GETNUM(0.,1.,1.)
      GOTO 2000

C -- OPTION 24 TO REVISE QUESTION WEIGHTS --
 2004 CALL GETQN(ID,QNAME,NUMQU)
      QSCORE(ID)=-1.
      WRITE(6,9240)
 9240 FORMAT(39H? NUMBER OF QUESTIONS TO BE REVISED -- )
      NUM=GETNUM(1.,FLOAT(NUMQU),0.)
      LOC=LOCQ(ID)
```

```
      READ(1*LOC,9500) R,NQQQ,NGRP
      LOC=LOC+2+2*NGRP
      READ(1*LOC,9243)(WEIGHT(K),K=1,8)
      IF(WEIGHT(1).LE.1) GOTO 241
      WRITE(6,9244)
 9244 FORMAT(47H? MUST REVISE ALL WEIGHTS. ENTER NEW WEIGHT -- )
      WEIGHT(2)=GETNUM(0.,1.,1.)
      WRITE(1*LOC,9243)(WEIGHT(K),K=1,8)
      GOTO 2000
  241 LOC=LOC+1
      IF(NUMQU.LE.8) GOTO 243
      READ(1*LOC,9243) (WEIGHT(K),K=9,NUMQU)
  243 DO 240 I=1,NUM
      WRITE(6,9241)
 9241 FORMAT(20H? QUESTION NUMBER = )
      NUMQ=GETNUM(1.,40.,0.)
      WRITE(6,9242)
 9242 FORMAT(11H? WEIGHT = )
      W=GETNUM(5.,1.,1.)
  240 WEIGHT(NUMQ)=W
 9243 FORMAT(8F5.3)
      LOC=LOCQ(ID)+2+NGRP*2
      WRITE(1*LOC,9243)(WEIGHT(K),K=1,NUMQU)
      CALL SCOREQ(ID)
      GOTO 2000

C -- OPTION 25 TO REVISE SCORING RULES --
 2005 CALL GETQN(ID,QNAME,NUMQU)
      QSCORE(ID)=-1.
      LOC=LOCQ(ID)
      READ(1*LOC,9500) R,NQQQ,NGRP
 9500 FORMAT(1A2,2(8X,1I2))
  251 WRITE(6,9502)
 9502 FORMAT(24H? ENTER GROUP NUMBER  -- )
      N=GETNUM(50.,FLOAT(NGRP)+49.,0.)
      WRITE(6,9503)
 9503 FORMAT(33H? ENTER RULE (HA,SA,AV,SO,OR) -- )
  254 READ(5,9504) R
 9504 FORMAT(1A2)
      DO 252 I=1,5
      IF(R.EQ.TEXTS(I)) GOTO 253
  252 CONTINUE
      WRITE(6,9505)
 9505 FORMAT(25H? BAD RULE, TRY AGAIN -- )
      GOTO 254
  253 L=LOC+2
      DO 255 I=1,NGRP
      READ(1*L,9506) IGRP,M,Z
 9506 FORMAT(2(1I2,8X),1A2)
      IF(IGRP.EQ.N) WRITE(1*L,9506) IGRP,M,R
  255 L=L+2
      CALL SCOREQ(ID)
      GOTO 2000

C -- OPTION 26 TO REVISE QUESTION RESPONSES --
```

```
 2006  CALL GETQN(ID,QNAME,NUMQU)
       QSCORE(ID)=-1.
       LOC=LOCR(ID)+1
       READ(2*LOC,9600) IRESP
 9600  FORMAT(40(1X,1A1))
       L=LOCQ(ID)+1
       READ(1*L,9600) IBEST
       WRITE(6,9240)
       N=GETNUM(0.,40.,0.)
       DO 260 I=1,N
       WRITE(6,9241)
       NUM=GETNUM(0.,40.,0.)
  261  WRITE(6,9603) IBEST(NUM)
 9603  FORMAT(16H? RESPONSE (A TO,1X,1A1,4H) = )
  260  READ(5,9260) IRESP(NUM)
 9260  FORMAT(1A1)
       X=1-(AAA-FLOAT(IRESP(NUM)))/(AAA-FLOAT(IBEST(NUM)))
       IF(X.LT.0..OR.X.GT.1.) GOTO 261
       WRITE(2*LOC,9600) IRESP
       CALL SCOREQ(ID)
       GOTO 2000

C -- OPTION 27 TO REVISE QUESTIONNAIRE NAMES
 2007  CALL GETQN(ID,QNAME,NUMQU)
       WRITE(6,9271)
 9271  FORMAT(15H? ENTER NAME -- )
       READ(5,9272) QNAME(ID)
 9272  FORMAT(1A4)
       GOTO 2000

C -- PRINT AND SELECT HIERARCHY MANIPULATION OPTIONS --
 2008  CALL PQNAME(QNAME,NUMQ)
       GOTO 2000

 4000  IF (FLAG(5).EQ.0) GOTO 4006
       IF(FLAG(4).EQ.1) GOTO 401
  402  WRITE(6,9400)
       WRITE(6,9401)
       WRITE(6,9402)
       WRITE(6,9403)
       WRITE(6,9404)
       WRITE(6,9405)
       WRITE(6,9407)
       WRITE(6,9205)
 9400  FORMAT(/,14H  SELECT ONE: ,/)
 9401  FORMAT(51H  41- COMPUTE SCORES        42- PRINT DATA              )
 9402  FORMAT(51H  43- ASSIGN SCORES         44- REVISE DELAY/RESP       )
 9403  FORMAT(51H  45- REVISE RULES          46- SELECT NEW HIERARCHY )
 9404  FORMAT(51H  47- PRINT BOX NAMES       48- FILE HIERARCHY DATA  )
 9405  FORMAT(51H  49- CHANGE BOX NAME       50- PRINT HIERARCHY      )
 9407  FORMAT(51H                  51- NO MORE REVISIONS              )
       GOTO 403
  401  WRITE(6,9406)
 9406  FORMAT(18H? SELECT 41-51 -- )
  403  IOPT=GETNUM(41.,47.,2.)
```

```
      IF(IOPT.LT.41.OR.IOPT.GT.51) GOTO 1100
      IOPT=IOPT-40
 404  FLAG(4)=1
C BRANCH TO PROPER HIERARCHY OPTION
      IF(FLAG(5).EQ.0) GOTO 4006
      GOTO(4001,4002,4003,4004,4005,4006,4007,4008,4009,4002,1000) IOPT

C -- OPTION 41 TO SCORE A HIERARCHY BOX --
 4001 CALL GETHN(ID,HNAME)
      CALL SCREH(ID)
      CALL SETH(ID,ISET)
      ISET(ID)=0
      GOTO 4000

C -- OPTION 42 TO PRINT HIERARCHY DATA --
 4002 CALL GETHN(ID,HNAME)
      IF(IOPT.EQ.2) CALL PRINTH(ID,HNAME)
      IF (IOPT.EQ.10) CALL HPR(ID,HNAME)
      GOTO 4000

C -- OPTION 43 TO ASSIGN HIERARCHY BOX SCORES --
 4003 CALL GETHN(ID,HNAME)
      WRITE(6,9430)
      CALL SETH(ID,ISET)
 9430 FORMAT(10H? SCORE = )
      SCOREH(ID)=GETNUM(-1.,1.,1.)
      IF(SCOREH(ID).LT.0) ISET(ID)=0
      IF(SCOREH(ID).GE.0) ISET(ID)=1
      GOTO 4000

C -- OPTION 44 TO REVISE DELAY/RESPONSE RULE. NOT CURRENTLY USED
 4004 CALL GETHN(ID,HNAME)
      IF(RULEH(ID).EQ.TEXTS(7)) GOTO 440
      WRITE(6,9440)
 9440 FORMAT(28H BOX DOES NOT USE DELAY/RESP)
      GOTO 4000
 440  CONTINUE
      GOTO 4000

C -- OPTION 45 TO REVISE SCORING RULES --
 4005 CALL GETHN(ID,HNAME)
      WRITE(6,9503)
 450  READ(5,9504) R
      DO 451 I=1,6
      IF (R.EQ.TEXTS(I)) GOTO 452
 451  CONTINUE
      WRITE(6,9505)
      GOTO 450
 452  RULEH(ID)=R
      SCOREH(ID)=-1.
      CALL SETH(ID,ISET)
      GOTO 4000

C -- OPTION 46 TO ENTER A NEW HIERARCHY INTO THE SYSTEM --
 4006 WRITE(6,9460)
```

```
      FLAG(5)=1
      READ(3*1,9464) NUMH
      READ(3*2,9465) (LOCH(I),I=1,NUMH)
 9464 FORMAT(1I2)
 9465 FORMAT(20I4)
      READ(4*1,9464) NUMH
      READ(4*2,9465)(LOCHS(I),I=1,NUMH)
 9460 FORMAT(28H? ENTER HIERARCHY NUMBER -- )
      HNUM=GETNUM(1.,FLOAT(NUMH),1.)
      DO 465 I=1,40
      ISET(I)=0
      DO 466 I1=1,10
  466 IDEX(I,I1)=0
      HNAME(I)=BLANK
  465 SCOREH(I)=-1.
      L=0
      LOC=LOCH(HNUM)
C BEGIN BY READING INFO FOR FIRST BOX
  460 READ(3*LOC,9461) ANAME,NUM,R,Q
 9461 FORMAT(1A6,4X,1I2,8X,1A2,8X,1A4)
      IF(ANAME.EQ.NOMO) GOTO 468
      CALL ATGET(ANAME,ID,HNAME)
      IF(ID.GT.0) GOTO 464
      L=L+1
      ID=L
      HNAME(ID)=ANAME
  464 IF(NUM.EQ.0) GOTO 463
      QDEX(ID)=Q
      RULEH(ID)=R
  461 IDEX(ID,1)=NUM
      IF(NUM.EQ.0) GOTO 463
      IF(NUM.GT.50) GOTO 4601
      DO 462 J=1,NUM
      J1=J+1
      LOC=LOC+1
      READ(3*LOC,9462) ANAME
      CALL ATGET(ANAME,IE,HNAME)
      IF(IE.GT.0) GOTO 462
      L=L+1
      IE=L
      HNAME(L)=ANAME
  462 IDEX(ID,J1)=IE
 9462 FORMAT(1A6)
  463 LOC=LOC+1
      GOTO 460
 4601 NUM=NUM-50
      DO 4602 J=1,NUM
      J1=J+1
      LOC=LOC+1
      READ(3*LOC,9272) ANAM
 4602 IDEX(ID,J1)=NQ(ANAM)
      LOC=LOC+1
      GOTO 460
  468 CONTINUE
      J=LOCHS(HNUM)
```

```
      READ(4*J,9463)((BNAME(I1),SCORE(I1)),I1=1,L)
      J=J+L/5+1
      READ(4*J,9480) QSCORE
 9463 FORMAT(5(1A6,1F6.3))
      DO 469 K=1,L
      CALL ATGET(BNAME(K),ID,HNAME)
      IF (ID.EQ.0) GOTO 469
      SCOREH(ID)=SCORE(K)
  469 CONTINUE
      DO 459 I=1,40
  459 IF(IDEX(I,1).EQ.0) RULEH(I)=BLANK
      GOTO 4000

C -- OPTION 47 TO PRINT CURRENT BOX NAMES --
 4007 CALL PNAME(HNAME)
      GOTO 4000

C -- OPTION 48 TO FILE HIERARCHY DATA --
 4008 LOC=LOCH(HNUM)
      DO 480 I=1,L
      IF(HNAME(I).EQ.BLANK) GOTO 480
      WRITE(3*LOC,9461) HNAME(I),IDEX(I,1),RULEH(I),QDEX(I)
      I4=IDEX(I,1)
      IF(I4.EQ.0) GOTO 480
      LOC=LOC+1
      I1=MOD(IDEX(I,1),50)+1
      DO 481 I2=2,I1
      I3=IDEX(I,I2)
      IF(I4.LE.50) GOTO 483
      WRITE(3*LOC,9272) QNAME(I3)
      GOTO 481
  483 WRITE(3*LOC,9462) HNAME(I3)
  481 LOC=LOC+1
  480 CONTINUE
      WRITE(3*LOC,9462) NOMO
      J=LOCHS(HNUM)
      WRITE(4*J,9463)(HNAME(I3),SCOREH(I3),I3=1,L)
      J=J+L/5+1
      WRITE(4*J,9480) QSCORE
 9480 FORMAT(10F8.5)
      GOTO 4000

