DOCUMENTATION OF DECISION-AIDING SOFTWARE:
RAM SYSTEM SPECIFICATION

by
Dorothy M. Amey, Phillip H. Feuerwerger, and Roy M. Gulick

Sponsored by
Defense Advanced Research Projects Agency
ARPA Order 3469

September 1979

DECISIONS and DESIGNS, INC.
Suite 600, 8400 Westpark Drive
P.O.Box 907
McLean, Virginia 22101
(703) 521-2628

DISTRIBUTION STATEMENT A
Approved for public release;
Distribution Unlimited
## CONTENTS

### FIGURES

<table>
<thead>
<tr>
<th>FIGURES</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>iii</td>
<td></td>
</tr>
</tbody>
</table>

### 1.0 INTRODUCTION

1.1 Purpose of the System Specification 1

1.2 References 1

1.3 Terms 2

1.3.1 RAM 2

1.3.2 HIPO 2

### 2.0 DESIGN DETAILS

2.1 Background 3

2.2 General Operating Procedures 3

2.3 System Logical Flow 3

2.4 HIPO Documentation 6

---

Accession For

NTIS CRAAI
DOD TAB
Unannounced

Justification

Distribution/ Availability Codes

Avail and/or Special

Dist A
## FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-1</td>
<td>LEGEND OF HIPO SYMBOLS</td>
<td>5</td>
</tr>
<tr>
<td>2-2</td>
<td>RAM SYSTEM OVERVIEW</td>
<td>6</td>
</tr>
<tr>
<td>2-3</td>
<td>BUILDRAM SUBSYSTEM OVERVIEW AND VISUAL TABLE OF CONTENTS</td>
<td>7</td>
</tr>
<tr>
<td>2-4</td>
<td>REPRAM SUBSYSTEM OVERVIEW AND VISUAL TABLE OF CONTENTS</td>
<td>8</td>
</tr>
</tbody>
</table>
RAM SYSTEM SPECIFICATION

1.0 INTRODUCTION

1.1 Purpose of the System Specification

The RAM System Specification is a technical document written for software development personnel. Together with the RAM Functional Description, it guides the software development effort by identifying the functional requirements and by providing structured logic diagrams that depict the flow, control, and processing of information within the system.

The system specification is generic and is intended to guide and facilitate the preparation of the language-specific program documentation and coding that are necessary to implement and operate RAM at an installation.

1.2 References


1.3 Terms

1.3.1 RAM - RAM is an abbreviation for Resource Allocation Model, reflecting the system's major area of applicability.

1.3.2 HIPO - The specification uses the standard Hierarchy plus Input-Process-Output (HIPO) diagramming technique to depict the structural design and logical flow of the system. A legend explaining the HIPO diagramming symbols is included. Reference 1.2.1 provides a complete description of the HIPO documentation technique.
2.0 DESIGN DETAILS

2.1 Background

Systems development personnel should refer to the RAM Functional Description, reference 1.2.2, in conjunction with the documentation contained in this specification. The functional description details the resource allocation model implemented by RAM and discusses the specific functions that the software performs. In addition, systems development personnel may wish to refer to the RAM User's Manual, reference 1.2.3.

2.2 General Operating Procedures

RAM is a menu-driven system. That is, the system is designed to interact with the user by presenting a sequential hierarchy of menus and asking the user to respond by selecting one option from the current menu. If the user does not select one of the menu options, the system displays the previous menu. In this manner, the user moves up and down the hierarchy, as desired. Whenever data entry is required as a result of option selection, the system specifically requests the data and specifies the format.

The system is also designed to be generally forgiving of procedural errors by the user.

2.3 System Logical Flow

RAM is a hierarchically structured, modular system. The system structure and logical flow lends itself to presentation in the form of HIPO diagrams, which are contained in this document.
The main purpose of the HIPO diagrams is to provide, in a pictorial manner, the complete set of modular elements necessary to the operation of RAM including all input, output, and internal functional processing. This is done by displaying input items to the process step which uses them, defining the process, and showing the resulting output of the process step.

