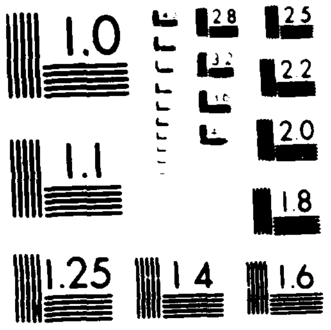
AD-A179 339 A DECISION SUPPORT SYSTEM FOR DATABASE MANAGEMENT SYSTEM SELECTION(U) AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING 1/1

UNCLASSIFIED D R DAVIDSON DEC 86 AFIT/CS/ENG/86D-12 F/G 5/2 NL



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

DTIC FILE COPY

1

AD-A179 339

A DECISION SUPPORT SYSTEM FOR DATABASE MANAGEMENT SYSTEM SELECTION

THESIS

Dennis R. Davidson
Captain, USAF

AFIT/GCS/ENG/86D-12

DTIC
S ELECTE D
APR 2 0 1987
A

This document has been approved for public release and sale; its distribution is unlimited.

DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

87 4 16 071

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION: UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT: Approved for public release; distribution unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S): AFIT/GCS/ENG/86D-12	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION: School Of Engineering — 6b. OFFICE SYMBOL *(If applicable)*: AFIT/ENG	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS *(City, State, and ZIP Code)*: Air Force Institute of Technology, Wright-Patterson AFB, Ohio 45433	7b. ADDRESS *(City, State, and ZIP Code)*
8a. NAME OF FUNDING/SPONSORING ORGANIZATION — 8b. OFFICE SYMBOL *(If applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State, and ZIP Code)*	10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*

A Decision Support System For Database Management System Selection

12. PERSONAL AUTHOR(S)
Davidson, Dennis Richard, Captain, USAF

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Year, Month, Day)*	15. PAGE COUNT
MS Thesis	FROM ________ TO ______	1986 December	74

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
05	02	

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Databases, Decision Theory, Management Information Systems

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

(see reverse)

[illegible] for [illegible] release: IAW AFR 190-1
[illegible] 5 Mar 87
[illegible] Development
Wright-Patterson [illegible]

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT: [✓ UNCLASSIFIED/UNLIMITED] ☐ SAME AS RPT ☐ DTIC USERS	21. ABSTRACT SECURITY CLASSIFICATION: UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL: Wade H. Shaw, Captain, USA	22b. TELEPHONE *(Include Area Code)*: 513-255-2057 — 22c. OFFICE SYMBOL: AFIT/ENG

19. An interactive micro-computer based decision support system was designed and developed to aid in the evaluation of Database Management Systems (DBMS). DBMS attributes were researched in order to develop a hierarchy of attributes that reflect the important factors that must be considered when implementing a DBMS. After the hierarchy was established, different DBMS were evaluated relative to the attributes, and a base of knowledge was established.

The Decision Support System for DBMS Selection (DSSDS) uses the Analytical Hierarchy Process (AHP) to allow the user to create a scenario that represents his requirements, and then evaluates this scenario against the base of knowledge.

Several problems including an incomplete hierarchy and excessive input requirements were identified and solutions were implemented in the system. A preliminary evaluation indicates that users with database experience find the system responsive and usable, while those without database experience were easily confused. Several suggestions for further research are proposed.

(1)

AFIT/CS/ENG/86D-2

A DECISION SUPPORT SYSTEM FOR DATABASE MANAGEMENT SYSTEM SELECTION

THESIS

Dennis R. Davidson
Captain, USAF

AFIT/CS/ENG/86D-12

Approved for public release; distribution unlimited

AFIT/CS/ENG/86D-12

A DECISION SUPPORT SYSTEM FOR DATABASE MANAGEMENT SYSTEM SELECTION

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Computer Systems

QUALITY INSPECTED 2

Dennis R. Davidson, B.S.

Captain, USAF

December 1986

A1

Approved for public release; distribution unlimited

ABSTRACT

An interactive micro-computer based decision ıpport system was designed and developed to aid in the evaluation of Database Management Systems (DBMS). DBMS attributes were researched in order to develop a hierarchy of attributes that reflect the important factors that must be considered when implementing a DBMS. After the hierarchy was established, different DBMS were evaluated relative to the attributes, and a base of knowledge was established.

The Decision Support System for DBMS Selection (DSSDS) uses the Analytical Hierarchy Process (AHP) to allow the user to create a scenario that represents his requirments, and then evaluates this scenario against the base of knowledge.

Several problems including an incomplete hierarchy and excessive input requirements were identified and solutions were implemented in the system. A preliminary evaluation indicates that users with database experience find the system responsive and usable, while those without database experience were easily confused. Several suggestions for further research are proposed.

Preface

The purpose of this study was to develop a model for evaluating candidate DBMS, and rating the DBMS with respect to a user's needs and requirements. The model was implemented as a Basic program written for PC compatible computers.

I chose this topic based on my interests in database technology and optimization techniques. I am deeply indebted to my faculty advisor, Captain Wade Shaw, for his ability to take my interests and convert them into a useful DBMS selection tool. Finally, I wish to thank my wife Venessa for her understanding and patience.

Dennis Davidson

Table of Contents

List of Figures

List of Tables

List of Acronyms

Acronym	Meaning
ACI	Average Consistency Index
AHP	Analytical Hierarchy Process
CI	Consistency Index
CR	Consistency Ratio
DBMS	Database Management System
DSS	Decision Support System
DSSDS	Decision Support System for DBMS Selection
ORW	Overall Relative Weight
QP/AP	Query Processing/ Applications Programming

A DECISION SUPPORT SYSTEM FOR DATABASE MANAGEMENT SYSTEM SELECTION

I. Introduction

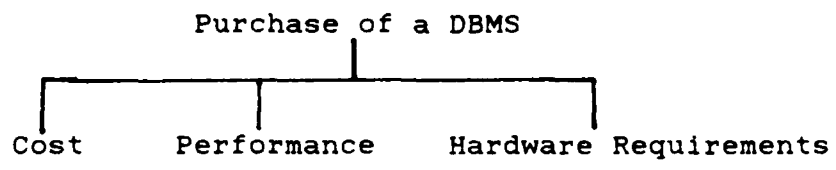
Choosing the best Data Base Management System (DBMS) to purchase is a problem which occurs regularly, and has no clear cut solution. Every manager who is looking at purchasing a DBMS must determine what features of a DBMS are important for his application, and which accessible DBMS has those features available. In implementing a data base, "The goal is to lay down a fundamental data framework that will serve the organization for many years to come" (3:164). The correct DBMS is critical to achieving this goal.