C -- OPTION 49 TO CHANGE NAME OF BOX --
 4009 CALL GETHN(ID,HNAME)
      WRITE(6,9271)
      READ(5,9462) HNAME(ID)
      GOTO 4000
      END
```

```
      SUBROUTINE GETHN(ID,HNAME)
C SUBROUTINE GETHN INTERACTIVELY REQUESTS A BOX NAME AND RETURNS ITS
C ID NUMBER.
      DOUBLE PRECISION ALL,HNAME(40),A
      DATA ALL/6HALL    /
      ID=1
   10 WRITE(6,9000)
 9000 FORMAT(20H? ENTER BOX NAME -- )
      READ(5,9001) A
 9001 FORMAT(1A6)
      IF(A.EQ.ALL) RETURN
      CALL ATGET(A,ID,HNAME)
      IF(ID.EQ.0) GOTO 20
      RETURN
   20 WRITE(6,9002)
 9002 FORMAT(11H? BAD NAME )
      CALL PNAME(HNAME)
      GOTO 10
      END

      SUBROUTINE ATGET(ANAME,ID,HNAME)
C SUBROUTINE ATGET CHECKS A BOX NAME AGAINST THOSE CURRENTLY IN
C THE SYSTEM AND RETURNS ITS ID NUMBER (0 IF NOT VALID).
      DOUBLE PRECISION ANAME,HNAME(40)
      DO 10 I=1,40
      IF (ANAME.EQ.HNAME(I)) GOTO 20
   10 CONTINUE
      ID=0
      RETURN
   20 ID=I
      RETURN
      END

      SUBROUTINE SCREH(ID)
C SUBROUTINE SCREH IS THE MASTER SUBROUTINE FOR SCORING BOXES
      COMMON L1(100),QSCORE(100),L2(100),SCOREH(40),RULEH(40)
      COMMON IDEX(40,10)
      CALL MULTEV(ID)
  102 WRITE(6,9000) SCOREH(ID)
 9000 FORMAT(13H THE SCORE IS,1F10.5)
      RETURN
      END

      SUBROUTINE BOX(ID)
C SUBROUTINE BOX SCORES BOXES WHICH HAVE QUESTIONNAIRES AS INPUT.
      COMMON L1(100),QSCORE(100),L2(100),SCOREH(40)
      COMMON RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40),RESP(100)
      NAT=IDEX(ID,1)-49
      DO 101 I=2,NAT
  101 INDEX(I)=IDEX(ID,I)
      INDEX(1)=NAT-1
      DO 102 I=1,100
  102 RESP(I)=QSCORE(I)
```

```
      RULE=RULEH(ID)
      CALL TXTURE(ID,RULE,INDEX,RESP,SCORE)
      SCOREH(ID)=SCORE
      RETURN
      END

      SUBROUTINE MULTEV(ID)
C SUBROUTINE MULTEV IS THE HIGHEST LEVEL OF THE
C NESTED BOX SCORING ROUTINE
      COMMON LL(300),SCOREH(40),RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE1(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE GETQN(ID,QNAME,NUMQ)
C SUBROUTINE GETQN INTERACTIVEL Y ACCEPTS A QUESTIONNAIRE NAME
C AND RETURNS ITS ID NUMBER
      COMMON LOCQ(100)
      DIMENSION QNAME(100)
   40 WRITE(6,9000)
 9000 FORMAT(29H? ENTER QUESTIONAIRE NAME -- )
      READ(5,9001) A
 9001 FORMAT(1A4)
      ID=NQ(A)
      IF(ID.EQ.0) GOTO 10
      IF(ID.LT.0) RETURN
      I=LOCQ(ID)
      READ(1'I,9002) NUMQ
 9002 FORMAT(10X,I12)
      RETURN
   10 CALL PQNAME(QNAME,NUMQ)
      GOTO 40
      RETURN
      END

      FUNCTION YESNO(Z)
C FUNCTION YESNO RETURNS THE ANSWER TO A YES-NO QUESTION
C 0 FOR NO, 1 FOR YES.
      DATA X/1HN/
```

```
      DATA Y/1HY/
      YESNO=0.
   20 READ(5,9000) A
 9000 FORMAT(1A1)
      IF(X.EQ.A) RETURN
      IF (Y.NE.A) GOTO 10
      YESNO=1.
      RETURN
   10 WRITE(6,9001)
 9001 FORMAT(20H? TYPE YES OR NO -- )
      GOTO 20
      END

      FUNCTION GETNUM(ALOW,AHIGH,TYPE)
C FUNCTION GETNUM RETURNS A NUMBER ENTERED INTERACTIVELY
C AFTER CHECKING FOR THE APPROPRIATE RANGE AND TYPE.
      REAL GETNUM
   10 READ (5,*,ERR=20) GETNUM
      IF (TYPE.EQ.0.AND.GETNUM.NE.AINT(GETNUM)) GOTO 50
      IF (GETNUM.GE.ALOW.AND.GETNUM.LE.AHIGH) RETURN
      IF (TYPE.EQ.2) GOTO 40
   30 WRITE(6,100) ALOW,AHIGH
  100 FORMAT(26H? ENTER A NUMBER BETWEEN  ,1G10.5,4HAND ,1G10.5,3H --)
      GOTO 10
   20 IF (TYPE.NE.2) GOTO 30
   40 CONTINUE
      RETURN
   50 WRITE(6,101) ALOW,AHIGH
  101 FORMAT(26H? ENTER AN INTEGER BETWEEN,G10.5,4HAND ,G10.5,3H-- )
      GOTO 10
      END

      SUBROUTINE TXTURE (NUM,RULE,INDEX,RESP,SCORE)
C SUBROUTINE TXTURE COMPUTES A SCORE USING A SPECIFIED
C SCORING RULE.
      COMMON LL(100),QSCORE(100),LL1(580),QDEX(40)
      DIMENSION INDEX(1),RESP(1),TEXTS(6)
      DATA TEXTS/'HA','SA','AV','SC','OR','Q'/
      M=INDEX(1)
      N=M+1
      DO 5 I=1,6
      IF(RULE.EQ.TEXTS(I)) GOTO 6
    5 CONTINUE
      WRITE(6,900)
  900 FORMAT(38H  BAD RULE ENCOUNTERED COMPUTING SCORE   )
      RETURN
    6 GOTO (10,20,30,40,50,70) I
   10 SCORE=1.
      DO 11 I=2,N
   11 SCORE=SCORE*RESP(INDEX(I))
      RETURN
   20 SCORE=1.
      DO 21 I=2,N
   21 SCORE=SCORE*(RESP(INDEX(I))+1.)
      C=1./(2.**M-1.)
```

```
      SCORE=C*(SCORE-1.)
      RETURN

   30 SCORE=0.
      DO 31 I=2,N
   31 SCORE=SCORE+RESP(INDEX(I))
      SCORE=SCORE*(1./M)
      RETURN
   40 SCORE=1.
      DO 41 I=2,N
   41 SCORE=SCORE*(1.-.5*RESP(INDEX(I)))
      SCORE=(2.**M/(2.**M-1.))*(1.-SCORE)
   60 CONTINUE
      RETURN
   50 SCORE=1.
      DO 51 I=2,N
   51 SCORE=SCORE*(1.-RESP(INDEX(I)))
      SCORE=1.-SCORE
      RETURN
   70 Q=QDEX(NUM)
      ID=NQ(Q)
      A=WSCORE(ID)
      B=0.2
      DO 65 J=1,5
      IF(A.LT.B) GOTO 66
      B=B+0.2
   65 CONTINUE
   66 RULE=TEXTS(J)
      I=J
      GOTO 6
      END

      SUBROUTINE MULTE1(ID)
C SUBROUTINE MULTE1 IS THE SECOND LEVEL OF THE BOX SCORING
C SYSTEM. IT IS A CLONE OF MULTEV, AS ARE MULTE2, MULTE3, AND MULTE4.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE2(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END
```

```
      SUBROUTINE MULTE2(ID)
C SUBROUTINE MULTE2 IS THE THIRD LEVEL OF THE BOX SCORING SYSTEM
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE3(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE MULTE3(ID)
C SUBROUTINE MULTE3 IS THE FOURTH LEVEL OF THE BOX SCORING SYSTEM.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
     NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE4(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE MULTE4(ID)
C SUBROUTINE MULTE4 IS THE FIFTH LEVEL OF THE BOX SCORING SYSTEM.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
```

```
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTES(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END


      SUBROUTINE GETR(ILOC,RESP)
C SUBROUTINE GETR RETRIEVES THE RESPONSES FROM A SPECIFIED QUESTIONNAIRE
C LOCATION.
      INTEGER RESP(40)
      J=ILOC+1
      READ(2*J,9000) RESP
 9000 FORMAT(40(1X,1A1))
      RETURN
      END

      SUBROUTINE MULTES(ID)
C SUBROUTINE MULTES SIMPLY PRINTS AN ERROR MESSAGE AND RETURNS.
      WRITE(6,900)
      WRITE(6,901)
  900 FORMAT(54H  YOU HAVE HIT THE LOWEST LEVEL IN THE SCORING ROUTINE)
  901 FORMAT(45H  PLEASE TRY SCORING LOWER LEVEL BOXES FIRST  )
      RETURN
      END
```

```
      SUBROUTINE PRINTH(ID,HNAME)
C SUBROUTINE PRINTH PRINTS A GRAPHICAL DESCRIPTION OF THE HIERARCHY
C BELOW A GIVEN BOX.
      COMMON LXX(100),QSCORE(100),LXXX(100),SCOREH(40),RULEH(40)
      COMMON IDEX(40,10),QDEX(40),QNAME(100)
      DOUBLE PRECISION HNAME(40)
      DATA Q/1HQ/
      WRITE(6,9000) HNAME(ID)
 9000 FORMAT(/,25H  HIERARCHY DATA FOR BOX ,1A6,/)
      I1=IDEX(ID,1)+1
      K1=MOD(I1,50)
      WRITE(6,9001) HNAME(ID),RULEH(ID),SCOREH(ID),QDEX(ID)
 9001 FORMAT(5H BOX:,1A6,7H  RULE:,1A2,8H  SCORE:,1F6.3,
     15H   Q:,1A4)
      IF (I1.EQ.1) RETURN
      DO 10 I2=2,K1
      WRITE(6,9002)
 9002 FORMAT(7H?      )
      I3=IDEX(ID,I2)
      IF(K1.EQ.I1) GOTO 15
      WRITE(6,9003) QNAME(I3),QSCORE(I3)
 9003 FORMAT(17H  QUESTIONNAIRE: ,1A4,8H  SCORE:,1F6.3)
      GOTO 10
   15 WRITE(6,9001) HNAME(I3),RULEH(I3),SCOREH(I3),QDEX(I3)
      I4=IDEX(I3,1)+1
      K2=MOD(I4,50)
      IF(I4.EQ.1) GOTO 10
      DO 40 I5=2,K2
      WRITE(6,9002)
      WRITE(6,9002)
      I8=IDEX(I3,I5)
      IF(K2.EQ.I4) GOTO 25
      WRITE(6,9003) QNAME(I8),QSCORE(I8)
      GOTO 40
   25 WRITE(6,9001) HNAME(I8),RULEH(I8),SCOREH(I8),QDEX(I8)
      I6=IDEX(I8,1)+1
      K3=MOD(I6,50)
      IF(I6.EQ.1) GOTO 45
      DO 30 I7=2,K3
      DO 11 J1=1,3
   11 WRITE(6,9002)
      J9=IDEX(I8,I7)
      IF(K3.EQ.I6) GOTO 35
      WRITE(6,9003) QNAME(J9),QSCORE(J9)
      GOTO 30
   35 WRITE(6,9001) HNAME(J9),RULEH(J9),SCOREH(J9),QDEX(J9)
   30 CONTINUE
   45 CONTINUE
   40 CONTINUE
   10 CONTINUE
      RETURN
      END