The documentation diagrams are designed and drawn in a hierarchical fashion from the main calling routines to the detail-level operation/calculation routines. Extended written descriptions are given below a HIPO diagram whenever it is deemed necessary.

A complete explanation of the symbolic notation used in the HIPO diagrams is given in reference 1.2.1. An abbreviated legend for the symbols used in this specification is given in Figure 2-1. Note that:

a. External subroutines appear partly in the process block and partly out. Internal subroutines are shown within the process block.

b. Overview diagrams show general inputs and outputs only, whereas detail/subroutine-level diagrams show specific input/output tables and/or displays.

c. Rectangular boxes inside the input/output block areas are generally used to denote single data items. Two or more boxes are grouped to show several data items are input/output.

d. Rectangular boxes inside the process block indicate repetitive subprocesses.
Control

Data movement

Pointer

Data reference

Keyed data arrows

Off-page connection arrows

General flow of data among subprocesses

Subroutine invoked (Return is made to calling routine)

Routine receives control

Routine exits or returns control

Information display by online indicators – prompted by program execution or by keyboard input, especially CRT

Figure 2-1

LEGEND OF HIPO SYMBOLS
The HIPO diagrams appear in the next section, which completes the system specification.

2.4 HIPO Documentation

The HIPO diagram identification numbers and figure numbers used in this section stand alone; i.e., they start with 1.0, increase hierarchically, and are independent of the numbering scheme used to this point in this document.

The RAM system comprises two subsystems: BUILDRAM, which builds and exercises the resource allocation model, and REPRAM, which produces various reports based on the model and its data. Figure 2-2 is the system structure chart. Figures 2-3 and 2-4 are the subsystem charts and represent the overall program logic flows in visual tables of contents. The Visual Table of Contents shows the hierarchical structure, the functional description labels, and the diagram (chart) identifiers of the functions implemented by the RAM subsystems.
Figure 2-2
RAM SYSTEM OVERVIEW
Figure 2-3
BUILDGRAM SUBSYSTEM OVERVIEW AND VISUAL TABLE OF CONTENTS
Figure 2-4

REPRAM SUBSYSTEM OVERVIEW AND VISUAL TABLE OF CONTENTS
The BUILDRAM subsystem allows the creation of a resource allocation model (a cost/benefit analysis model), the calculation of resulting cost/benefit (C/B) ratios, the rank ordering of these C/B ratios and the display and storage/retrieval of the calculated results.

Input/output devices are made ready for use by the RAM program in step 2 and they are freed by processing in step 12.

Functional processing begins by having the user input a selection from an optional procedures list (MENU). Once the selected processing is completed, the same menu is displayed again and the user may select another process (or the same one again).

This means that procedures 4–11 are repetitive in structure. These functional subprocesses are described in detail in diagrams 4.0 through 11.0.
8. Develop cross-sponsor weights.
9. Edit the model.
10. Calculate results.
11. Compare benefits w/cost/benefit.
12. Stop; finalize input/output.

Exit
1. Display a message to ready the desired RAM data file on the selected storage device.
2. Wait for a user response indicating device/file is online and ready.
4. Read in model definition variables, data matrices and results.
5. Return.

Extended Description
1. The device address of online storage which contains the RAM data and model definitions is available at the start of program processing. It is either encoded in the program or requested from the user. It is assumed that only one RAM data set is available per dev/file address (i.e., diskette, drum cylinder, etc.)
3. A system device OPEN command is required in order to access the RAM model and data. The interface with the system OPEN routine should be OPEN's setting of I/O flags in (or adjacent to) an I/O buffer area to denote the success or failure of access to the device. If an error condition is detected, a non-zero error code is returned to the caller.
4. Model variables are read according to a preset order which is identical to the manner in which the variables are stored (see diagram 4.2).
1. Issue a CLOSE command for the storage device (file) containing the RAM model data and definitions.

2. Test for ERROR conditions.
   a. If no error, set the return code to zero.
   b. If an error is detected, set return code to a non-zero error code and display an error message.