There are numerous reasons why selecting a DBMS is a difficult problem. Aside from the fact that a DBMS is one of the most complex varieties of software in existence (15:2), the selection of a DBMS is usually done under severe time and cost restrictions (9:34). While lack of time and money hinders DBMS evaluation, complexity is the biggest problem in choosing a DBMS. Even the most simplistic view of a DBMS would require five different areas of concern: efficient access to secondary storage, protection and concurrency control, automatic integrity maintenance, crash recovery services, and human-computer interfaces (7:28).

Another major reason why selecting a DBMS is a difficult problem is because users can not accurately express their needs. Using the abbreviated hierarchy of Figure 1-1, users were asked to weight their preferences for each of the three attributes with respect to the overall goal of purchasing a DBMS. They were told to select a DBMS to perform the functions of a student scheduling database which includes: students, advisors, classes and theses (More details on the survey can be found in Appendix A).

FIGURE 1-1

THE DSSDS HIERARCHY



The users were asked to weight each of the three attributes such that the three weights summed to 100. Then the users were asked to do a pairwise comparison of the attributes using the method that will be explained in Chapter 2. These pairwise comparisons were then converted into weights through a process Saaty calls synthesis (12:81) which will also be explained in Chapter 2. The average weights that resulted from this pilot study are shown in

Table 1-1. The initial results indicate that users give too much weight to less important attributes, and not enough weight to the more important attributes. However, in order to prove that the means were different, paired-t tests were completed on the data.

TABLE 1-1

COMPARISON OF RELATIVE WEIGHTS

CRITERION	DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	t VALUE
COST	26.5	18.4	2.687**
PERFORMANCE	56.5	69.1	4.906**
HARDWARE REQUIREMENTS	17.0	11.5	4.490**

** For a confidence level of 0.95 and a sample size of 10, t must be greater than 1.833 to reject the hypothesis.

TABLE 1-2

COST DATA

DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	DELTA
40	15.8	24.2
40	27.9	12.1
40	48.7	- 8.7
30	21.8	8.2
30	14.3	15.7
20	20.7	- 0.7
20	11.7	8.3
20	11.1	8.9
20	5.8	14.2
5	6.2	- 1.2
26.5	18.4	8.1

Using the data for cost in Table 1-2, the difference between the means ($\bar{x}_D$), and the standard deviation (sigma) were computed to be:

$$\bar{x}_D = 8.1 \tag{1}$$

and

$$\text{sigma} = 9.53 \tag{2}$$

The t statistic was then computed using:

$$t = \bar{x}_D /(\text{sigma}/(n)) \tag{3}$$

For the attribute cost, this yielded:

$$t = 8.1/(9.53/3.16) \tag{4}$$

$$= 2.687 \tag{5}$$

Using a table of values for t distribution, the hypothesis that the means are equal can be rejected at the 95% confidence level if the computed t value is greater than 1.833 or less than -1.833. Since 2.687 is outside this range, the data reflects a 95% confidence level that the means for cost are not equal. Similarly, the t values for performance and hardware requirements were -4.906 and 4.490 respectively. These values are also well beyond the range and imply that their means are not the same (tables of values for performance and hardware requirements and all calculations are shown in detail in Appendix A).

This data shows that users need assistance in determining the relative importance of attributes. Although there are other ways to perform this function, pairwise comparison was the method used in this example and throughout the research. After collecting the data, a method is needed for actually choosing the DBMS.

There are many methods for choosing a DBMS. It has been recommended that an organization form special teams to choose a DBMS for each area's application (1:128). But what about the user who does not have an organizational team to assist him in selecting a DBMS, or a small organization which can not spare the manpower to form a team? There must be a better way to assist them in selection of a DBMS. A cost effective solution is a computer aided Decision Support System (DSS) that will assist the user in first analyzing his data to determine the type of DBMS required and then evaluate all candidate DBMS relative to that user's requirements.

In order to be effective, a DSS must satisfy three basic requirements. It must provide a base of knowledge, be easy to use, and be readily accessible (6:66). The complication arises when the DSS is used to replace an evaluation team. The DSS must perform the duties of all of the team members: hardware manager, software manager, model builder, and end user (14:171). The DSS can accomplish these different functions because it can objectively evaluate the available information. It is an algorithmic approach to deciding which DBMS is preferable. The prime advantage of a DSS is that it does not choose the best overall DBMS, but will select the DBMS that is best suited to fulfilling a specific user's objectives.

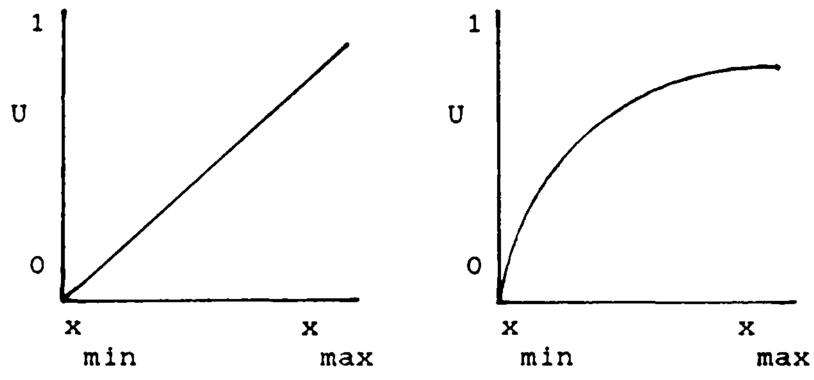
The intricacies involved in designing a DSS to accurately meet a user's objectives can be attacked at several different levels. The first approach assumes that the problem is linear. This means that each DBMS attribute is assigned a weight which corresponds to the significance of that attribute in the user's application. Each DBMS is then evaluated relative to each attribute and assigned a value corresponding to its strength or weakness. The value is then multiplied by the weight assigned to that attribute. A numeric evaluation of each DBMS is provided by summing the weighted value of each attribute. However, it has already been shown that the assignment of weights to attributes is a difficult and error prone process. The result is a method that will provide a user with a recommendation on which DBMS evaluated is the best for his perceived application; however, it may not recommend the DBMS which is actually best-suited to meet his needs.

A second approach is to provide the user with a selection of nonlinear utility curves, and allow him to select the curve which best meets his needs. This is the approach taken by the DSS for Robot Selection (6:67) where fourteen different curves are provided, and the user is allowed to select the curve which best meets his actual requirements. This approach allows the user to not only determine the importance of different attributes, but also to determine the appropriate utility function to scale that importance.