      SUBROUTINE SETH(ID,ISET)
C SUBROUTINE SETH IDENTIFIES WHAT BOX SCORES WILL BE CHANGED
```

```
C WHEN A LOW LOVEL BOX'S SCORE IS CHANGED AND REINITIALIZES THEM.
      COMMON LXX(300),SCOREH(40),RULEH(40),IDEX(40,10)
      DIMENSION ISET(40)
      IE=ID
      DO 20 I3=1,10
      ICK=0
      DO 40 I1=1,40
      I2=IDEX(I1,1)+1
      IF (I2.EQ.1.OR.I2.GE.50) GOTO 40
      DO 10 I4=2,I2
      I5=IDEX(I1,I4)
      IF(I5.NE.IE) GOTO 10
      SCOREH(I1)=-1.
      IE=I1
      GOTO 20
   10 CONTINUE
   40 CONTINUE
      IF (ICK.EQ.0) GOTO 50
   20 CONTINUE
   50 RETURN
      END

      FUNCTION NQ(ANAME)
C FUNCTION NQ RETURNS THE NUMBER OF THE PASSED QUESTIONNAIRE NAME.
      COMMON LL(820),QNAME(100)
      DATA ALL/4HALL /
      DO 10 I=1,100
      IF (ANAME.EQ.QNAME(I)) GOTO 20
   10 CONTINUE
      IF(ANAME.NE.ALL) GOTO 30
      NQ=-1
      RETURN
   30 WRITE(6,9003)
 9003 FORMAT(10H  BAD NAME)
      NQ=0
      RETURN
   20 NQ=I
      RETURN
      END
```

```
      SUBROUTINE SCOREQ(ID)
C SUBROUTINE SCOREQ SCORES QUESTIONNAIRES.
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      DIMENSION IRESP(40),RESP(40),RULE(10),IWRST(40),WEIGHT(40)
      DIMENSION SCORE(40),SCO(40),INDEX(40)
      INTEGER QDEX(10,40)
      DATA IBEST/1HA/
      BEST=FLOAT(IBEST)
      CALL GETQ(LOCQ(ID),RULE,IWRST,WEIGHT,NUMQ,NUMGP,QDEX)
      CALL GETR(LOCR(ID),IRESP)
      DO 10 I=1,NUMQ
      IF (IWRST(I).EQ.IBEST) GOTO 10
      RESP(I)=(BEST-FLOAT(IRESP(I)))/(BEST-FLOAT(IWRST(I)))
      RESP(I)=1.-WEIGHT(I)*RESP(I)
   10 CONTINUE
      DO 150 I=1,10
  150 SCORE(I)=-1.
      DO 140 I1=1,10
      FLAG=0.
      DO 100 I=1,NUMGP
      J1=QDEX(I,1)+1
      DO 110 J2=2,J1
      J3=QDEX(I,J2)
      IF(J3.GE.100) GOTO 115
      IF(SCORE(J3).LT.0) GOTO 120
  115 CONTINUE
  110 CONTINUE
      DO 130 J4=2,J1
      J5=QDEX(I,J4)
      IF(J5.LT.100) GOTO 160
      J5=J5-100
      SCO(J4)=RESP(J5)
      GOTO 170
  160 SCO(J4)=SCORE(J5)
  170 INDEX(J4)=J4
  130 CONTINUE
      FLAG=1.
      INDEX(1)=QDEX(I,1)
      CALL TXTURE(I,RULE(I),INDEX,SCO,SCORE(I))
      IF(I.EQ.1) GOTO 145
  120 CONTINUE
  100 CONTINUE
      IF(FLAG.EQ.0) GOTO 145
      IF(SCORE(1).GE.0) GOTO 145
  140 CONTINUE
  145 QSCORE(ID)=SCORE(1)
      WRITE(6,9000) QSCORE(ID)
 9000 FORMAT(13H  THE SCORE =,1F6.3)
      RETURN
      END

      SUBROUTINE GETQ(LOC,RULE,IWRST,WEIGHT,NUMQU,NUMGP,QDEX)
C SUBROUTINE GETQ READS THE STRUCTURE OF A QUESTIONNAIRE OFF DISC FILE 1
```

```
      INTEGER QDEX(10,40),ISAVE(40),IWRST(40)
      DIMENSION RULE(10),WEIGHT(40)
      J=LOC
      READ(1'J,9000) X,NUMQU,NUMGP
 9000 FORMAT(1A4,6X,1I2,8X,1I2)
      J=J+1
      READ(1'J,9002) IWRST
 9002 FORMAT(40(1X,1A1))
 9001 FORMAT(8F5.3)
      DO 50 I2=1,NUMGP
      J=J+1
      READ(1'J,9003) NAME,NUM,R
 9003 FORMAT(2(1I2,8X),1A2)
      NAME=NAME-49
      QDEX(NAME,1)=NUM
      RULE(NAME)=R
      J=J+1
      READ(1'J,9004) (ISAVE(J2),J2=1,NUM)
      DO 50 I=1,NUM
      I1=I+1
      IF(ISAVE(I).LT.50) QDEX(NAME,I1)=ISAVE(I)+100
      IF(ISAVE(I).GE.50) QDEX(NAME,I1)=ISAVE(I)-49
   50 CONTINUE
      J=J+1
      READ (1'J,9001)(WEIGHT(I2),I2=1,8)
      IF(WEIGHT(1).GT.1) GOTO 70
      IF(NUMQU.LE.8) RETURN
      J1=J+1
      READ(1'J1,9001)(WEIGHT(I2),I2=9,NUMQU)
      RETURN
   70 DO 100 J2=1,NUMQU
  100 WEIGHT(J2)=WEIGHT(2)
 9004 FORMAT(40I2)
      RETURN
      END


      SUBROUTINE PRINTQ(ID,QNAME)
C SUBROUTINE PRINTQ PRINTS THE STRUCTURE FOR A QUESTIONNAIRE.
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      INTEGER QDEX(10,40)
      DIMENSION RULE(10),RESP(40),QNAME(100),IWRST(40),WEIGHT(40)
      DIMENSION IRESP(40),X(40),Y(40)
      DATA IBEST/1HA/
      CALL GETQ(LOCQ(ID),RULE,IWRST,WEIGHT,NUMQU,NUMGP,QDEX)
      BEST=FLOAT(IBEST)
      CALL GETR(LOCR(ID),IRESP)
      WRITE(6,8999)
      WRITE(6,9000) QNAME(ID)
 8999 FORMAT (//,20X,22H QUESTIONAIRE DATA FOR)
 9000 FORMAT(23X,13HQUESTIONAIRE ,1A4)
      WRITE(6,9002) QSCORE(ID),RULE(1)
 9002 FORMAT(8X,16HOVERALL SCORE = ,1F10.5,5X,7HRULE : ,1A4,//)
      DO 5 I=1,40
      IF (IWRST(I).NE.IBEST) GOTO 4
```

```
      X(I)=0.
      GOTO 6
    4 X(I)=1-(BEST-FLOAT(IRESP(I)))/(BEST-FLOAT(IWRST(I)))
    6 Y(I)=1.-WEIGHT(I)*(1-X(I))
    5 CONTINUE
      I=1
      J1=50
      WRITE(6,9004) J1,RULE(1)
 9004 FORMAT(7H  BOX: ,1I2,8H  RULE: ,1A2)
      I1=QDEX(I,1)+1
      IF(I1.EQ.1) GOTO 10
      DO 20 I2=2,I1
      WRITE(6,9005)
 9005 FORMAT(7H?      )
      I3=QDEX(I,I2)
      IF(I3.GE.100) GOTO 30
      J1=I3+49
      WRITE(6,9004) J1,RULE(I3)
      I4=QDEX(I3,1)+1
      IF(I4.EQ.1) GOTO 20
      DO 40 I5=2,I4
      WRITE(6,9005)
      WRITE(6,9005)
      I8=QDEX(I3,I5)
      IF(I8.GE.100) GOTO 50
      J1=I8+49
      WRITE(6,9004) J1,RULE(I8)
      I6=QDEX(I8,1)+1
      IF(I6.EQ.1) GOTO 45
      DO 60 I7=2,I6
      DO 11 J2=1,3
   11 WRITE(6,9005)
      J9=QDEX(I8,I7)
      IF(J9.GE.100) GOTO 70
      J1=J9+49
      WRITE(6,9004) J1,RULE(J9)
      GOTO 60
   70 J1=J9-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
 9006 FORMAT(4H  Q=,1I2,7H  RESP=,1F6.3,3H W=,1F6.3,3H S=,1F6.3)
   60 CONTINUE
      GOTO 45
   50 J1=I8-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
   45 CONTINUE
   40 CONTINUE
      GOTO 20
   30 J1=I3-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
   20 CONTINUE
   10 RETURN
      END
```

```
      SUBROUTINE HPR(ID,HNAME)
C SUBROUTINE HPR PRINTS A PICTURE OF A PORTION OF THE HIERARCHY.
      COMMON LXX(300),SCCREH(40),RULEH(40),IDEX(40,10)
      DIMENSION DOT(20,5),IPR(20,5)
      DOUBLE PRECISION HNAME(40)
      DATA STAR/4H****/BLANK/4H    /
      DO 1 I=1,20
      DO 1 J=1,5
      IPR(I,J)=0
    1 DOT(I,J)=BLANK
      LEV=1
      I1=IDEX(ID,1)+1
      IPR(LEV,1)=ID
      IF (I1.EQ.1.OR.I1.GT.50) GOTO 5
    6 DO 10 I2=2,I1
      I3=IDEX(ID,I2)
      IPR(LEV,2)=I3
      I4=IDEX(I3,1)+1
      IF(I4.EQ.1.OR.I4.GT.50) GOTO 15
   16 DO 20 I5=2,I4
      I6=IDEX(I3,I5)
      IPR(LEV,3)=I6
      I7=IDEX(I6,1)+1
      IF(I7.EQ.1.OR.I7.GT.50) GOTO 25
   26 DO 30 I8=2,I7
      I9=IDEX(I6,I8)
      IPR(LEV,4)=I9
      I10=IDEX(I9,1)+1
      IF(I10.EQ.1.OR.I10.GT.50) GOTO 501
      DO 500 I11=2,I10
      IPR(LEV,5)=IDEX(I9,I11)
  500 LEV=LEV+1
      GOTO 30
  501 LEV=LEV+1
   30 CONTINUE
      GOTO 20
   25 LEV=LEV+1
   20 CONTINUE
      GOTO 10
   15 LEV=LEV+1
   10 CONTINUE
    5 CONTINUE
      DO 60 I=1,19
      DO 60 J=1,4
      I1=IPR(I,J)
      IF(I1.EQ.0) GOTO 60
      I2=IDEX(I1,1)+1
      IF (I2.LE.2.OR.I2.GT.50) GOTO 60
      I5=I+1
      DO 61 I3=I5,19
      J1=J+1
      DOT(I3,J)=STAR
      IF(IPR(I3,J1).EQ.IDEX(I1,I2)) GOTO 62
   61 CONTINUE
   62 CONTINUE
```