3. Delete I/O Buffer and RETURN.
1. Display the two process options: load or save.

2. Determine the desired process and do one of steps 3 – 5. Repeat 1 after steps 3 or 4.

3. Load a RAM model.

4. Save the current model.

5. Return to calling routine.

Extended Description

1. Display the options to Load or Save a model in MENU format so that the user may select a process numerically.

2. If the user inputs a blank response, then do step 5.
1. Display the name of the model which is online and determine if it is the desired model.

2. If the desired model is not online, then initiate I/O for the correct model. Do step 4.

3. Read in the model definition variables, data matrices and results.

4. Return to calling routine.

Extended Description

1. The user has the option of switching the online device/file at this time, since only one RAM model is stored per device/file.

2. If the desired device is not already online, then the appropriate one must be placed online and opened for input (see diagram 2.0). An input buffer is created for the newly selected model.

3. Read data commands are issued for the required model definition and data variables. This is accomplished according to an encoded variable list and in the same order as the data items are stored.
Extended Description

1. While the user is operating the RAM program, a file that contains a RAM model should always be online and opened for I/O by the INITIATE routine (see diagram 2.6). The model name is among the stored variables which were read in at "LOAD A MODEL" time (see diagram 4.1).

2. The user may indicate after prompting that he does not want to save a model on the currently available file: the routine allows him to return to the main calling routine. If the user wants to store on the available file, he enters a new model name for the file.

3. Write commands are issued causing the model definition variables to be stored in an exact order. The following variables are stored:

   1. model name
   2. sponsor labels and item labels
   3. encoded sponsor-item numbers (the identifiers)
   4. a vector of the associated sponsor number per item
   5. cross-sponsor weights
   6. RAM matrix of benefits and computed results
   7. sum of overall benefits per sponsor
   8. COST matrix of cost values per item
   9. cost-component labels
   10. the rank order of cost/benefit values
1. The user inputs the name of the sponsor group to be displayed or inputs a blank line to terminate this process.

2. Cost-benefit values from the RAM matrix as well as the cost-benefit ratio and rank order of the CB ratio are displayed beside each item name which was created under the particular sponsor group. The sponsor group name and sum of the overall benefit values for the sponsor are displayed as headings (see figure 5.1).

The RAM matrix contains a set of cost-benefit values for each sponsor item. These values are the following (see figure 5.2):

1) The input benefit value
2) The overall benefit value. This value was computed using cross-sponsor weights (see diagram 10.0)
3) The total cost value computed from the five component cost values
4) The cost-benefit ratio
<table>
<thead>
<tr>
<th>ITEM</th>
<th>SPONSOR</th>
<th>BENEFIT</th>
<th>OVERALL</th>
<th>BENEFIT</th>
<th>COST</th>
<th>C/B</th>
<th>RANK</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 2) ALUMNI SURVEY</td>
<td>25.0</td>
<td>2.5</td>
<td>5.0</td>
<td>2.0</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1 3) ARCHITECTURAL SURVEY</td>
<td>100.0</td>
<td>10.0</td>
<td>80.0</td>
<td>8.0</td>
<td>17</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 4) EXPANSION PLANNING</td>
<td>65.0</td>
<td>6.5</td>
<td>113.0</td>
<td>17.4</td>
<td>21</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 1) SECRETARIAL SERVICES</td>
<td>10.0</td>
<td>1.0</td>
<td>95.0</td>
<td>95.0</td>
<td>24</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PLEASE RETURN CARRIAGE TO CONTINUE**

**HIPO Figure 5.1**

**DISPLAY OF SPONSOR RESULTS**
RAM AND COST MATRICES

**RAM matrix** - Number of Items = \( k \)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( k )</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Columns No. 1: Benefit value specified by the user  
No. 2: Over-all benefit value  
No. 3: Total cost across cost components  
No. 4: Cost-benefit ratio

**COST matrix** - Number of items = \( k \)  
- Number cost components = \( r \)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>\cdots</th>
<th>( r )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( k )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Columns No. 1 through \( r \) = costs for each of the \( r \) components (usually \( r = 5 \))  
No. \( r + 1 \) : "other" costs single value

HIPO Figure 5.2  
STRUCTURE OF THE RAM AND COST MATRICES
To create a model structure, certain variable lists and data matrices must be formed. These variables may be stored after creation in order to preserve the model definitions.