For example, a user who is purchasing a DBMS might feel there is a significant difference between DBMS A which requires 96K of memory and DBMS B which requires 128K. However, the difference between DBMS C which requires 256K and DBMS D which requires 320K might be nearly insignificant. A more suitable curve might be the nonlinear utility function in Figure 1-2.

FIGURE 1-2

LINEAR AND NONLINEAR UTILITY FUNCTIONS



x_{min} denotes worst case, x_{max} denotes best case.

While this method is an improvement over a linear model, it is not the optimal method since it still provides only a finite number of choices. A preferable method for implementing a DSS is to allow the user to input multiple critical values, and then compute a curve that exactly matches these values. That is, besides maximum and minimum values, the user would also be required to provide numerous intermediate values and a best fit curve could be computed rather than arbitrarily selected. The difficulty in this method lies in finding a simplified approach which allows the user to input all of these critical values without becoming overwhelmed by a voluminous amount of data.

The design of the DSSDS concentrated on the attributes in the hierarchy and on reducing the nonobjectivity of the user supplied inputs and was based on a linear model. This decision was based on the observation by Keeney and Raiffa that "You can go only so far without introducing subjective attitudes" (8:12). Allowing the user to weight the attributes is critical for establishing the specific user requirements, but asking the user to define a nonlinear relationship would drastically increase the complexity of the DSSDS, decrease the system objectivity and possibly compromise model validity.

Problem

A DBMS is a complex software system that must be capable of storing data and providing it to the user in a format that is readily usable. There are numerous DBMS available today, each of which a user could attempt to use to solve any DBMS application problem. Due to the complexity of a DBMS and the numerous systems available, it is very difficult to decide which DBMS is the best one for a particular application. Every DBMS user has something unique about his data that makes his requirements different from any other user. There is no evidence that a support system is available which will allow a user to interactively specify his requirements and then evaluate all candidate DBMS with respect to those requirements.

Scope

There were three steps performed in creating the Decision Support System for DBMS Selection (DSSDS). The different DBMS attributes were researched to develop a complete and consistent attribute set. Multiple criteria decision making concepts were reviewed to determine an appropriate method for DBMS evaluation with respect to the relationships between the selected attributes. The DSSDS was implemented as a DSS which allows for DBMS evaluation in a user specified environment. The DSSDS implements these three objectives using Saaty's Analytical Hierarchy Process (AHP) model (12).

Standards

All software was developed in accordance with the AFIT/ENG Software Development Documentation Guidelines and Standards (4). The design concentrated on thoroughness of evaluation and ease of use over response time. The reason behind this decision was that the DSSDS should be usable by an engineer, with no prior knowledge of how the system worked, in less than one week. Since this is replacing months of research, the time savings was significant enough that response time was not considered a critical factor.

Methodology

The method used for completing the DSSDS was a carefully structured software engineering life cycle. The four steps of requirements analysis, design, implementation and testing were completed. Each step was thoroughly documented and the final package includes a set of user instructions as well as on-line assistance.

The requirements analysis included four important areas of study:

1. Existing DSS were examined for their applicability to DBMS evaluation.
2. DBMS attributes were researched to determine those that were significant to DBMS evaluation.
3. Functions for rating DBMS attributes were established.
4. Data collection was performed to obtain the information required to evaluate each candidate DBMS.

After requirements analysis was completed, the DSSDS was designed in three phases:

1. A complete and consistent set of measurable DBMS

attributes was determined.

2. An AHP model was developed.

3. A database of DBMS attributes was designed.

Implementation consisted of building a user interface to the AHP model that allows the user to access the attributes in the database, and then use the model to evaluate each candidate DBMS.

Testing was accomplished through the evaluation of test cases for consistency and correctness. This evaluation was performed by DBMS selection experts. This is by no means a guarantee that the DSSDS will always select the best DBMS for a given application. However, it is used to show that the DSSDS is functioning correctly according to the design specifications.

Equipment and Support

The DSSDS was implemented on an existing Z-100 computer system using only the ZBASIC software package and the MS-DOS operating system. This prevents users from having to purchase any additional software to run the DSSDS and provides the opportunity for maximum use of the system. There was no other equipment or support required for this project.

Sequence of Presentation

Chapter 2 is an explanation of AHP. It covers the four steps involved in the process, possible drawbacks of AHP, and the mathematics involved. Chapter 3 is a detailed analysis of the DBMS attributes that comprise the hierarchy and a brief comparison between the DSSDS and another AHP model for DBMS selection. Chapter 4 explains the software implementation of the DSSDS and provides a preliminary evaluation. Chapter 5 gives the conclusions that have been made concerning the problem of DBMS selection, and implementation of an AHP hierarchy, and discusses some suggestions for further research.

II. The Analytical Hierarchy Process

The Analytical Hierarchy Process (AHP) was developed by Saaty as a method for ranking alternatives. It was designed to organize feelings, intuition and logic into a structured decision making approach (12:6). Among the many specific applications which have used AHP are portfolio selection, marketing, projecting oil prices, microcomputer selection, and DBMS selection (12:137,209,212; 16:100-101). In general, AHP has been used for planning, resolving conflict, benefit/cost analysis and resource selection, and group decision making (12:2).

Four Step Process

The four steps required to use AHP are hierarchy construction, paired comparisons, evaluation with eigenvalues, and synthesis (5:373). The first step in AHP is always construction of the hierarchy, which involves breaking the problem down into manageable pieces. The hierarchy that was used in the DSSDS will be discussed in Chapter 3. The hierarchy was structured such that the ultimate objective was at level 1, and the alternatives compromise the lowest level of the hierarchy. The levels in between contain attributes which contribute to the quality of the decision (16:97).

Incompleteness

One possible problem that was considered when constructing the hierarchy was incompleteness. If the hierarchy is constructed such that "each item on one level is actually linked to all possible items on the next (5:375)," then it is complete; otherwise the hierarchy is incomplete. Since the DSSDS hierarchy is incomplete by inspection, it had to be determined if the incompleteness provided counterintuitive results. Briefly, an incomplete hierarchy will give counterintuitive results if any of the lower level attributes which comprise an upper level attribute are mutually exclusive. In other words, if an attribute can be completely represented by less than the total number of attributes it is comprised of, then the lower level attributes as criteria are mutually exclusive for the upper level attribute of concern. Assume a DBMS had an initial cost and either a maintenance cost or an upgrade cost, but not both. Then the attribute cost in Figure 1-1 could be comprised of initial and maintenance costs, or initial and upgrade costs, but not all three. Since this is not the case, and none of the attributes in the DSSDS hierarchy are mutually exclusive, incompleteness is not a problem in this application.