```
   60 CONTINUE
      WRITE(6,9000) HNAME(ID)
 9000 FORMAT(32H  HIERARCHY INFORMATION FOR BOX ,1A6,//)
 9001 FORMAT(11H?**********)
 9002 FORMAT(3H?* ,1A6,2H *)
 9003 FORMAT(4H?*S=,1F6.3,1H*)
 9004 FORMAT(6H?*RULE: ,1A2,1H*)
 9005 FORMAT(3H?**)
 9006 FORMAT(1H?1A1)
 9007 FORMAT(3H?  )
 9008 FORMAT(2H? )
 9009 FORMAT(11H?          )
 9011 FORMAT(3H?  ,1A1,2H  )
 9012 FORMAT(',2H?  )
      I=LEV-1
      WRITE(6,9008)
      DO 40 LEV=1,I
      LE=LEV+1
      MAX=1
      DO 101 III=1,5
  101 IF(IPR(LEV,III).NE.0.OR.DOT(LEV,III).EQ.STAR) MAX=III
      DO 41 I1=1,MAX
      IF(IPR(LEV,I1).NE.0) GOTO 39
      WRITE(6,9009)
      GOTO 41
   39 WRITE(6,9001)
   41 WRITE(6,9011) DOT(LEV,I1)
      WRITE(6,9012)
      DO 42 I2=1,MAX
      I3=IPR(LEV,I2)
      IF(I3.EQ.0) GOTO 43
      WRITE(6,9002) HNAME(I3)
      GOTO 38
   43 WRITE(6,9009)
   38 WRITE(6,9011) DOT(LEV,I2)
   44 CONTINUE
   42 CONTINUE
      WRITE(6,9012)
      DO 66 I1=1,4
      IF(DOT(2,I1).EQ.STAR) DOT(1,I1)=STAR
   66 IF(DOT(1,I1).EQ.STAR.AND.MAX.LT.I1) MAX=I1
      DO 45 I2=1,MAX
      I3=IPR(LEV,I2)
      IF(I3.EQ.0) GOTO 47
      IF(I2.GT.1) WRITE(6,9005)
      WRITE(6,9003) SCOREH(I3)
      IF(I2.EQ.4) GOTO 45
      I4=I2+1
      IF(IPR(LEV,I4).GT.0) GOTO 46
      WRITE(6,9007)
      GOTO 65
   46 WRITE(6,9005)
      WRITE(6,9006) STAR
      GOTO 45
   47 WRITE(6,9009)
```

```
      WRITE(6,9007)
      IF(I2.GT.1) WRITE(6,9007)
   65 WRITE(6,9006) DOT(LEV,I2)
   45 CONTINUE
      WRITE(6,9012)
      DO 49 I2=1,MAX
      I3=IPR(LEV,I2)
      IF(I3.EQ.0) GOTO 50
      WRITE(6,9004) RULEH(I3)
      GOTO 51
   50 WRITE(6,9009)
   51 WRITE(6,9011) DOT(LE,I2)
   49 CONTINUE
      WRITE(6,9012)
      DO 52 I1=1,MAX
      IF(IPR(LEV,I1).NE.0) GOTO 53
      WRITE(6,9009)
      GOTO 55
   53 WRITE(6,9001)
   55 WRITE(6,9011) DOT(LE,I1)
   52 CONTINUE
      WRITE(6,9012)
      DO 54 I1=1,4
      WRITE(6,9009)
   54 WRITE(6,9011) DOT(LE,I1)
      WRITE(6,9012)
   40 CONTINUE
      RETURN
      END

      SUBROUTINE PNAME(HNAME)
C SUBROUTINE PNAME PRINTS THE NAME OF THE CURRENT HIERARCHY BOXES.
      DOUBLE PRECISION HNAME(40)
       WRITE(6,9470)
 9470 FORMAT(38H  THE CURRENT HIERARCHY INCLUDES BOXES ,/)
      DO 470 I=1,5
      I1=8*(I-1)+1
      I2=I1+7
  470 WRITE(6,9471)(HNAME(I3),I3=I1,I2)
 9471 FORMAT(8(1X,1A6,1X))
      RETURN
      END

      SUBROUTINE PQNAME(QNAME,NUMQ)
C SUBROUTINE PQNAME PRINTS THE NAMES OF THE CURRENT QUESTIONNAIRES.
      DIMENSION QNAME(100)
      WRITE(6,9000)
 9000 FORMAT(33H  THE CURRENT QUESTIONNAIRES ARE: ,/)
   10 WRITE(6,9001)(QNAME(I2),I2=1,NUMQ)
 9001 FORMAT(2X,10(1A4,2X))
      RETURN
      END
```

```
C QUEASI -- A PROGRAM FOR EVALUATION OF SAFEGUARDS QUESTIONNAIRES
C AND HIERARCHIES
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      COMMON SCOREH(40),RULEH(40),IDEX(40,10),QDEX(40),QNAME(100)
      DOUBLE PRECISION BNAME(40),BLANK,NOMO,HNAME(40),ANAME
      INTEGER LOCHS(5),ISET(40),FLAG(10),LOCH(5),IRESP(40),IBEST(40)
      REAL SCORE(40),WEIGHT(40),RULE(10),TEXTS(7)
      DATA TEXTS/2HHA,2HSA,2HAV,2HSD,2HOR,1HQ,2HOR/
      DATA IAA/1HA/,NOMO/6HNOMORE/BLANK/6H      /
      AAA=FLOAT(IAA)
      DO 399 I=1,40
  399 SCOREH(I)=-1.
      J=2
C READ QUESTIONNAIRE LOCATIONS AND NAMES
      READ(2,9464) NUMQ
      READ(2,9465) (LOCR(I2),I2=1,NUMQ)
      READ(1,9464) NUMQ
      READ(1,9465) (LOCQ(I1),I1=1,NUMQ)
      DO 30 J=1,NUMQ
   30 READ(2,9727) QNAME(J)
 9727 FORMAT(1A4/)
      REWIND 2
      READ(2,9464) NUMQ
      READ(2,9465) (LOCR(I2),I2=1,NUMQ)
      DO 10 I=1,100
   10 QSCORE(I)=-1.
      DO 20 I=1,10
   20 FLAG(I)=0
C SELECT INITIAL OPTION
 1000 WRITE(6,9100)
 9100 FORMAT(54H  SELECT 1- HIERARCHIES, 2- QUESTIONAIRES, 3- STOP -- )
      IOP=GETNUM(1.,3.,2.)
 1001 IF (IOP.EQ.3) STOP
      GOTO (4000,2000) IOP
C REVIEW OPTION SELECTION AND BRANCH TO PROPER OPTION
      IOPT=IOP
 1100 CONTINUE
      IF (IOPT.GE.1.AND.IOPT.LE.3) GOTO 1101
      IF(IOPT.GE.21.AND.IOPT.LE.26) GOTO 1102
      IF (IOPT.GE.41.AND.IOPT.LE.50) GOTO 1102
      GOTO (1000,230,1000,402) IOP
 1101 IOP=IOPT
      GOTO 1001
 1102 IOP=INT(FLOAT(IOPT)/10.)
      IF(IOP*10.EQ.IOPT) GOTO 1001
      IOPT=IOPT-IOP*10
      GOTO (1000,201,1000,404) IOP
      GOTO 1000
C PRINT MENU AND GET QUESTIONNAIRE OPTION
 2000 IF (FLAG(2).EQ.1) GOTO 210
  230 WRITE(6,9200)
      WRITE(6,9201)
      WRITE(6,9202)
      WRITE(6,9203)
      WRITE(6,9204)
```

```
      WRITE(6,9205)
 9200 FORMAT(/,14H  SELECT ONE: ,/)
 9201 FORMAT(51H  21- COMPUTE SCORES      22- PRINT SCORES          )
 9202 FORMAT(51H  23- SET SCORES          24- REVISE WEIGHTS        )
 9203 FORMAT(51H  25- REVISE RULES        26- REVISE RESPONSES      )
 9207 FORMAT(51H  27- REVISE NAMES        28- PRINT NAMES           )
 9204 FORMAT(51H            29- NO MORE REVISIONS                   )
 9205 FORMAT(/,9H  WHICH   )
      GOTO 220
  210 WRITE(6,9206)
 9206 FORMAT(18H  SELECT 21-29 -- )
  220 IOPT=GETNUM(21.,29.,2.)
      IF (IOPT.LT.21.OR.IOPT.GT.29) GOTO 1100
      IOPT=IOPT-20
  201 FLAG(2)=1
C BRANCH TO PROPER OPTION
     GOTO (2001,2000,2000,2000,2000,2000,2000,2008,1000) IOPT

C -- OPTION 21 TO COMPUTE A QUESTIONNAIRE SCORE --
 2001 CONTINUE
  214 DO 211 I=1,NUMQ
      WRITE(6,9301) QNAME(I)
 9301 FORMAT(16H  QUESTIONNAIRE ,1A4,1H:)
  211 CALL SCOREQ(I)
      GOTO 2000

 9503 FORMAT(33H  ENTER RULE (HA,SA,AV,SO,OR) -- )
 9504 FORMAT(1A2)
 9505 FORMAT(25H  BAD RULE, TRY AGAIN -- )
 9271 FORMAT(15H  ENTER NAME -- )
 9272 FORMAT(1A4)
C -- PRINT AND SELECT HIERARCHY MANIPULATION OPTIONS --
 2008 CALL PQNAME(QNAME,NUMQ)
      GOTO 2000

 4000 IF (FLAG(5).EQ.0) GOTO 4006
      IF(FLAG(4).EQ.1) GOTO 401
  402 WRITE(6,9400)
      WRITE(6,9401)
      WRITE(6,9402)
      WRITE(6,9403)
      WRITE(6,9404)
      WRITE(6,9405)
      WRITE(6,9407)
      WRITE(6,9205)
 9400 FORMAT(/,14H  SELECT ONE: ,/)
 9401 FORMAT(51H  41- COMPUTE SCORES      42- PRINT DATA            )
 9402 FORMAT(51H  43- ASSIGN SCORES       44- REVISE DELAY/RESP     )
 9403 FORMAT(51H  45- REVISE RULES        46- SELECT NEW HIERARCHY  )
 9404 FORMAT(51H  47- PRINT BOX NAMES     48- FILE HIERARCHY DATA   )
 9405 FORMAT(51H  49- CHANGE BOX NAME     50- PRINT HIERARCHY       )
 9407 FORMAT(51H                   51- NO MORE REVISIONS            )
      GOTO 403
  401 WRITE(6,9406)
 9406 FORMAT(18H  SELECT 41-51 -- )
```

```
  403  IOPT=GETNUM(41.,47.,2.)
       IF(IOPT.LT.41.OR.IOPT.GT.51) GOTO 1100
       IOPT=IOPT-40
  404  FLAG(4)=1
C BRANCH TO PROPER HIERARCHY OPTION
       IF(FLAG(5).EQ.0) GOTO 4006
       GOTO(4001,4002,4003,4004,4005,4006,4007,4008,4009,4002,1000) IOPT

C -- OPTION 41 TO SCORE A HIERARCHY BOX --
 4001  CALL GETHN(ID,HNAME)
       CALL SCREH(ID)
       CALL SETH(ID,ISET)
       ISET(ID)=0
       GOTO 4000

C -- OPTION 42 TO PRINT HIERARCHY DATA --
 4002  CALL GETHN(ID,HNAME)
       IF(IOPT.EQ.2) CALL PRINTH(ID,HNAME)
       IF (IOPT.EQ.10) CALL HPR(ID,HNAME)
       GOTO 4000

C -- OPTION 43 TO ASSIGN HIERARCHY BOX SCORES --
 4003  CALL GETHN(ID,HNAME)
       WRITE(6,9430)
       CALL SETH(ID,ISET)
 9430  FORMAT(10H  SCORE = )
       SCOREH(ID)=GETNUM(-1.,1.,1.)
       IF(SCOREH(ID).LT.0) ISET(ID)=0
       IF(SCOREH(ID).GE.0) ISET(ID)=1
       GOTO 4000

C -- OPTION 44 TO REVISE DELAY/RESPONSE RULE. NOT CURRENTLY USED
 4004  CALL GETHN(ID,HNAME)
       IF(RULEH(ID).EQ.TEXTS(7)) GOTO 440
       WRITE(6,9440)
 9440  FORMAT(28H BOX DOES NOT USE DELAY/RESP)
       GOTO 4000
  440  CONTINUE
       GOTO 4000

C -- OPTION 45 TO REVISE SCORING RULES --
 4005  CALL GETHN(ID,HNAME)
       WRITE(6,9503)
  450  READ(5,9504) R
       DO 451 I=1,6
       IF (R.EQ.TEXTS(I)) GOTO 452
  451  CONTINUE
       WRITE(6,9505)
       GOTO 450
  452  RULEH(ID)=R
       SCOREH(ID)=-1.
       CALL SETH(ID,ISET)
       GOTO 4000