1–3. The creation of sponsor name and item lists and identifiers as well as the creation of cost-component and labels for output are explained in detail in diagrams 6.1–6.3.

4. Arrays of the correct length and size are created for later cost-benefit ratio and cross-weighting computations. Cost and benefit arrays must have the same number of rows as the total number of cost items which were entered in step 3.

5. Sponsor cost items may be entered at step 3 in a different order from the input of sponsor names at step 1; therefore, the numerical identifiers must be sorted for compatibility.

6. Display a message to remind the user that cost and benefit values are now zero and should be updated.
Extended Description
1. Labels for the sponsor groups are entered directly from the keyboard by the user. The number of characters allowed per name label is predefined and therefore constant.

Diagram 12.2 describes the process by which a label is entered and returned.

2. If no name is specified by the user, a blank or zero is returned from the labels subroutine, the entering of labels process ends.

3. The sponsor name list is a contiguous array created and updated in this function. If the name entered in step 2 is the first name, the list is created; otherwise, the list is updated.

4. A pre-determined maximum for the number of sponsors is used in this step.

5. Normally, steps 1–4 are repeated until the user fails to enter a non-blank label. At this point a query prompts the user for completion of list.
2. The labels entered in step 1 are stored in the cost-labels array. Also, the label "other" for other costs is added to the array.
1. Prompt the user to enter an item name.

2. If no name is specified, repeat from 1.

3. Update ITEM name list and sponsor identifier vectors.

4. If the number of items for this sponsor is less than the maximum number allowed, repeat from 2.

5. Otherwise, display a "MAX" message and repeat 1.

6. Return.

Extended Description

1. Prompt the user for the name or the index number of the sponsor group for the next set of items to be entered. This is done so that the user may start specifying items for particular sponsors in any desired order.

4. On first time through, the item list and sponsor-identifier number (for each item) vectors are created. As new item labels are entered, the item name list is expanded, the associated sponsor group number is added to the identifier vector, and an encoded sponsor-item number is added to the sponsor-item identifier list.

The sponsor-item identifier numbers will be used in the program to determine index locations in cost/benefit matrices and for display purposes.
Extended Description

1. Compute the index values for sponsor identifiers of increasing numerical order and save these values as the new sorting order.

2. Rearrange the cost item names by the new sorting order of index values. Replace the old item list with the new sequenced list.

3. Rearrange the sponsor identifier number for each item by the new sorting order index values. Replace the old identifier list. Rearrange the sponsor-item encoded list also.

Note that the sponsor-identifiers vector contains an element for each item in the new sequence order of items: each element is the index value of the sponsor name in the sponsor name list.
1. Prompt for the type of data to be entered; display optional selection menu for entering benefits or costs. If no selection then do step 4; otherwise do step 2 or step 3 and repeat from 1.

2. Enter sponsor benefits.

3. Enter costs by sponsor group.

4. Return.

Extended Description

This routine is called when the user selects the option to enter data from the main options menu.

RAM (cost-benefit) matrix and the COST components matrix are created prior to the calling of this routine.

For the detail descriptions of sub-processes 2 and 3, see diagrams 7.1 and 7.2.
1. Initialize a cost item index value K.

2. Increment cost item index value K by one.

3. Prompt for the relative benefit value.

4. Store the cost item K's benefit value in the RAM matrix.

5. If the index K is less than or equal to the number of cost items, repeat from 2.

6. Return.

Extended Description

1. An index value is initialized so that it may be used to display the appropriate item names and list identifier numbers in step 3 and to update the associated benefit value in the RAM matrix (step 4.)
1. Determine the sponsor group for which costs are to be entered; if none, do 5.