Data Collection

The second step was to collect data through paired comparisons of the attributes at each level that contribute to the same element at the next higher level. The values that may be used in performing the paired comparisons are shown in Table 2-1.

Table 2-1(12:Table 5-1)

Intensity of Importance	Definition	Explanation
1	Equal importance of both elements	Two elements contribute equally to the property
3	Weak importance of one element over another	Experience and judgement slightly favor one element over another
5	Essential or strong importance of one element over another	Experience and judgement strongly favor one element over another
7	Demonstrated importance of one element over another	An element is strongly favored and its dominance is demonstrated in practice
9	Absolute importance of one element over another	The evidence favoring one element over another is of the highest possible order of affirmation
2,4,6,8	Intermediate values between two adjacent	Compromise is needed between two judgements
Reciprocals	If activity i has one of the preceding numbers assigned to it when compared with j, then j has the reciprocal value when compared with i	

Using an example from the DSSDS hierarchy, initial cost would be compared to maintenance cost and to upgrade cost since they are all part of cost, but initial cost would not be compared to error handling because it is an attribute of performance. The reason for using paired comparisons was because "direct assignments of weights is too abstract for the evaluator (16:98)," as was shown by the pilot study discussed in Chapter 1.

The input data is placed in a matrix of pairwise comparisons for all elements at one level that contribute to an element at the next higher level. For example, the elements initial cost, maintenance cost, and upgrade cost all contribute to the element cost. Therefore, the input matrix for cost looks like Table 2-2.

TABLE 2-2

SAMPLE INPUT MATRIX FOR COST

	INITIAL	MAINTENANCE	UPGRADE
INITIAL	1	A	B
MAINTENANCE	1/A	1	C
UPGRADE	1/B	1/C	1

Where the values for A, B, and C are the pairwise comparisons supplied by the user. The diagonal of the matrix is all 1's since each attribute is equally important to itself. Also, since the lower triangle elements of the matrix are reciprocals of the upper triangle elements, only half of the remaining elements need to have data collected.

Relative Weight Calculation

Step 3 of AHP is to estimate the relative weights of decision attributes using the "eigenvalue" method. This step is required because in step 2 the assumption was made that the evaluator could not directly assign weights. Therefore, step 3 will be used to convert the pairwise comparisons of step 2 into relative weights for each attribute.

If the evaluator had known the actual weights of the attributes in Table 2-2, then the matrix of Table 2-3, where W1, W2 and W3 are the actual weights of the three attributes, would have been the result.

TABLE 2-3

INPUT MATRIX USING ACTUAL WEIGHTS

	INITIAL	MAINTENANCE	UPGRADE
INITIAL	W1/W1	W1/W2	W1/W3
MAINTENANCE	W2/W1	W2/W2	W2/W3
UPGRADE	W3/W1	W3/W2	W3/W3

In these cases where the matrix is perfectly consistent, the relative weights can be taken directly from any row of the matrix. This implies that

$$A * W = n * W \tag{6}$$

where A is the matrix in Table 2-3, W is the vector of actual weights, (W1,W2,W3) , and n is the number of attributes. W is called the right eigenvector of matrix A and n is the eigenvalue.

However, since matrix A is presumed to contain inconsistencies, W must be approximated by

$$A' * W' = \text{lambda}(\max) * W' \tag{7}$$

where A' is the inconsistent matrix, W' is the right eigenvector, and lambda(max) is the largest eigenvalue. It can be shown that lambda(max) is always greater than or equal to n and that the closer lambda(max) is to n, the more consistent the values in A' are (11:850). The result is a value known as the consistency *index* (CI), which is computed as follows:

$$CI = (\text{lambda}(\max) - n)/(n-1) \tag{8}$$

and the consistency ratio (CR), which is:

$$CR = CI/ACI \tag{9}$$

where ACI is the average consistency index shown in Table 2-4. A CR value less than 0.10 is usually considered acceptable (17:102).

TABLE 2-4

AVERAGE CONSISTENCY INDEX

n	1	2	3	4	5	6	7	8	9	10
ACI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

The easiest way to comprehend step 3 is through a simple example. Assume that the pairwise comparisons for the cost attribute resulted in the matrix of Table 2-5.

TABLE 2-5
EXAMPLE INPUT MATRIX FOR COST

	INITIAL	MAINTENANCE	UPGRADE
INITIAL	1	1/4	2
MAINTENANCE	4	1	4
UPGRADE	1/2	1/4	1

The first step is to obtain a normalized matrix. this is accomplished by summing the three columns and then dividing each element by the sum of that column. The result is in Table 2-6.

TABLE 2-6
NORMALIZED MATRIX FOR COST

	INITIAL	MAINTENANCE	UPGRADE	ROW SUM	AVE R.S.
INITIAL	2/11	1/6	2/7	0.63	0.21
MAINTENANCE	8/11	4/6	4/7	1.97	0.66
UPGRADE	1/11	2/6	1/7	0.40	0.13

Then, by taking the row sums and dividing by the number of elements, an average row sum is obtained. This average row sum is in fact the relative weight of the attribute on that row of the matrix. Therefore, for this example, the relative weights for initial, maintenance, and upgrade costs are 0.21, 0.66, and 0.13 respectively.

Next, the consistency of these relative weights must be determined. This is accomplished by multiplying each attribute column of Table 2-5 by the relative weights in column 5 of Table 2-6, yielding the matrix in Table 2-7.

TABLE 2-7
COST MATRIX FOR CONSISTENCY CHECKING

	INITIAL	MAINTENANCE	UPGRADE	ROW TOTALS
INITIAL	0.21	0.17	0.26	0.64
MAINTENANCE	0.84	0.66	0.52	2.02
UPGRADE	0.11	0.17	0.13	0.41

The rows are summed and then divided by the relative weights to yield:

$$(0.64, 2.02, 0.41) / (0.21, 0.66, 0.13) = (3.05, 3.06, 3.15) \quad (10)$$

and

$$\text{lambda(max)} = (3.05 + 3.06 + 3.15) / 3 \quad (11)$$

$$= 3.09 \quad (12)$$

From (3), the result is

$$CI = (3.09 - 3) / 2 \quad (13)$$

$$= 0.045 \quad (14)$$

and from (4) the result is

$$CR = 0.045 / 0.58 \quad (15)$$

$$= 0.08 \quad (16)$$

Since this value is less than 0.10, the matrix in this example is considered consistent enough for use in a complete hierarchy.