C -- OPTION 46 TO ENTER A NEW HIERARCHY INTO THE SYSTEM --
```

```
 4006  WRITE(6,9460)
       FLAG(5)=1
       READ(3,9464) NUMH
       READ(3,9465) (LOCH(I),I=1,NUMH)
 9464  FORMAT(1I2)
 9465  FORMAT(20I4)
       READ(4,9464) NUMH
       READ(4,9465)(LOCHS(I),I=1,NUMH)
 9460  FORMAT(28H  ENTER HIERARCHY NUMBER -- )
       HNUM=GETNUM(1.,FLOAT(NUMH),1.)
       DO 465 I=1,40
       ISET(I)=0
       DO 466 I1=1,10
  466  IDEX(I,I1)=0
       HNAME(I)=BLANK
  465  SCOREH(I)=-1.
       L=0
       LOC=LOCH(HNUM)
C BEGIN BY READING INFO FOR FIRST BOX
  460  READ(3,9461) ANAME,NUM,R,Q
 9461  FORMAT(1A6,4X,1I2,8X,1A2,8X,1A4)
       IF(ANAME.EQ.NOMO) GOTO 468
       CALL ATGET(ANAME,ID,HNAME)
       IF(ID.GT.0) GOTO 464
       L=L+1
       ID=L
       HNAME(ID)=ANAME
  464  IF(NUM.EQ.0) GOTO 463
       QDEX(ID)=Q
       RULEH(ID)=R
  461  IDEX(ID,1)=NUM
       IF(NUM.EQ.0) GOTO 463
       IF(NUM.GT.50) GOTO 4601
       DO 462 J=1,NUM
       J1=J+1
       LOC=LOC+1
       READ(3,9462) ANAME
       CALL ATGET(ANAME,IE,HNAME)
       IF(IE.GT.0) GOTO 462
       L=L+1
       IE=L
       HNAME(L)=ANAME
  462  IDEX(ID,J1)=IE
 9462  FORMAT(1A6)
  463  LOC=LOC+1
       GOTO 460
 4601  NUM=NUM-50
       DO 4602 J=1,NUM
       J1=J+1
       LOC=LOC+1
       READ(3,9272) ANAM
 4602  IDEX(ID,J1)=NQ(ANAM)
       LOC=LOC+1
       GOTO 460
  468  CONTINUE
```

```
      J=LOCHS(HNUM)
      READ(4,9463)((BNAME(I1),SCORE(I1)),I1=1,L)
      J=J+L/5+1
      READ(4,9480) GSCORE
 9463 FORMAT(5(1A6,1F6.3))
      DO 469 K=1,L
      CALL ATGET(BNAME(K),ID,HNAME)
      IF (ID.EQ.0) GOTO 469
      SCOREH(ID)=SCORE(K)
  469 CONTINUE
      DO 459 I=1,40
  459 IF(IDEX(I,1).EQ.0) RULEH(I)=BLANK
      GOTO 4000

C -- OPTION 47 TO PRINT CURRENT BOX NAMES --
 4007 CALL PNAME(HNAME)
      GOTO 4000

C -- OPTION 48 TO FILE HIERARCHY DATA --
 4008 LOC=LOCH(HNUM)
      REWIND 3
      REWIND 4
      READ(3,9464) NUMH
      READ(3,9465) (LOCH(I),I=1,NUMH)
      READ(4,9464) NUMH
      READ(4,9465) (LOCHS(I),I=1,NUMH)
       DO 480 I=1,L
       IF(HNAME(I).EQ.BLANK) GOTO 480
       WRITE(3,9461) HNAME(I),IDEX(I,1),RULEH(I),QDEX(I)
       I4=IDEX(I,1)
       IF(I4.EQ.0) GOTO 480
       LOC=LOC+1
       I1=MOD(IDEX(I,1),50)+1
       DO 481 I2=2,I1
       I3=IDEX(I,I2)
       IF(I4.LE.50) GOTO 483
       WRITE(3,9272) GNAME(I3)
       GOTO 481
  483  WRITE(3,9462) HNAME(I3)
  481  LOC=LOC+1
  480  CONTINUE
       WRITE(3,9462) NOMO
       J=LOCHS(HNUM)
       WRITE(4,9463)(HNAME(I3),SCOREH(I3),I3=1,L)
       J=J+L/5+1
       WRITE(4,9480) GSCORE
 9480  FORMAT(10F8.5)
       GOTO 4000

C -- OPTION 49 TO CHANGE NAME OF BOX --
 4009  CALL GETHN(ID,HNAME)
       WRITE(6,9271)
       READ(5,9462) HNAME(ID)
       GOTO 4000
       END
```

```
      SUBROUTINE GETHN(ID,HNAME)
C SUBROUTINE GETHN INTERACTIVELY REQUESTS A BOX NAME AND RETURNS ITS
C ID NUMBER.
      DOUBLE PRECISION ALL,HNAME(40),A
      DATA ALL/6HALL   /
      ID=1
   10 WRITE(6,9000)
 9000 FORMAT(20H  ENTER BOX NAME -- )
      READ(5,9001) A
 9001 FORMAT(1A6)
      IF(A.EQ.ALL) RETURN
      CALL ATGET(A,ID,HNAME)
      IF(ID.EQ.0) GOTO 20
      RETURN
   20 WRITE(6,9002)
 9002 FORMAT(11H  BAD NAME )
      CALL PNAME(HNAME)
      GOTO 10
      END

      SUBROUTINE ATGET(ANAME,ID,HNAME)
C SUBROUTINE ATGET CHECKS A BOX NAME AGAINST THOSE CURRENTLY IN
C THE SYSTEM AND RETURNS ITS ID NUMBER (0 IF NOT VALID).
      DOUBLE PRECISION ANAME,HNAME(40)
      DO 10 I=1,40
      IF (ANAME.EQ.HNAME(I)) GOTO 20
   10 CONTINUE
      ID=0
      RETURN
   20 ID=I
      RETURN
      END

      SUBROUTINE SCREH(ID)
C SUBROUTINE SCREH IS THE MASTER SUBROUTINE FOR SCORING BOXES
      COMMON L1(100),QSCORE(100),L2(100),SCOREH(40),RULEH(40)
      COMMON IDEX(40,10)
      CALL MULTEV(ID)
  102 WRITE(6,9000) SCOREH(ID)
 9000 FORMAT(13H THE SCORE IS,1F10.5)
      RETURN
      END

      SUBROUTINE BOX(ID)
C SUBROUTINE BOX SCORES BOXES WHICH HAVE QUESTIONNAIRES AS INPUT.
      COMMON L1(100),QSCORE(100),L2(100),SCOREH(40)
      COMMON RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40),RESP(100)
      NAT=IDEX(ID,1)-49
      DO 101 I=2,NAT
  101 INDEX(I)=IDEX(ID,I)
      INDEX(1)=NAT-1
      DO 102 I=1,100
  102 RESP(I)=QSCORE(I)
```

```
      RULE=RULEH(ID)
      CALL TXTURE(ID,RULE,INDEX,RESP,SCORE)
      SCOREH(ID)=SCORE
      RETURN
      END

      SUBROUTINE MULTEV(ID)
C SUBROUTINE MULTEV IS THE HIGHEST LEVEL OF THE
C NESTED BOX SCORING ROUTINE
      COMMON LL(300),SCOREH(40),RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE1(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE GETQN(ID,QNAME,NUMQ)
C SUBROUTINE GETQN INTERACTIVEL Y ACCEPTS A QUESTIONNAIRE NAME
C AND RETURNS ITS ID NUMBER
      COMMON LOCQ(100)
      DIMENSION QNAME(100)
   40 WRITE(6,9000)
 9000 FORMAT(29H  ENTER QUESTIONAIRE NAME -- )
      READ(5,9001) A
 9001 FORMAT(1A4)
      ID=NQ(A)
      IF(ID.EQ.0) GOTO 10
      IF(ID.LT.0) RETURN
      I=LOCQ(ID)
     REWIND 1
     READ(1,9002) NUMQ
 9002 FORMAT(10X,1I2)
      RETURN
   10 CALL PQNAME(QNAME,NUMQ)
      GOTO 40
      RETURN
      END

      FUNCTION YESNO(Z)
C FUNCTION YESNO RETURNS THE ANSWER TO A YES-NO QUESTION
C 0 FOR NO, 1 FOR YES.
```

```
      DATA X/1HN/
      DATA Y/1HY/
      YESNO=0.
   20 READ(5,9000) A
 9000 FORMAT(1A1)
      IF(X.EQ.A) RETURN
      IF (Y.NE.A) GOTO 10
      YESNO=1.
      RETURN
   10 WRITE(6,9001)
 9001 FORMAT(20H  TYPE YES OR NO -- )
      GOTO 20
      END

      FUNCTION GETNUM(ALOW,AHIGH,TYPE)
C FUNCTION GETNUM RETURNS A NUMBER ENTERED INTERACTIVELY
C AFTER CHECKING FOR THE APPROPRIATE RANGE AND TYPE.
      REAL GETNUM
   10 READ (5,*,ERR=20) GETNUM
      IF (TYPE.EQ.0.AND.GETNUM.NE.AINT(GETNUM)) GOTO 50
      IF (GETNUM.GE.ALOW.AND.GETNUM.LE.AHIGH) RETURN
      IF (TYPE.EQ.2) GOTO 40
   30 WRITE(6,100) ALOW,AHIGH
  100 FORMAT(26H  ENTER A NUMBER BETWEEN  ,1G10.5,4HAND ,1G10.5,3H --)
      GOTO 10
   20 IF (TYPE.NE.2) GOTO 30
   40 CONTINUE
      RETURN
   50 WRITE(6,101) ALOW,AHIGH
  101 FORMAT(26H  ENTER AN INTEGER BETWEEN,G10.5,4HAND ,G10.5,3H-- )
      GOTO 10
      END

      SUBROUTINE TXTURE (NUM,RULE,INDEX,RESP,SCORE)
C SUBROUTINE TXTURE COMPUTES A SCORE USING A SPECIFIED
C SCORING RULE.
      COMMON LL(100),QSCORE(100),LL1(580),QDEX(40)
      DIMENSION INDEX(1),RESP(1),TEXTS(6)
     DATA TEXTS/2HHA,2HSA,2HAV,2HSO,2HOR,1HQ/
      M=INDEX(1)
      N=M+1
      DO 5 I=1,6
      IF(RULE.EQ.TEXTS(I)) GOTO 6
    5 CONTINUE
      WRITE(6,900)
  900 FORMAT(38H  BAD RULE ENCOUNTERED COMPUTING SCORE  )
      RETURN
    6 GOTO (10,20,30,40,50,70) I
   10 SCORE=1.
      DO 11 I=2,N
   11 SCORE=SCORE*RESP(INDEX(I))
      RETURN
   20 SCORE=1.
      DO 21 I=2,N
   21 SCORE=SCORE*(RESP(INDEX(I))+1.)
```

```
      C=1./(2.**M-1.)
      SCORE=C*(SCORE-1.)
      RETURN

   30 SCORE=0.
      DO 31 I=2,N
   31 SCORE=SCORE+RESP(INDEX(I))
      SCORE=SCORE*(1./M)
      RETURN
   40 SCORE=1.
      DO 41 I=2,N
   41 SCORE=SCORE*(1.-.5*RESP(INDEX(I)))
      SCORE=(2.**M/(2.**M-1.))*(1.-SCORE)
   60 CONTINUE
      RETURN
   50 SCORE=1.
      DO 51 I=2,N
   51 SCORE=SCORE*(1.-RESP(INDEX(I)))
      SCORE=1.-SCORE
      RETURN
   70 Q=GDEX(NUM)
      ID=NG(Q)
      A=GSCORE(ID)
      B=0.2
      DO 65 J=1,5
      IF(A.LT.B) GOTO 66
      B=B+0.2
   65 CONTINUE
   66 RULE=TEXTS(J)
      I=J
      GOTO 6
      END

      SUBROUTINE MULTE1(ID)
C SUBROUTINE MULTE1 IS THE SECOND LEVEL OF THE BOX SCORING
C SYSTEM. IT IS A CLONE OF MULTEV, AS ARE MULTE2, MULTE3, AND MULTE4.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE2(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
```