2. Prompt for cost values for each sponsor item.

3. Store cost values in the cost matrix.

4. Store the total costs per sponsor item in the RAM matrix.

5. Return.

Extended Description:

1. Prompt and obtain from the user the next group for which cost values are to be entered.

2-3. For the selected group, prompt the user for cost values for each cost component per item. Repeat for each item within the selected sponsor group.

4. Compute the total cost over the components for each item and store in the RAM matrix.
1. A brief description of the manner in which cross-sponsor weights are to be entered by the user is displayed.

2. A weight vector is created with the number of elements equal to the number of sponsors. The initial value of all vector elements is one.

3. The user is prompted for the identifier numbers of the most significant sponsor item. The most significant sponsor cost item will be given a weight = 1. All other significant items in a different sponsor group will be weighted relative to this item.

Steps 4–6 are repeated until a cross-sponsor weight has been specified for each sponsor group.

4. Call subroutine LOCATE (see diagram 12.5) to get the next sponsor item to be weighted.

5. Call subroutine ENTERLINE (see diagram 12.3) to obtain the weight value from keyboard input. The value should be less than one and greater than 0.

6. The benefit value (obtained from the RAM matrix) for the most significant item is multiplied by the cross-sponsor weight. This new value is then divided by the benefit value for the item just weighted and the result is stored in the appropriate WEIGHT element for the sponsor group to which the item belongs.
1. Display the edit selection list and prompt the user for choice of processing. If no selection, do step 7; otherwise, do one of steps 2 – 6 and repeat from 1.

2. Edit sponsor benefits. 9.1
3. Edit item labels. 9.2
4. Add a sponsor item. 9.3
5. Delete a sponsor item. 9.4
6. Edit cost data. 9.5
7. Return.
5. The new value may be entered directly from keyboard and then converted, tested for validity, and stored. As an alternative, the "ENTERLINE" routine (diagram 12.3) may be used to obtain the new values.
1. Display a heading for the edit mode.
2. Prompt the user for the sponsor item identifier.
3. If no number specified, do step 7.
4. Display the current item label and prompt for a new label.
5. Store the new label in the item name list.
6. Repeat from step 2.
7. Prompt for edit completion; if not complete, repeat from step 2. Otherwise, return.

System/Program: BUILDGRAM
Diagram ID: 31-2
Description: Edit Item Labels
Name: [Blank]
1. Display a heading for new item entry mode.

2. Obtain the sponsor and new item numbers; if none, do step 9.

3. Obtain the new item label and add to name list.

4. Add sponsor-item to the sponsor-item identifier list.

5. Determine the new sequence order for sponsor-item numbers.
6. Sort sponsor-item numbers.

7. Add a row to the COST and RAM matrices. Then sort matrices and item name list.

8. Repeat from step 2.

9. Return.
1. Display a heading specifying mode of operations.

2. Determine the sponsor-item which is to be deleted: if none is specified, do step 5.

3. Display the item name and prompt the user again for item deletion.
   a. If the user does not want to delete this item, repeat from 2.
   b. If this item is the correct one to be deleted, modify the affected model variables:
      - delete the item identifier, item name and sponsor-item numbers from lists
      - delete associated rows of the RAM and COST matrices

4. Repeat from step 2.

5. Return.
1. Display a heading for edit mode.
2. Display the cost-component labels for the model.
3. Prompt the user for the sponsor-item identifier.
4. If no number, do step 8.
5. Obtain the new input cost values.
6. Update the cost matrix with the new values.
7. Repeat from step 3.

Extended Description

9. Sum the costs across cost-components for each item and store these values in the appropriate column of the RAM matrix.
8. Prompt for edit completion. If not complete, repeat 2.
9. Update RAM matrix with total costs and return.
1. Overall benefit values are computed by multiplying each sponsor-item benefit values by its fractional component in the WEIGHTS vector.

2. Normalize the overall benefits across items.

3. For each sponsor group: rescale all input benefit values within the sponsor group relative to the largest benefit value.