Synthesis

Step 4 of AHP is the synthesis which results from taking the relative weights of each element in Step 3 and producing a composite weight for each alternative at the lowest level of the hierarchy. This weight can then be used to produce a rank ordered list of the alternatives.

In order to produce the rank ordered list, the values for each attribute of a DBMS were multiplied by the weight of that attribute in the hierarchy. The DBMS values were scaled prior to synthesis so that the attribute values did not bias the weights which were established to reflect the user's requirements.

Scaling the attribute values was accomplished by normalizing all attribute values to a number between zero and one. The method used to do this was simply dividing all values by the maximum value for that attribute. In cases where low values were preferable to high values, this number was subtracted from one to reverse the scale.

Zahedi's Model

Before leaving the AHP to discuss the DBMS attributes, reference should be made to a previous AHP model for DBMS evaluation and selection (17). This model was used to evaluate two DBMS on their functional aspects (How easy it is to create classes, etc.), and required 81 comparisons. This has several drawbacks, many of which were listed as areas for further research. First, it does not include any reference to cost or hardware requirements. Second, this method stressed comparisons at the lower level and did not allow the user to construct a unique environment by weighting the upper level attributes. Third, and probably most critical, since it could require up to an additional 81 comparisons for each aspect, it would require too much detail and would become unmanageable. It is this drawback which led to the DSSDS being designed at a higher level, to allow for evaluation of a DBMS prior to purchase, rather than evaluating a DBMS after it is being used. These issues were all considered and are reflected in the detailed analysis of DBMS attributes.

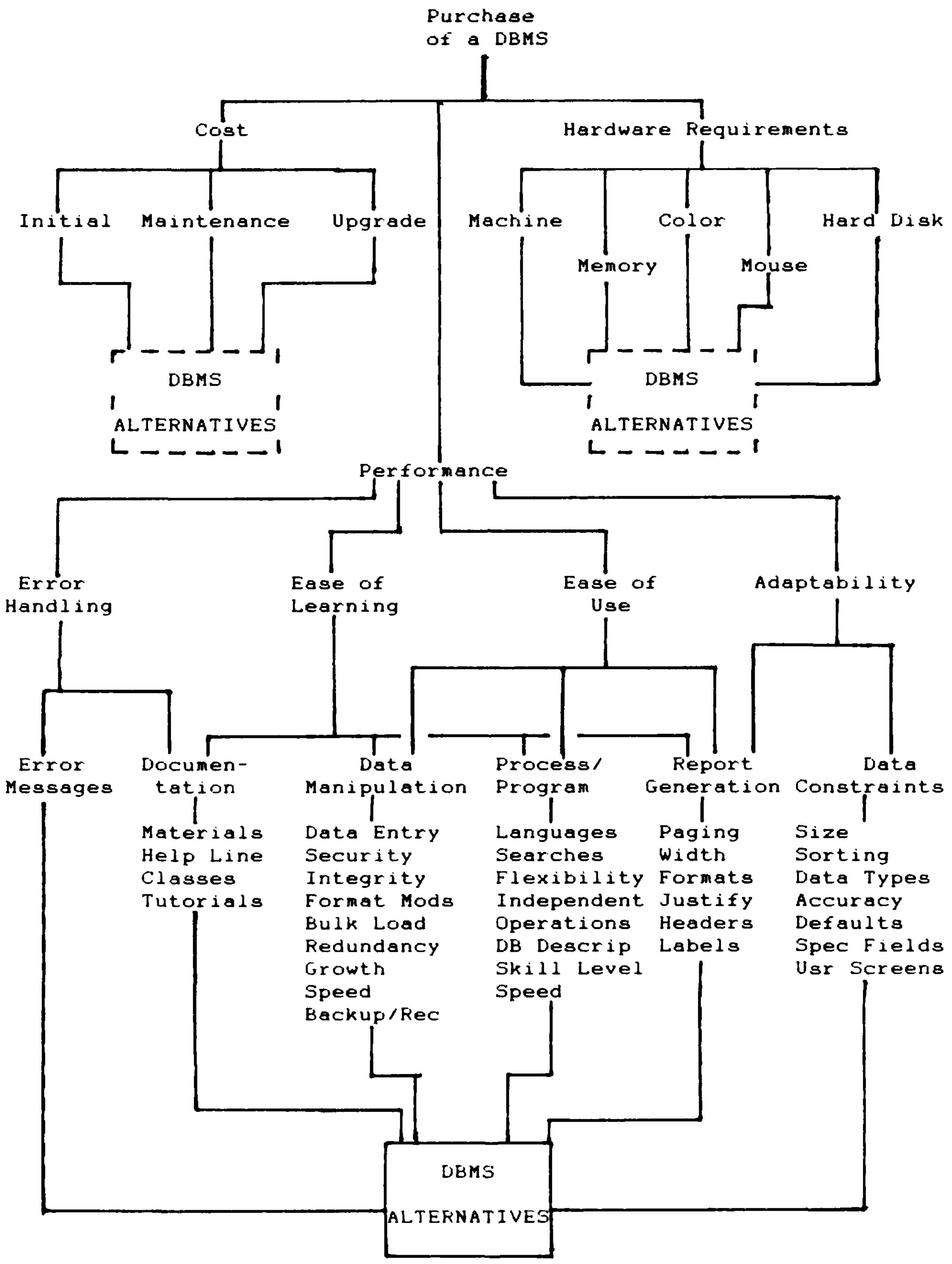
III. Detailed Analysis of Attributes

The detailed analysis explains the underlying methodology that the DSSDS was based on. This methodology used the AHP model as the framework for structuring the attributes that must be considered when selecting a DBMS. The detailed analysis explains these attributes in depth, showing how they relate to each other and why they are important in DBMS evaluation. Figure 3-1 shows the DSSDS hierarchy. The attributes in the hierarchy are a combination of the attributes that were used by The Software Digest, Cohen, and Palmer (2;10;13). There was no theoretical attempt to prove that the attributes and the hierarchy constituted a complete and irrefutable solution to the DBMS selection process; however, validation efforts are documented in Chapter 5.

Hurdle Rates

Before discussing the individual attributes, it is helpful to introduce the concept of attributes with hurdle rates. This concept implies that a DBMS will only be considered for evaluation if it meets the minimum requirements or hurdle rates. Two examples of hurdle rates might be initial cost or color. A user might want to eliminate any DBMS whose initial cost was greater than the

FIGURE 3-1
THE DSSDS HIERARCHY



amount of money available, and then rate the remaining DBMS based on their respective costs. Similarly, a user might not want to consider any DBMS which did not use color, and then base the evaluation of the remaining DBMS on how well they used color to enhance their product. Although hurdle rates originally appeared as an implementation constraint of the DSSDS, they are discussed here to show how attributes with overriding constraints can be handled without disrupting the structure of the hierarchy.