```
      END

      SUBROUTINE MULTE2(ID)
C SUBROUTINE MULTE2 IS THE THIRD LEVEL OF THE BOX SCORING SYSTEM
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE3(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE MULTE3(ID)
C SUBROUTINE MULTE3 IS THE FOURTH LEVEL OF THE BOX SCORING SYSTEM.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
     NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE4(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END

      SUBROUTINE MULTE4(ID)
C SUBROUTINE MULTE4 IS THE FIFTH LEVEL OF THE BOX SCORING SYSTEM.
      COMMON LL(300),SCOREH(40)RULEH(40),IDEX(40,10)
      DIMENSION INDEX(40)
      NUMAT=IDEX(ID,1)
      IF(NUMAT.EQ.0) RETURN
      IF(NUMAT.GT.50) GOTO 40
```

```
      DO 10 I=1,NUMAT
      I1=I+1
      L=IDEX(ID,I1)
      IF(SCOREH(L).GE.0) GOTO 10
      CALL MULTE5(L)
   10 CONTINUE
      J1=NUMAT+1
      DO 30 J=1,J1
   30 INDEX(J)=IDEX(ID,J)
      CALL TXTURE(ID,RULEH(ID),INDEX,SCOREH,SCORE)
      SCOREH(ID)=SCORE
      RETURN
   40 CALL BOX(ID)
      RETURN
      END


      SUBROUTINE GETR(ILOC,RESP)
C SUBROUTINE GETR RETRIEVES THE RESPONSES FROM A SPECIFIED QUESTIONNAIRE
C LOCATION.
      INTEGER RESP(40)
      J=ILOC+1
      READ(2,9000) RESP
 9000 FORMAT(/40(1X,1A1))
      RETURN
      END

      SUBROUTINE MULTE5(ID)
C SUBROUTINE MULTE5 SIMPLY PRINTS AN ERROR MESSAGE AND RETURNS.
      WRITE(6,900)
      WRITE(6,901)
  900 FORMAT(54H  YOU HAVE HIT THE LOWEST LEVEL IN THE SCORING ROUTINE)
  901 FORMAT(45H  PLEASE TRY SCORING LOWER LEVEL BOXES FIRST  )
      RETURN
      END
```

```
      SUBROUTINE SCOREQ(ID)
C SUBROUTINE SCOREQ SCORES QUESTIONNAIRES.
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      DIMENSION IRESP(40),RESP(40),RULE(10),IWRST(40),WEIGHT(40)
      DIMENSION SCORE(40),SCO(40),INDEX(40)
      INTEGER QDEX(10,40)
      DATA IBEST/1HA/
      BEST=FLOAT(IBEST)
      CALL GETQ(LOCQ(ID),RULE,IWRST,WEIGHT,NUMQ,NUMGP,QDEX)
      CALL GETR(LOCR(ID),IRESP)
      DO 10 I=1,NUMQ
      IF (IWRST(I).EQ.IBEST) GOTO 10
      RESP(I)=(BEST-FLOAT(IRESP(I)))/(BEST-FLOAT(IWRST(I)))
      RESP(I)=1.-WEIGHT(I)*RESP(I)
   10 CONTINUE
      DO 150 I=1,10
  150 SCORE(I)=-1.
      DO 140 I1=1,10
      FLAG=0.
      DO 100 I=1,NUMGP
      J1=QDEX(I,1)+1
      DO 110 J2=2,J1
      J3=QDEX(I,J2)
      IF(J3.GE.100) GOTO 115
      IF(SCORE(J3).LT.0) GOTO 120
  115 CONTINUE
  110 CONTINUE
      DO 130 J4=2,J1
      J5=QDEX(I,J4)
      IF(J5.LT.100) GOTO 160
      J5=J5-100
      SCO(J4)=RESP(J5)
      GOTO 170
  160 SCO(J4)=SCORE(J5)
  170 INDEX(J4)=J4
  130 CONTINUE
      FLAG=1.
      INDEX(1)=QDEX(I,1)
      CALL TXTURE(I,RULE(I),INDEX,SCO,SCORE(I))
      IF(I.EQ.1) GOTO 145
  120 CONTINUE
  100 CONTINUE
      IF(FLAG.EQ.0) GOTO 145
      IF(SCORE(1).GE.0) GOTO 145
  140 CONTINUE
  145 QSCORE(ID)=SCORE(1)
      WRITE(6,9000) QSCORE(ID)
 9000 FORMAT(13H  THE SCORE =,1F6.3)
      RETURN
      END

      SUBROUTINE GETQ(LOC,RULE,IWRST,WEIGHT,NUMQU,NUMGP,QDEX)
C SUBROUTINE GETQ READS THE STRUCTURE OF A QUESTIONNAIRE OFF DISC FILE 1
```

```
      INTEGER QDEX(10,40),ISAVE(40),IWRST(40)
      DIMENSION RULE(10),WEIGHT(40)
      J=LOC
      READ(1,9000) X,NUMQU,NUMGP
 9000 FORMAT(1A4,6X,1I2,8X,1I2)
      J=J+1
      READ(1,9002) IWRST
 9002 FORMAT(40(1X,1A1))
 9001 FORMAT(8F5.3)
      DO 50 I2=1,NUMGP
      J=J+1
      READ(1,9003) NAME,NUM,R
 9003 FORMAT(2(1I2,8X),1A2)
      NAME=NAME-49
      QDEX(NAME,1)=NUM
      RULE(NAME)=R
      J=J+1
      READ(1,9004) (ISAVE(J2),J2=1,NUM)
      DO 50 I=1,NUM
      I1=I+1
      IF(ISAVE(I).LT.50) QDEX(NAME,I1)=ISAVE(I)+100
      IF(ISAVE(I).GE.50) QDEX(NAME,I1)=ISAVE(I)-49
   50 CONTINUE
      J=J+1
      READ (1,9001)(WEIGHT(I2),I2=1,8)
      IF(WEIGHT(1).GT.1) GOTO 70
      IF(NUMQU.LE.8) RETURN
      J1=J+1
      READ(1,9001)(WEIGHT(I2),I2=9,NUMQU)
      RETURN
   70 DO 100 J2=1,NUMQU
  100 WEIGHT(J2)=WEIGHT(2)
 9004 FORMAT(40I2)
      RETURN
     END


      SUBROUTINE PRINTQ(ID,QNAME)
C SUBROUTINE PRINTQ PRINTS THE STRUCTURE FOR A QUESTIONNAIRE.
      COMMON LOCQ(100),QSCORE(100),LOCR(100)
      INTEGER QDEX(10,40)
      DIMENSION RULE(10),RESP(40),QNAME(100),IWRST(40),WEIGHT(40)
      DIMENSION IRESP(40),X(40),Y(40)
      DATA IBEST/1HA/
      CALL GETQ(LOCQ(ID),RULE,IWRST,WEIGHT,NUMQU,NUMGP,QDEX)
      BEST=FLOAT(IBEST)
      CALL GETR(LOCR(ID),IRESP)
      WRITE(6,8999)
      WRITE(6,9000) QNAME(ID)
 8999 FORMAT (//,20X,22H QUESTIONAIRE DATA FOR)
 9000 FORMAT(23X,13HQUESTIONAIRE ,1A4)
      WRITE(6,9002) QSCORE(ID),RULE(1)
 9002 FORMAT(8X,16HOVERALL SCORE = ,1F10.5,5X,7HRULE : ,1A4,//)
      DO 5 I=1,40
      IF (IWRST(I).NE.IBEST) GOTO 4
```