4. Rescale overall benefit values relative to the largest overall benefit value.

5. Determine the "Must Buy" order.

6. Compute RANK vector.
1. Compute the sum of the vector elements.

2. Divide each element by the sum.

3. Multiply the vector elements by 100 and return the result.

Extended Description

1. The input to this routine is a vector or row of numbers (otherwise, input will be converted to a numerical string of values). The sum of all the numbers in the vector is computed.

2. This step rescales all of the values to their percentage of the sum.

3. The normalized result from step 2 is typically multiplied by 100 to preserve percentage figures for output to the calling routines.
1. Create a vector with the same number of elements as there are number of items in the current model. Initialize the vector elements to zero.

2. Invoke the LOCATE subroutine (see diagram 12.5) to obtain from the user the sponsor-item identifier numbers and thereby derive the item index value for the requested item.

3. Display the item name and prompt the user again for corrections. If corrections, repeat 2.

4. If no corrections are needed, add the identifier number to the Must Buy order list. Repeat 2.

5. Return.
Extended Description

1. The user is prompted to specify a sponsor group name (or number) for the analysis. If a blank entry is given, then the routine terminates at step 9. Otherwise, steps 2–8 are processed.

4. The sponsor's item benefit values are all divided by the sum of the sponsor's item benefit values and then multiplied by 100.

5. The set of item index values are obtained which specify the numerical decreasing sequence order of benefit values. Another set of item index values are obtained which specify the increasing order of the computed total cost to benefit ratio per item.

6. The benefit values and the total cost values for the items which are indexed by the decreasing (step 5) benefit order are accumulated. Next, the benefit and total cost values for items which are indexed by the increasing C/B ratio order defined in step 5 are accumulated.

7. Cumulative values displayed are the first value of the specified order, then the first plus the second value of the indexed order, etc. . .
<table>
<thead>
<tr>
<th>Ordered by Benefit</th>
<th>Cumulative Benefit</th>
<th>Cumulative Cost</th>
<th>Ordered by Cost-Benefit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Career Placement CTR</td>
<td>35 44.0</td>
<td>44.0</td>
<td>35 Career Placement CTR</td>
</tr>
<tr>
<td></td>
<td>73.0</td>
<td>42 Extracurricular</td>
<td></td>
</tr>
<tr>
<td></td>
<td>125.0</td>
<td>53 Student Center</td>
<td></td>
</tr>
<tr>
<td></td>
<td>156.0</td>
<td>58 Housing Guide</td>
<td></td>
</tr>
<tr>
<td></td>
<td>266.0</td>
<td>74 Counseling Program</td>
<td></td>
</tr>
<tr>
<td>Health Services</td>
<td>61 322.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Counseling Program</td>
<td>77 432.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Student Center</td>
<td>88 484.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extracurricular</td>
<td>95 513.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Housing Guide</td>
<td>100 544.0 544.0 100 Health Services</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Please return carriage to continue.

HIPO Figure 11.1
COST-BENEFIT VS. BENEFIT-ONLY DISPLAY
Extended Description

Generalized routines are directly invoked by functional procedures and return to the calling programs.
This routine is used for display purposes to enhance user control over changing displays.

3. Any keyboard input or a specific response may be requested of the user before processing can continue.
1. Prompt the user for label input after displaying the label number/ID.

2. Strip the label of trailing blanks.

3. Check the length of the input label.
   a. If the length of the label is greater than the maximum, display an error message and repeat from step 1.
   b. Return the valid label.

Return
1. Prompt the user via the DISPLAY for K input values.

2. Screen and connect input to numerical values.

3. Return K numerical values.
1. Compare terminal input values with valid numerical character set, retaining only the valid input.

2. If the input values are blank, set the return value to zero.

3. Otherwise, translate the character input to numerical values and return these.
1. Display a request to the user to input the sponsor identification number and the sponsor item number for the desired cost item.

2. Call the CHECKNUMS subroutine (see diagram 12.4) for a screening of the input values. The subroutine returns numerical input that has been decoded from the terminal keyboard input field.