Description of Attributes

The AHP model used in the DSSDS starts with the overall objective, purchase of a DBMS, as the first level of the hierarchy, and works down to the DBMS being evaluated representing the lowest level in the hierarchy. The purchase of a DBMS was first broken down into the three second level attributes of cost, hardware requirements, and performance. Cost must include not only the initial purchase price of the DBMS, but also any additional charges that the user might incur. The hardware requirements define the types and amount of hardware that the system runs on as well as any special hardware features it can utilize. The performance attribute can best be described as how the DBMS interfaces with the user after it starts to execute. This includes things such as how easy the DBMS is to learn and use, and how easily it can be adapted to the user's data.

The attributes cost, hardware requirements, and performance are all compound attributes which can not be defined in terms of a single unit of measure. The breakdown of these attributes will be covered in subsequent sections.

Cost

The cost attribute was broken down into three third level attributes which can be quantitatively evaluated by a single unit of measure. The attributes are initial, maintenance and upgrade costs. The initial cost is the purchase price for the DBMS and is evaluated in terms of the dollar amount that is spent to procure it. The maintenance cost represents any fees which must be paid for supplier support of the DBMS software and is evaluated in terms of dollars per year. The upgrade cost is how much the software supplier charges to provide new version releases of the DBMS and is evaluated in terms of dollars per upgrade.

Hardware Requirements

The hardware requirements attribute can be broken down into five different third level hardware components. These hardware components are of two types: minimal requirements and user options. Minimal requirements specify hardware components without which the DBMS can not run, while user options are hardware enhancements which will make the DBMS more usable. The hardware requirements that represent minimal requirements are the machines that the DBMS can run on and the amount of memory required by the DBMS. The hardware requirements that represent user options are color, mouse and hard disk capabilities.

Evaluation of the machines that a DBMS can operate on is one dimensional. The DBMS will either meet the user's requirement or it will not. (It is possible to create a scenario in which a user owns more than one type of machine and wants to rate each DBMS based on the different machines; however, this capability is not currently considered in the methodology.) The memory requirement represents the minimal amount of memory required to operate each DBMS, The evaluation of memory requirements will be done by rating a DBMS high if it requires a minimal amount of memory, and rating a DBMS low if it requires a greater amount of memory.

The color and mouse capabilities are both one dimensional evaluations of whether the DBMS can use that function. The final hardware requirement is the ability to run the DBMS from a hard disk. This capability is important since it increases the speed at which the DBMS will execute. This evaluation is also one dimensional in that all DBMS with this capability are equal and will be rated higher than all DBMS without this capability. All five of the attributes which contribute to the hardware requirements attribute can be measured quantitatively, and none of them will be broken down into subsequent levels.

Performance

The performance attribute was broken down into four third level attributes: ease of learning, ease of use, adaptability, and error handling. All of these attributes were broken down into two subsequent levels of the hierarchy.

Ease of learning represents how much introductory support is provided for the DBMS and it was broken down into documentation, report generation, data manipulation, and query processing/applications programming (QP/AP). Ease of use is an approximation of the skill level required to effectively operate the DBMS and was broken down into report generation, data manipulation, and QP/AP. Adaptability is the usability of the system for storing and outputting the user's specific data requirements and was broken down into data constraints and report generation. Error handling represents how difficult it is to recover from errors and was broken down into documentation and error messages.

It should be noted that at this level of the hierarchy, the third level attributes were broken down into some of the same fourth level attributes. These fourth level attributes are only discussed once, but mention is made of how they apply to each of the level three attributes. The fifth level attributes which comprise each of the fourth level attributes are not discussed in detail, but their definitions can be found in Appendix B.

Fourth Level Attributes

Error messages was the only fourth level attribute that was not broken down into another level. It is evaluated by rating the DBMS on the quantity and quality of error messages that are provided. This attribute was only used in determining the error handling attribute of level 3.

Documentation was used in determining both ease of learning and error handling. Documentation was not considered in determining ease of use because during normal usage, the concern is in how much work can be done without the need for external references. Documentation was evaluated in terms of the quantity and quality of printed materials provided as well as the availability of help lines, classes, and on-line tutorials.

Data manipulation was used in determining ease of learning and ease of use. Data manipulation involves how the user can get data into the DBMS, the speed at which this can be performed, and the ability of the DBMS to adjust to changes in both content and format. The fifth level attributes which comprised data manipulation were data entry, security, integrity, format modifications, bulk load, redundancy, growth, speed, and backup/recovery.

QP/AP was also used in both ease of learning and ease of use. QP/AP is primarily how the user can interface with the DBMS through other software. This included the programming languages that could be used, the level of programming expertise required, and the speed at which these transactions take place. The fifth level attributes that comprised QP/AP were languages, searches, flexibility, independence, operations, database description methods, skill level, and speed.

Report generation was used in ease of learning, ease of use, and adaptability. Report generation is critical since the complete mechanization of society has not yet arrived and paperwork is still an integral part of the office environment. Providing printed output in a format that is convenient for the user is critical in making the DBMS an effective tool. The fifth level attributes that comprised report generation were paging, width, saving formats, justifying fields, headers, and mailing labels.

The final fourth level attribute was data constraints which was used in determining adaptability. Data constraints concerns the different types of data that could be entered into the DBMS. This included the maximum values allowed, the maximum accuracy to the right of the decimal point, and data such as monetary fields. Data constraints was comprised of the fifth level attributes size, sorting, data types, accuracy, default values, specialized fields, and user defined screens.

Now that the hierarchy has been explained in detail, it is necessary to show how this hierarchy has actually been implemented into a DSS.