```
      X(I)=0.
      GOTO 6
    4 X(I)=1-(BEST-FLOAT(IRESP(I)))/(BEST-FLOAT(IWRST(I)))
    6 Y(I)=1.-WEIGHT(I)*(1-X(I))
    5 CONTINUE
      I=1
      J1=50
      WRITE(6,9004) J1,RULE(1)
 9004 FORMAT(7H  BOX: ,1I2,8H  RULE: ,1A2)
      I1=QDEX(I,1)+1
      IF(I1.EQ.1) GOTO 10
      DO 20 I2=2,I1
      WRITE(6,9005)
 9005 FORMAT(7H?      )
      I3=QDEX(I,I2)
      IF(I3.GE.100) GOTO 30
      J1=I3+49
      WRITE(6,9004) J1,RULE(I3)
      I4=QDEX(I3,1)+1
      IF(I4.EQ.1) GOTO 20
      DO 40 I5=2,I4
      WRITE(6,9005)
      WRITE(6,9005)
      I8=QDEX(I3,I5)
      IF(I8.GE.100) GOTO 50
      J1=I8+49
      WRITE(6,9004) J1,RULE(I8)
      I6=QDEX(I8,1)+1
      IF(I6.EQ.1) GOTO 45
      DO 60 I7=2,I6
      DO 11 J2=1,3
   11 WRITE(6,9005)
      J9=QDEX(I8,I7)
      IF(J9.GE.100) GOTO 70
      J1=J9+49
      WRITE(6,9004) J1,RULE(J9)
      GOTO 60
   70 J1=J9-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
 9006 FORMAT(4H  G=,1I2,7H  RESP=,1F6.3,3H W=,1F6.3,3H S=,1F6.3)
   60 CONTINUE
      GOTO 45
   50 J1=I8-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
   45 CONTINUE
   40 CONTINUE
      GOTO 20
   30 J1=I3-100
      WRITE(6,9006) J1,X(J1),WEIGHT(J1),Y(J1)
   20 CONTINUE
   10 RETURN
      END
```

This appendix contains additional information concerning the input and output for the example shown in Section 4.3. First, a table of component effectiveness test questionnaires is presented. Second, a sample questionnaire showing the questions and multiple choice responses is illustrated. Third, the disk files for the computer program are listed. FILE1 contains the questionnaire structures. FILE2 contains the questionnaire responses. FILE3 contains the hierarchy structure for the right side of the capability. FILE4 contains the results of the computer run for the right side. FILL3 contains the hierarchy structure for the left side of the capability. FIL4 contains the results of the computer run for the left side. The mnemonics for the left side correspond directly to the hierarchy boxes shown in Figure 1-1(1). Finally, additional output is presented for the left side of the capability hierarchy. (Note that the computer program simply references units 1,2,3 and 4. The user via computer system commands can make these units correspond to any file of his choosing).

COMPONENT EFFECTIVENESS TEST QUESTIONNAIRES

1. Admittance Authorization Criteria and Schedules
2. Admittance Authorization/Verification Procedures
3. Air and Utility Inlet Barriers
4. Annunciation Systems - Computer Assisted Annunciation
   Individual Alarm Annunciation
   Multiplex Alarm Annunciation
5. Area Zoning
6. Balanced Magnetic Switches
7. Breakwire Systems (Foil Strip and Grid Wire)
8. Buried Line Sensors - Seismic
   Magnetic
   Geophone String
   Piezo-electric String
9. Capacitance Alarms
10. CCTV Monitoring/Surveillance
11. CCTV Systems
12. Central and Secondary Alarm Stations
13. Close out Inspection by Third Party
14. Coded Credential System - Active Electronic Badge Reader
    Capacitance Coded Badge Reader
    Electric Circuit Badge Reader
    Magnetic Coded Badge Reader
    Magnetic Stripe Badge Reader
    Magnetic Strip Badge Reader
    Optical Coded Badge Reader
    Passive Electric Badge Reader
15. Commercial Telephone System
16. Contingency Plans and Procedures
17. Controlled Security Lighting
18. Data Link Via Radio Frequency
19. Direct-Line Telephone/Intercom
20. Direct Monitoring/Surveillance
21. Doors and Associated Hardware
22. Duress Alarms
23. E-Field Fence
24. Electret Cable and Tilt Switch Fence Systems
25. Emergency Access/Egress Procedures
26. Emergency Battery System
27. Emergency Evacuation Procedures
28. Emergency Exits
29. Emergency Generator Systems
30. Equipment Checks/Maintenance

31. Escorts
32. Explosive Detector - Hand Held Package Search
33. Explosive Detector - Hand Held Personnel Search
34. Explosive Detector - Hand Held Vehicle Search
35. Explosive Detector - Volume
36. Explosive Detector - Walk Through
37. Fence Systems
38. Floors
39. Functional Zoning
40. Gates and Associated Hardware
41. Guard Force Personal Equipment
42. Guard Force Qualification
43. Guard Patrols/Intervention
44. Guard Post Assignments
45. Hardware Video Systems
46. Infrared Beam Systems, Exterior
47. Interface Between Alarm Station and Sensors
    - Individual Hardwire Alarms
    - Multiple Hardwire Alarms
    - Hardwire Command Signals
48. Isolation Zones
49. K-9s, Use of - Package Search
50. K-9s, Use of - Vehicle Search
51. Local Audible/Visible Alarms
52. Locks - (Key Locks, Keyless Locks)
53. Manual Alarm Recording
54. Master Fixed Radio
55. Microwave Systems, Exterior
56. Mobile Radio
57. Motion Detectors - Infrared Systems, Interior
    Microwave Systems, Interior
    Ultrasonic and Sonic Systems
58. Multi-Man Rule
59. Night Vision Devices
60. Package Search - Visual Inspection
61. Pat-Down Search
62. Personal Identification Numbers/Passwords
63. Photo Identification Badges
64. Physical Controls and Procedures for Keys, Locks, Combinations, and Cipher Systems
65. Portable Radio
66. Positive Personnel Identification
    - Fingerprint Personnel I.D. Verification
    - Handwriting Personnel I.D. Verification
    - Hand Geometry Personnel I.D. Verification
    - Voice Print Personnel I.E. Verification
67. Response Vehicles
68. Roof
69. Sally Ports, Pedestrian

70. Sally Ports, Vehicular
71. Shielding Detector - Volume
72. Shielding Detector - Walk Through
73. SNM Containers
74. SNM Detector - Hand Held Package Search
75. SNM Detector - Hand Held Personnel Search
76. SNM Detector - Volume
77. SNM Detector - Walk Through
78. SNM Holding/Storage Area
79. SNM Identification/Authorization Procedures
80. SNM Liquid and Solid Waste Handling Procedures
81. SNM Scrap Removal Procedures
82. SNM Shipping/Receiving Procedures
83. Tamper Indicating Circuitry
84. Tamper Indicating Seals and Tamper Seal Inspection
85. Team Zoning
86. Uninterruptible Power System
87. Vaults
88. Vehicle Search - Visual Inspection
89. Vibration Sensors
90. Walls
91. Weapons - Handgun
    Semi-Automatic
    Shotgun
92. Weapons Detector - Hand Held Package Search
93. Weapons Detector - Hand Held Personnel Search
94. Weapons Detector - Volume
95. Weapons Detector - Walk Through
96. Windows and Associated Hardware
97. X-Ray Package/Container Search

EFFECTIVENESS TEST

FUNCTION

The function of the annunciation system will be to alert security personnel to alarm activation.

CONDITIONS

Performance Conditions

Installation

1. Where will peripheral equipment such as computers and communications electronics be located?

Operation

2. How much console space will be occupied by primary controls and displays that require observation or action several times per shift?
3. Where will the primary control and display area be situated with respect to the operator?
4. Where will all primary controls be located with respect to their accessability to the operator?
5. How will the operator's attention be directed to the annunciators?
6. Will security annunciators be monitored by the same operator who monitors other annunciators?
7. Will the status of sensors (secure/access/alarm/tamper) within a security zone be available to the operator?
8. How will the importance or priority of an alarm be determined?
9. When an alarm occurs, to what extent will the sensor's location be available to the operator?
10. What additional information will be available to the operator if an alarm occurs?
11. To what extent will the annunciation system indicate multiple concurrent alarms?
12. How will significant events be recorded?

Reliability

13. How frequently will the system be checked for proper operation?

14. What provisions will be made to maintain operational capabilities when critical elements, i.e., CPU, CRT, audio and visual devices, etc., fail?

15. If the system is equipped with self-test capability, what will be the test frequency?

Vulnerabilities

16. What techniques will be used to deter unauthorized modification of programs or data?

ANSWERS

CONDITIONS

Performance Conditions

Installation

1. a. In a separate access-controlled room.
   b. In the same room but away from primary display and control area.
   c. In the same console area as the primary displays and controls.

Operation

2. a. Less than 250 square inches.
   b. 250 to 700 square inches.
   c. 700 to 1700 square inches.
   d. More than 1700 square inches.

3. a. Approximately perpendicular to a seated operator's line of sight.
   b. In a vertical plane.
   c. In a horizontal plane.

4. a. Completely within convenient reach of the operator.
   b. Partially within the operator's reach.
   c. Not within reach from the operator's normal location and will require the operator to move from his location.

5. a. By an audible signal which varies depending on type of alarm plus visual indicators.
   b. By an unchanging audible signal plus visual indicators.
   c. By visual indicators only.

6. a. No.
   b. Yes.

7. a. The status of each sensor will be available.
   b. The most significant status within a group of sensors will be available.
   c. The most significant status within the security zone will be indicated.
   d. Only the occurrence of an alarm will be indicated.

8. a. Automatically, by a hardware or software priority structure.
   b. By the operator in a predetermined priority structure.
   c. By the operator using real-time judgment.

9. a. The location of the specific sensor in alarm will be available.

b. The location of the sensor group containing the specific sensor in alarm will be available.
c. The location of the general area containing the specific sensor in alarm will be available.

10. a. 1. The time of alarm,
2. The priority of alarm,
3. Emergency telephone numbers,
4. Special precautionary instructions associated with a zone, and
5. Area maps.
b. 1., 2., 3., and 4. above.
c. 1., 2., and 3. above.
d. 1. and 2. above.
e. Only 2. above.

11. a. It will advise the operator of multiple concurrent alarms.
b. It will permit only a sequential display of multiple concurrent alarms.
c. It will display only one of multiple concurrent alarms.

12. a. They will be automatically printed out.
b. They will be recorded automatically and manually in combination.
c. They will be manually recorded.
d. They will not be recorded.

Reliability

13. a. Every few seconds.
b. Every few minutes.
c. Every few hours.
d. Once per shift.
e. Once per day.
f. Once per week.
g. Less than once per week.

14. a. A fully redundant system of annunciation is to be provided.
b. Full redundancy is to be provided for all critical subsystems and computers.
c. Significant increase of patrols will be provided.

15. a. At 10- to 30-second intervals.
b. At 30- to 60-second intervals.
c. At 1- to 5-minute intervals.
d. The system will not have self-test capability.

16. a. By encryption.
b. By multiple passwords.
c. By single password.
d. By administrative controls.
e. None.

```
38
   4   9  14  19  24  29  34  39  44  49  54  59  64  69  74  79  84  89  94  99
 104 109 114 119 124 129 134 139 144 149 154 159 164 169 174 179 184 189 194 199
4          16          01
 C D C C C B C C B C B C G C D E
50         16          SA
 1 2 3 4 5 6 7 8 910111213141516
2.    .5
6          15          01
 C C C B B B B D C B D D B D D
50         15          SA
 1 2 3 4 5 6 7 8 9101112131415
2.    .5
10         16          01
 D C C D B B E C C B E C E D B C
50         16          SA
 1 2 3 4 5 6 7 8 910111213141516
2.    .5
47         10          01
 D B C F E C D C E D
50         10          SA
 1 2 3 4 5 6 7 8 910
2.    .5
57         20          01
 D C E C B D C B C B B D D D C D C C D D
50         20          SA
 1 2 3 4 5 6 7 8 91011121314151617181920
2.    .5
1          05          01
 C C C C C
50         05          SA
 1 2 3 4 5
2.     .5
2          17          01
 D C C C B C D D C C C C C C C B C
50         17          SA
 1 2 3 4 5 6 7 8 91011121314151617
2.     .5
3          05          01
 C B D A D
50         04          SA
 1 2 3 5
2.     .5
11         16          01
 C C C E D D B D B D B D D D C D
50         16          SA
 1 2 3 4 5 6 7 8 910111213141516
2.     .5
14         10          01
 B B C B B C B D C D
50         10          SA
 1 2 3 4 5 6 7 8 910
2.     .5
16         04          01
 C C B C
```

```
50          04          SA
 1 2 3 4
2.     .5
21          05          01
 A B B C C
50          04          SA
 2 3 4 5
2.     .5
22          11          01
 C B C B C B C B D B C
50          11          SA
 1 2 3 4 5 6 7 8 91011
2.     .5
25          19          01
 B D C C B B C C C B B B B B C B C B B
50          19          SA
 1 2 3 4 5 6 7 8 910111213141516171819
2.     .5
28          13          01
 B C B B B B A B D D B D D
50          12          SA
 1 2 3 4 5 6 8 910111213
2.     .5
32          14          01
 B D B C B B B B B B B C C B
50          14          SA
 1 2 3 4 5 6 7 8 91011121314
2.     .5
36          18          01
 C C C B B B B C B B B B B C C B B D
50          18          SA
 1 2 3 4 5 6 7 8 9101112131415161718
2.     .5
36          04          01
 A B C B
50          03          SA
 2 3 4
2.     .5
43          15          01
 B B B B C C B B B C C B B B B
50          15          SA
 1 2 3 4 5 6 7 8 9101112131415
2.     .5
51          13          01
 C C C C B C C D D D C D C
50          13          SA
 1 2 3 4 5 6 7 8 910111213
2.      .5
60          14          01
 D C C D B B B C C C C B D C
50          14          SA
 1 2 3 4 5 6 7 8 910111213141
2.     .5
63          16          01
 C C C D B C C B C     B D B B C C C
```

```
50        18        SA
 1 2 3 4 5 6 7 8 9101112131415161718
2.    .5
66        14        01
 B B C B B C B C B B D C C D
50        14        SA
 1 2 3 4 5 6 7 8 91011121314
2.    .5
68        03        01
 A C E
50        02        SA
 2 3
.5    .5
69        06        01
 B B A C C E
50        06        SA
 1 2 3 4 5 6
2.    .5
74        14        01
 B D B B E B B B C C E B B B
50        14        SA
 1 2 3 4 5 6 7 8 91011121314
2.    .5
75        17        01
 D B C B B B C C C B C B B B B C B
50        17        SA
 1 2 3 4 5 6 7 8 91011121314151617
2.    .5
83        07        01
 C D D F B B E
50        07        SA
 1 2 3 4 5 6 7
2.     .5
84        09        01
 B B C C B B E D D
50        09        SA
 1 2 3 4 5 6 7 8 9
2.    .5
87        05        01
 C A B D B
50        04        SA
 1 3 4 5 6
2.    .5
90        03        01
 A B D
50        02        SA
 2 3
.5    .5
95        29        01
 C B C C C C C B C C C B B B B C B C C B B D C C B B B B B
50        29        SA
 1 2 3 4 5 6 7 8 91011121314151617181920212223242526272829
2.    .5
12        33        01
 C D C C B C C B B C B B B C B D E C C D D B B C B B C D B D C B D
```

```
50          33          SA
 1 2 3 4 5 6 7 8 9101112131415161718192021222324252627282930313233
2.     .5
33          13          01
 D B C B B C B B C B C C B
50          13          SA
 1 2 3 4 5 6 7 8 910111213
2.     .5
ALAS        07          01
 D B D C E C D
50          07          SA
 1 2 3 4 5 6 7
2.     .5
PNSS        06          01
 B B C B E B
50          06          SA
 1 2 3 4 5 6
2.     .5
17          11          01
 D D C B D D D D D B D
5L          11          SA
 1 2 3 4 5 6 7 8 91011
2.     .5
16          15          01
 B D E C B C F B D B C D D E D
50          15          SA
 1 2 3 4 5 6 7 8 9101112131415
2.     .5
```