3. If the resulting numerical input is zero, return a zero value to the calling routine. Also, the numerical values should be compared with the valid sponsor and sponsor item identifiers in the model; invalid input results in a zero return value.

For valid input values, the sponsor and item numbers are combined to allow the computation of the appropriate index value for this item into the item name list, RAM and COST matrices.
1. Display the current set of vector numbers.

2. Prompt the user for an update to the same display line of numbers.

3. Screen input values and check numbers.

4. Return the new set of numbers.

Extended Description

This routine differs from the "ENTERLINE" routine in that it attempts to read input values from the previously displayed line location.
1. The title is passed to this routine so that the display will remain in context with the processing function. For example, a title may be 'DISPLAY RESULTS'.

2. The selections that describe what is optimal are passed as input and are displayed in a list or cookbook MENU format along with item sequence numbers.

3. Prompt the user for the item sequence number of the choice selection.

   Check the validity of the user input.
The REPRAM subsystem may be entered from system control separate from the BUILD/DISPLAY section of the program. The functional procedures described in diagrams 2.0, 3.0, and 4.0 of the BUILD/DISPLAY section describe steps 2, 9, and 4, respectively of this diagram.

2. The sort listing order is a pre-defined report output sequence of item identification numbers and has been defined according to a previous ordering (sorting); for example, decreasing Cost/Benefit ratio values is one such ordering. When the program begins, this step sets the sort listing order to depict increasing sponsor-item identifier numbers.
1. Display a message to ready the desired RAM data file on the selected storage device.

2. Wait for a user response indicating device/file is online and ready.


4. Read in model definition variables, data matrices and results.

5. Return.

4. Model variables are read according to a preset order which is identical to the manner in which the variables are stored (see diagram 4.2).

Extended Description

1. The device address of online storage which contains the RAM data and model definitions is available at the start of program processing. It is either encoded in the program or requested from the user. It is assumed that only one RAM data set is available per dev/file address (i.e., diskette, drum cylinder, etc.)

3. A system device OPEN command is required in order to access the RAM model and data. The interface with the system OPEN routine should be OPEN's setting of I/O flags in (or adjacent to) and I/O buffer area to denote the success or failure of access to the device.

If an error condition is detected, a non-zero error code is returned to the caller.
1. Issue a CLOSE command for the storage device (file) containing the RAM model data and definitions.

2. Test for ERROR conditions.
   a. If no error, set the return code to zero.
   b. If an error is detected, set return code to a non-zero error code and display an error message.

3. Delete I/O Buffer and RETURN.
Extended Description

1. The user has the option of switching the online device/file at this time, since only one RAM model is stored per device/file.

2. If the desired device is not already online, then the appropriate one must be placed online and opened for input (see diagram 2.0). An input buffer is created for the newly selected model.

3. Read data commands are issued for the required model definition and data variables. This is accomplished according to an encoded variable list and in the same order as the data items are stored.
1. Display optional displays menu and determine desired process from the user. If none, do step 6. Otherwise, one of steps 2 - 5 is invoked. Repeat from 1.

2. Display overall results.

3. Display sponsor results.

4. Display overall costs.

5. Display sponsor costs.

6. Return.
Extended Description

The format for the displays requires the listing of results for a maximum of ten (10) items at a time. The user is allowed to discontinue the display of results after each set of ten items is reviewed.

1. The index numbers of all the sponsor items in the currently loaded model are arranged by the currently specified sort listing order. (This sorting may have been changed recently via the SORT DISPLAYS option of the main program's MENU).

2. Display of the item results are preceded by the model name and result-type labels, such as "cost," "overall benefit," and "rank."

3. Return.

b. The information displayed per item includes the following:

- sponsor-item identifier
- item name
- benefit value
- overall (weighted) benefit value
- total cost per item
- cost/benefit ratio
- rank of C/B ratio among all items
1. Display the list of sponsors and prompt user for the desired sponsor group. If none specified, do step 5.