IV. Software Development

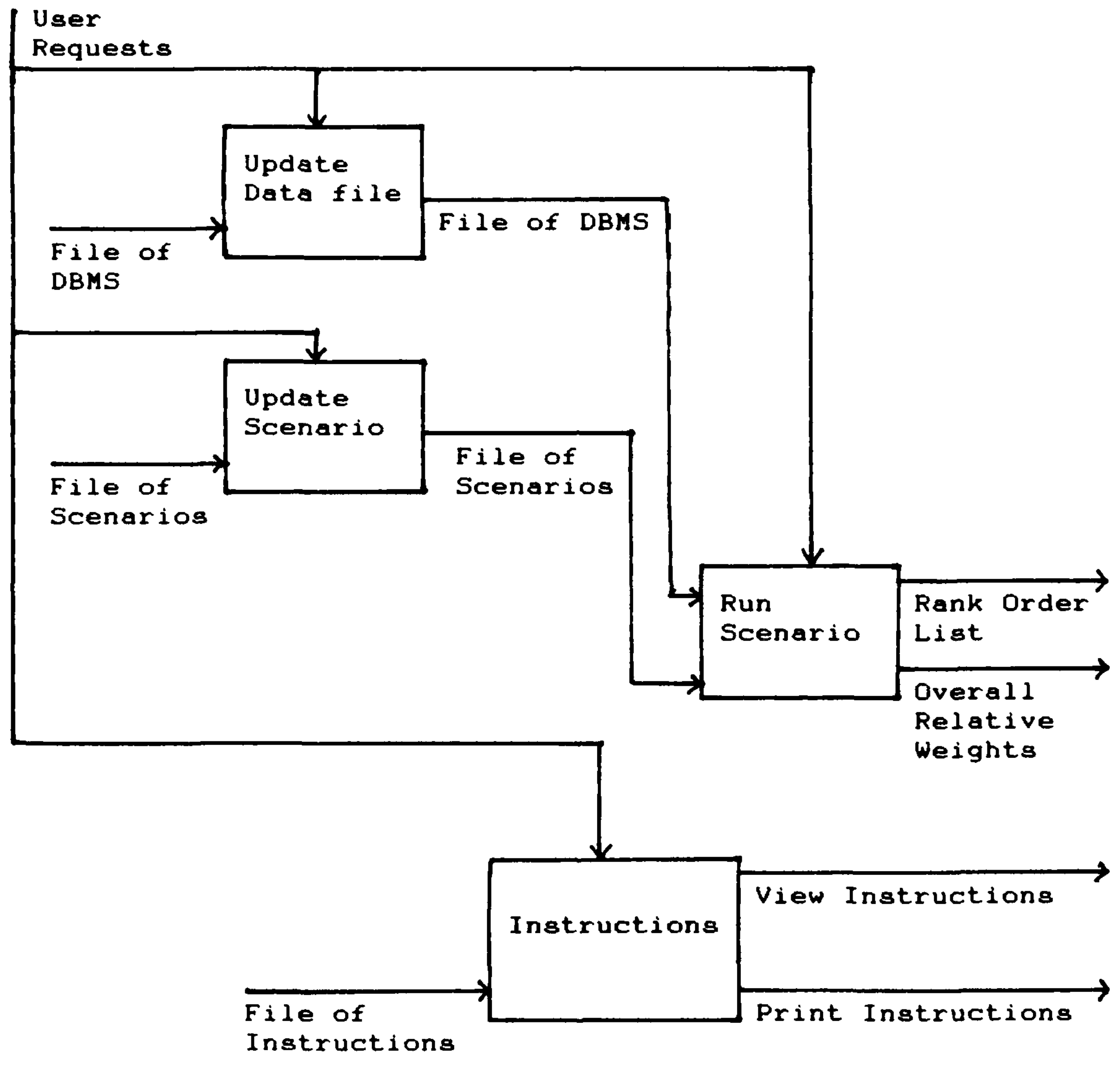
The DSSDS software is a modification and enhancement of a program written by Saaty (2:252) which establishes a fixed hierarchy for DBMS evaluation. It is written in Basic with the intention that it be transportable and available to the widest possible range of users. Although the prototype was originally written in Z-basic on a Z-100 micro-computer, the DSSDS can be used on any PC compatible system, and the methodology could have been implemented using any language.

Software Engineering Concerns

The DSSDS is a menu driven system with no more than five options available on any menu. This was done to increase readability, and prevent the screens from becoming too cluttered. In addition, Saaty's convention of prompting a user to verify inputs was also used. This allows the user to verify, and if necessary change, a series of inputs before the program uses those inputs in any calculations. During the design of the DSSDS, the software engineering concepts of top down structuring, modularity, readability, and maintainability were enforced at all times, and time space tradeoffs were examined to optimize the system execution. Figure 4-1 is the high level SADT diagram for the DSSDS.

FIGURE 4-1

LEVEL 1 SADT FOR DSSDS



Implementation

The implementation of the DSSDS segmented the process into three distinct parts. The first part is the maintenance of a file of information on each DBMS which is available for evaluation. The second part is the maintenance of a file of scenarios which the user has created. The third and final part is the evaluation algorithm which runs a user's scenario and the chosen DBMS alternatives through the AHP algorithm to produce a rank ordered listing of the DBMS.

File of DBMS

In maintaining a file of information on each DBMS, the temptation is to refer to it as a database of databases, and in most decision support systems this information would be stored in a database. However, it seemed illogical that a system designed to evaluate DBMS should require a DBMS to operate. For this reason, the information about each DBMS is stored in a sequential file. The user is allowed to perform three operations on this file. The operations are addition of another DBMS, deletion of a DBMS, and modification of the information on an existing DBMS. The information contained is the evaluation of each DBMS against all of the attributes which appear at the next to the lowest level of the hierarchy. It should be noted that this data represents attribute values and therefore can be normalized to attribute relative weights. Again, nonlinear utility functions could be used to provide a better approximation of the data to the real world.

File of Scenarios

The second part of the implementation is the file of scenarios. Scenarios are the collection of pairwise comparisons that represent user defined environments in which a DBMS will operate. Maintaining a file of scenarios provides many advantages. It allows the user to create the scenario during more than one work session, and also allows for saving modified versions of a scenario. This greatly enhances the user's ability to do "what if" analysis, and study the impact of changes to a scenario. Scenarios can also be added, deleted or modified.

Scenario Execution

The third part of the DSSDS is the numerical evaluation which includes calculation of the relative weights and the synthesis. It takes as inputs a scenario and a list of candidate DBMS. The scenario is in the form of a series of matrices of pairwise comparisons, with one matrix for each attribute that can be decomposed into lower level attributes. These comparisons are converted to relative weights as discussed earlier. The candidate DBMS are in the form of a matrix of actual weights and are also converted to relative weights. Once all of the relative weights are calculated, synthesis is performed to determine the rank ordering of the DBMS.

Synthesis involves calculating the overall relative weight of each DBMS by multiplying the relative weight of each DBMS for a given attribute by the relative weight of that attribute, and then summing the attributes on a common level of the hierarchy. This continues all the way up the hierarchy, until the ultimate objective is reached and the overall relative weights are determined. These results are then sorted to produce a rank ordered list of the DBMS alternatives.