```
38
   4   6   8  10  12  14  16  18  20  22  24  26  28  30  32  34  36  38  40  42
  44  46  48  50  52  54  56  58  60  62  64  66  68  70  72  74  76  78  80  82
4
 A B A A A A A A A A A A B A B B
6
 A B A A A A A A B A A A A A A
10
 A B B A A A A B A A A A A A A A
47
 A A A A B A A B A A
57
 A A A A C A A A A A A C C A A A A A A
1
 A A A A A
2
 A A A A A A B A A A A A A A A A A
3
 C A B A B
11
 A B A B A A A A A A A A A A A A
14
 A A A A A A A A A A
16
 A A A A
21
 A A A A B
22
 A A A B A A C B D B A
25
 A A A A A A A A A A A A B A A A C A A
28
 A A A A A A A B A B B A A
32
 A A A A A A B A A A A A A A
36
 A B A A A A A A A A A A A A A A A A
38
 A A A A
43
 A A A A A A A A A A A A A A A
51
 A A A B A A B A A A A A A
60
 D A A A A A A A C A A A B A
63
 C C A B A A A A A A A A A A A A C A A
66
 A A A A A A A A A A A A A A
68
 A A A
69
 A B A C A D
74
 A A A A A A A A A A A A A A
```

```
75
 A A A A A A A A A A A A A A A A A
83
 A B B B A A B
84
 A A A B A A C A C
87
 A A A A B
90
 A B A
95
 A A A B B B A A A B A A A A A A A B A A A A B A B A A A A
12
 A B A A A A A A A A A B B B A A A A A A A A A A B A A A A A A A A
33
 A A A A A A A A A A A A A
ALAS
 D A B A A B A
PNSS
 A A A A A A
17
 A A A A A A A A A A A
18
 A A A A A A A A A A A A A A A
```

```
01
C003
DENACC     2      SA
DETACC
RESACC
DETACC     3      HA
SENSE
REPALR
ASSESS
RESACC     2      HA
COMRSP
RESP
SENSE      2      SC
MULTS
INCM1
REPALR     2      HA
TSIC
ANALRM
ASSESS     3      AV
MULTA
INDA1
CASSAS
COMRSP     4      SA
BETGDS
GDSSTN
BETSTN
ONOFF
RESP       3      SA
DELRSP
EFFRSP
DRRSP
MULTS      53     AV
6
57
10
INCM1      3      SA
DDS
GP
BARR
TSIG       52     SC
47
18
ANALRM     53     SC
4
51
43
MULTA      51     AV
10
INDA1      3      SA
DDA
GP
BARR
CASSAS     51     AV
12
GDSSTN     51     HA
```

```
43
DELRSP    2     SO
BARR
GP
EFFRSP    2     SA
ONSITE
OFFSIT
BARR     57     SA
3
21
28
38
68
69
90
GP       51     AV
43
ONSITE    2     SA
REQOFF
CONADV
OFFSIT    2     SA
RSPREG
ENGADV
REQOFF   52     HA
12
16
CONADV   52     SA
16
43
RSPREG   51     HA
16
ENGADV   51     SA
16
NOMORE
```

```
01
0003
DENACC 0.436DETACC 0.335RESACC 0.730SENSE  0.702REPALR 0.868
ASSESS 0.549COMRSP 1.000RESP   0.730MULTS  0.671INDM1  0.575
TSIG   0.940ANALRM 0.523MULTA  0.670INDA1  0.640CASSAS 0.338
BETGCS 1.000GCSSTN 1.000BETSTN 1.0000NOFF  1.000DELRSP 0.790
EFFRSP 0.706DRRSP  1.000BARR   0.370GP     1.000ONSITE 0.559
OFFSIT 1.000REQOFF 0.338CONADV 1.000RSPREQ 1.000ENGADV 1.000
DDS    0.833DDA    1.000
 0.75493 0.76562 0.66992 0.82014 0.57870 1.00000 0.91667 0.60556 0.82031 1.00000
 1.00000 0.91111 0.23693 0.56250 0.51551 0.74998 0.87500 1.00000 1.00000 0.76560
 0.51559 0.38672 1.00000 1.00000 0.54839 1.00000 1.00000 0.74639 0.63731 0.73333
 0.66667 0.33660 0.33838 1.00000 0.59842 1.00000 1.00000 1.00000 0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
```

```
01
0003
CONACC     2      SA
NORMAL
EMERGE
NORMAL     2      MA
ADAUTH
PROCON
ADAUTH    51      AV
2
PROCON     2      SA
PERSON
MATERI
PERSON     3      SA
VERIF
CONTRA
RESPVI
MATERI     3      SA
VERIF2
CONTR2
RESPVI
VERIF     54      AV
14
63
2
66
CONTRA    51      AV
95
RESPVI     2      MA
COMRSP
RESP
VERIF2    51      AV
2
CONTR2    52      SA
32
60
RESP       3      SA
DELRSP
EFFRSP
ORRSP
NOMORE
```

```
FIL4      DATA     P ID=WCCWR      16.21.34  THURSDAY 6 DECEMBER 1979       PAGE 1
   NATIONAL CSS, INC. (SUNNYVALE DATA CENTER)          SUNY

01
0003
CONACC 0.547NORMAL 0.442EMERGE 0.832ADAUTH 0.917PROCON 0.482
PERSON 0.496MATERI 0.635VERIF  0.826CONTRA 0.337RESPVI 0.832
VERIF2 0.917CONTR2 0.551COMRSP 1.000RESP   0.832DELRSP 1.000
EFFRSP 0.706DRRSP  1.000
 0.75493 0.76562 0.66992 0.82014 0.57870 1.00000 0.91667 0.60556 0.82031 1.00000
 1.00000 0.91111 0.23693 0.56250 0.51551 0.74998 0.87500 1.00000 1.00000 0.76563
 0.51559 0.38672 1.00000 1.00000 0.54839 1.00000 1.00000 0.74639 0.63731 0.73333
 0.66667 0.33660 0.33836 1.00000 0.59842 1.00000 1.00000 1.00000 0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
 0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0     0.0
```

```
WHICH? 42
ENTER BOX NAME -- CONACC

HIERARCHY DATA FOR BOX CONACC

BOX:CONACC  RULE:SA  SCORE: 0.547   Q:
      BOX:NORMAL  RULE:HA  SCORE: 0.442   Q:
            BOX:ADAUTH  RULE:AV  SCORE: 0.917   Q:
                   QUESTIONNAIRE: 2      SCORE: 0.917
            BOX:PROCON  RULE:SA  SCORE: 0.482   Q:
                  BOX:PERSON  RULE:SA  SCORE: 0.496   Q:
                  BOX:MATERI  RULE:SA  SCORE: 0.635   Q:
      BOX:EMERGE  RULE:     SCORE: 0.832   Q:
SELECT 41-51 --       42
ENTER BOX NAME -- PROCON

HIERARCHY DATA FOR BOX PROCON

BOX:PROCON  RULE:SA  SCORE: 0.482   Q:
      BOX:PERSON  RULE:SA  SCORE: 0.496   Q:
            BOX:VERIF   RULE:AV  SCORE: 0.826   Q:
                   QUESTIONNAIRE: 14     SCORE: 1.000
                   QUESTIONNAIRE: 63     SCORE: 0.337
                   QUESTIONNAIRE: 2      SCORE: 0.917
                   QUESTIONNAIRE: 66     SCORE: 1.000
            BOX:CONTRA  RULE:AV  SCORE: 0.337   Q:
                   QUESTIONNAIRE: 95     SCORE: 0.337
            BOX:RESPVI  RULE:HA  SCORE: 0.832   Q:
                  BOX:COMRSP  RULE:     SCORE: 1.000   Q:
                  BOX:RESP    RULE:SA  SCORE: 0.832   Q:
      BOX:MATERI  RULE:SA  SCORE: 0.635   Q:
            BOX:VERIF2  RULE:AV  SCORE: 0.917   Q:
                   QUESTIONNAIRE: 2      SCORE: 0.917
            BOX:CONTR2  RULE:SA  SCORE: 0.551   Q:
                   QUESTIONNAIRE: 32     SCORE: 0.750
                   QUESTIONNAIRE: 60     SCORE: 0.516
            BOX:RESPVI  RULE:HA  SCORE: 0.832   Q:
                  BOX:COMRSP  RULE:     SCORE: 1.000   Q:
                  BOX:RESP    RULE:SA  SCORE: 0.832   Q:
SELECT 41-51 -- 42
ENTER BOX NAME -- RESPVI

HIERARCHY DATA FOR BOX RESPVI

BOX:RESPVI  RULE:HA  SCORE: 0.832   Q:
      BOX:COMRSP  RULE:     SCORE: 1.000   Q:
      BOX:RESP    RULE:SA  SCORE: 0.832   Q:
            BOX:DELRSP  RULE:     SCORE: 1.000   Q:
            BOX:EFFRSP  RULE:     SCORE: 0.706   Q:
            BOX:DRRSP   RULE:     SCORE: 1.000   Q:
```

```
SELECT 41-51 -- 50
ENTER BOX NAME -- CONACC
HIERARCHY INFORMATION FOR BOX CONACC

***********       ***********       ***********
* CONACC  *       * NORMAL  *       * ADAUTH  *
*S= 0.547*********S= 0.442*********S= 0.917*
*RULE: SA*    *   *RULE: HA*    *   *RULE: AV*
***********   *   ***********   *   ***********
              *                 *
              *                 *   ***********       ***********       ***********
              *                 *   * PROCON  *       * PERCON  *       * VERIF   *
              *                 ****S= 0.462**********S= 0.496******S= 0.826****
              *                     *RULE: SA*    *   *RULE: SA*    *   *RULE: AV*
              *                     ***********   *   ***********   *   ***********
              *                                   *                 *
              *                                   *                 *   ***********
              *                                   *                 *   * CONTRA  *
              *                                   *                 ****S= 0.337****
              *                                   *                 *   *RULE: AV*
              *                                   *                 *   ***********
              *                                   *                 *
              *                                   *                 *   ***********
              *                                   *                 *   * RESPVI  *
              *                                   *                 ****S= 0.832*
              *                                   *                     *RULE: HA*
              *                                   *                     ***********
              *                                   *
              *                                   *   ***********       ***********
              *                                   *   * MATERI  *       * VERIF2  *
              *                                   ****S= 0.635******S= 0.917*
              *                                       *RULE: SA*    *   *RULE: AV*
              *                                       ***********   *   ***********
              *                                                     *
              *                                                     *   ***********
              *                                                     *   * CONTR2  *
              *                                                     ****S= 0.551*
              *                                                     *   *RULE: SA*
              *                                                     *   ***********
              *                                                     *
              *                                                     *   ***********
              *                                                     *   * RESPVI  *
              *                                                     ****S= 0.832*
              *                                                         *RULE: HA*
              *                                                         ***********
              *
              *   ***********
              *   * EMERGE  *
              ****S= 0.832*
                  *RULE:    *
                  ***********

SELECT 41-51 -- 3
```

DISTRIBUTION:

U.S. Nuclear Regulatory Commission (50 copies for AN)
Division of Document Control
Distribution Services Branch
7920 Norfolk Avenue
Bethesda, MD 20014
Attn: P. Larkins

U.S. Nuclear Regulatory Commission
Division of Safeguards
MSSS881
7915 Eastern Avenue
Silver Spring, MD 20852
Attn: P. A. Dwyer (5)

Woodward-Clyde Consultants
Three Embarcadero Center
Suite 700
San Francisco, CA 94111
Attn: A. Sicherman (5)

400	C. Winter
1000	G. A. Fowler
1230	W. L. Stevens, Attn: R. E. Smith, 1233
1700	W. C. Myre
1710	V. E. Blake
1720	C. H. Mauney
1730	J. D. Kennedy
1750	J. E. Stiegler
1760	J. Jacobs
4400	A. W. Snyder
4410	D. J. McCloskey
4412	J. W. Hickman
4413	N. R. Ortiz
4414	C. E. Varnado
4416	L. D. Chapman (5)
4416	K. G. Adams
4416	J. A. Allensworth
4416	H. A. Bennett (7)
4416	D. Engi
4416	L. M. Grady
4416	C. P. Harlan
4416	R. D. Jones
4416	M. T. Olascoaga (7)
4416	C. J. Pavlakos
4416	D. W. Sasser
4416	D. R. Strip
8266	E. A. Aas
3141	T. L. Werner (5)
3151	W. L. Garner (3)
	For: DOE/TIC (Unlimited Release)
3154-3	R. P. Campbell (25)
	For NRC Distribution to NTIS

☆U.S. Government Printing Office: 1980 — 677-116/530