2. Determine index values of only the selected sponsor group items.

3. Display results according to the prior sorting for selected group items in sets of ten (or less).
   a. If all items of the sponsor have been displayed, do step 4. Otherwise, display next 10.
   b. Display headings for output.
   c. Display results on a single line of output.
   d. Prompt the user for continuation of more output. If continuation is desired, repeat 2a.

4. Repeat from step 1.

5. Return.

Extended Description

3. C. The information displayed per item includes the following:
   - sponsor-item identifier number
   - item name
   - sponsor benefit value
   - overall (weighted) benefit value
   - total cost for the item
   - the computed cost/benefit ratio
   - the rank of the C/B ratio among all sponsor items
1. Determine index values for items in sets of ten (or less) according to sort listing order.

2. Display next set of ten items with cost values.
   a. If all items have been displayed, do step 3.
   b. Display headings and cost-components of the model.
   c. Display on a single line:
      - sponsor-item identifier
      - item name
      - item cost for each component
   d. Prompt user for continuation. If user wants to see more items, repeat from 2. Otherwise, do step 3.

3. Return.
1. Determine which sponsor group is to be displayed.

2. Display costs for the selected sponsor group items in sets of ten (or less) items. Display according to sort order.
   a. If all sponsor items have been displayed, do step 3.
   b. Display headings:
      - sponsor name, cost components in model analysis
   c. Display on single line:
      - sponsor-item identifier
      - item name
      - cost for each component
   d. Prompt user for continuation. If user wants to see more, repeat from 2 with next set of items.

3. Return.
1. Display a message to user to get printer online and ready; wait for a response.

2. Print out the overall results according to sort listing order.
   a. Print headings for benefits, total costs, C/B ratio and rank order.
   b. Print overall results for each item, one line of output per item.

3. Print out the cost component values. Print by sort listing order.
   a. Print headings for cost components of analysis.
   b. Print cost for each component on a single line of output per item.

4. Prompt the user to disable printer output and wait for a response.

5. Return.
1. Recompute cost/benefit ratios.

2. Determine the variable on which sorting depends.

3. Determine whether the desired listing arrangement is for increasing or decreasing order.

4. Compute sort order index values according to the user selection in steps 2 and 3.

5. Replace the previously set sort listing order.

6. Return.

Extended Description

1. Editing may have changed the previous C/B ratios.

2. List of available SORT options includes the following:
   a) by sponsor item number
   b) by sponsor benefit value
   c) overall benefit values
   d) total cost values
   e) cost-benefit ratio
   f) rank order
INPUT

PROCESS

1. Display instructions for entering/editing and requesting displays of rationale.

2. Determine the sponsor item for the rationale. If none, do step 6.

   LOCATE

3. Display the current rationale.

4. Prompt the user for updates.
   a. If no update, do 5.
   b. Determine the line number for the update and prompt for the update.
   c. Read in the rationale.

5. Write out the rationale and repeat 2.

6. Return.

OUTPUT

Extended Description

Rationale text is usually limited to a pre-formatted display page/screen size which is compatible with the operating computer system.

The number of lines of text available for each sponsor item (and/or the number of available pages of text) may be determined by on-site systems personnel.

2. The LOCATE subroutine is described in diagram 12.5 of the “BUILDRAM subsystem” HIPO documentation.
1. Display the sponsor-item identifier and item name as headings.
2. Read/get a display page full of text. If error, do step 5.
3. Format the text so that it fits on the display.
4. Display the rationale.
5. Return.

Extended Description

4. If no rationale has been specified previously for the sponsor-item, then a page screen of blank lines should appear.
1. Determine the line number for next text update. If none, do step 6.

2. Prompt user for update to specified line of text.

3. Read display screen line input and update text.

4. Display updated text line.

5. Update rationale and repeat from step 1.

6. Return.
1. Format the current item rationale for storage output.

2. Determine the location for output.

3. Write text to storage in write/update mode.

4. If error on write, display error message and wait for user acknowledgement.

5. Return.