Consistency Checking

At this point it would appear that the process is complete and the final output is available, but there is one step which has been conveniently omitted. That step is the consistency check. In a perfect world, it would be desirable to perform the consistency check during the creation of the scenario and allow the user to correct the inconsistencies immediately. However, the consistency check can not be done until the relative weights have been calculated.

There are several solutions to this problem. The first solution is to ignore the consistency checks during scenario creation and perform them during the final numerical evaluation. The disadvantage of this solution is that it allows the user to get to the end of the process before being informed that the initial inputs were inconsistent. Since this solution is a disaster from a software engineering viewpoint and a potential source of frustration and dissatisfaction for the user, this solution was not implemented.

Another possible solution is to move the calculation of relative weights up to the scenario creation, perform the consistency checks, and then save the relative weights instead of the comparisons. While this solution eliminates the disadvantage of the prior solution, it also creates a new problem. If the relative weights are saved instead of the comparisons, then the user will be unable to modify a scenario once it has been created. This is because relative weights can not be converted back to the original paired comparison values. The solution is to save both the relative weights and the paired comparisons. In this way, all information can be reconstructed and the consistency checks are available during scenario creation.

The other possible solution that will be discussed also involves calculating the relative weights and performing the consistency checks during scenario creation. However, only the pairwise comparisons will be saved. The relative weights will then be recalculated during the numerical evaluation, prior to the synthesis.

The differences between the last two solutions provides a classic space versus time tradeoff. Should the additional space be used to store the comparisons and the relative weights, or should the additional time be spent to calculate the relative weights twice. It was concluded that recalculation of relative weights was the preferred solution due to its straightforward implementation which allowed reusability of code and reduced data storage requirements.

Level of Comparisons Required

During the implementation and initial testing of the DSSDS, there was one potential drawback of the system that stood out. A system with 55 attributes requiring 139 paired comparisons could easily allow a user to be overcome by the abundance of inputs required. The easy solution was to assign initial values of one to all of the paired comparisons, and then allow the user to perform paired comparisons only when he wanted to change the initial values.

While this solution does alleviate the problem of requiring too many inputs, it does little to assist the user in solving the original problem of selecting the best DBMS for the user's requirements. The problem at this level appeared to be twofold: 1) which pairwise comparisons does the user need to change, and 2) should this be left to the user's discretion or was there some way the DSSDS could assist the user in deciding which comparisons to change. The answers to these questions was shown to be dependent upon the hierarchy used, and the numbers chosen were only valid for the DSSDS hierarchy.

Definition of Terms

The first step was to define what is meant by attributes with maximum, equal and minimum weights. The maximum weight that an attribute can attain is produced when it has absolute dominance over all other attributes it is compared against, and those attributes are all equal with respect to each other.

Attributes in a hierarchy are made equal by placing a 1 in the input matrix. If an input matrix contained only ones, then the resulting relative weights are the equivalent of taking the reciprocal of the number of attributes in the matrix. These attributes will be referred to as having equal weight. In other words, when it is stated that attributes have equal weight, it means that all of the attributes being considered have the same weight and not that two of the attributes have equal weight.

The minimum weight an attribute can attain is produced when attribute A1 has absolute importance over all other attributes it is compared against, attribute A2 has absolute importance over all other attributes it is compared against except A1, and so on until the last attribute which all attributes have absolute importance over. This final attribute will have the minimum possible weight at that level of the hierarchy. The CI for this type of hierarchy will be 0.28 or higher and should never be used in an actual hierarchy; however, the results will be used here as worst case estimates for the minimum weight.

The input matrices for maximum, equal and minimum weights are shown in Tables 4-1, 4-2, and 4-3, while the actual minimum, equal and maximum weights for n=2 through n=9 are shown in Table 4-4.

TABLE 4-1
INPUT MATRIX FOR MAXIMUM WEIGHTS

	A	B	C
A	1	9	9
B	1/9	1	1
C	1/9	1	1

TABLE 4-2
INPUT MATRIX FOR EQUAL WEIGHTS

	A	B	C
A	1	1	1
B	1	1	1
C	1	1	1

TABLE 4-3
INPUT MATRIX FOR MINUMUM WEIGHTS

$$\begin{array}{c|ccc} & A & B & C \\ \hline A & 1 & 9 & 9 \\ B & 1/9 & 1 & 9 \\ C & 1/9 & 1 & 1/9 \end{array}$$

TABLE 4-4
MINIMUM, EQUAL, AND MAXIMUM WEIGHTS

# of attributes	2	3	4	5	6	7	8	9
Minimum Weight	.100	.041	.025	.017	.014	.011	.009	.008
Equal Weight	.500	.333	.250	.200	.167	.143	.125	.111
Maximum Weight	.900	.818	.750	.692	.643	.600	.563	.529

Hierarchy Dependence

Assume attribute B1 represents .10 of the overall weight. If attribute B1 is comprised of attributes C1, C2, and C3, then the initial equal weights would give all three attributes .033 (.333*.10) as an overall relative weight. The minimum weight any of these attributes could attain is .004 (.041*.10), while .082 (.818*.10) is the maximum overall weight attainable. Is the difference between .082 and .033, or .033 and .004 significant enough that the user should perform the pairwise comparisons on C1, C2, and C3. The answer to this question is hierarchy dependent.

Consider the hierarchies of Figure 4-2 and Figure 4-3. If attribute B1 constitutes .10 of the overall weight, then it is relatively important with respect to the other attributes in Figure 4-2, while attribute B1 in Figure 4-3 is relatively unimportant with respect to attribute B2. Based on this, the solution that has been implemented in the DSSDS is dependent on the hierarchy of Figure 3-1, and should not be considered as the solution for all hierarchies.

Implementation of Levels of Comparisons

The DSSDS implementation is straightforward. During scenario creation, if the difference between the minimum and equal weights for an attribute with respect to all attributes at the next higher level exceeds .020, or the difference between the maximum and equal weights exceeds .040, then the DSSDS recommends to the user that pairwise comparisons should be continued for the next level of attributes.

There are four conditions that the DSSDS will recognize, and indicate to the user. The four messages that can be sent to the screen are:

1) The user should continue pairwise comparisons,
2) The user should continue pairwise comparisons if one attribute is relatively unimportant,
3) The user should continue pairwise comparisons if one attribute is absolutely important,
4) The user should not perform any more comparisons.

FIGURE 4-2

MAXIMAL HIERARCHY

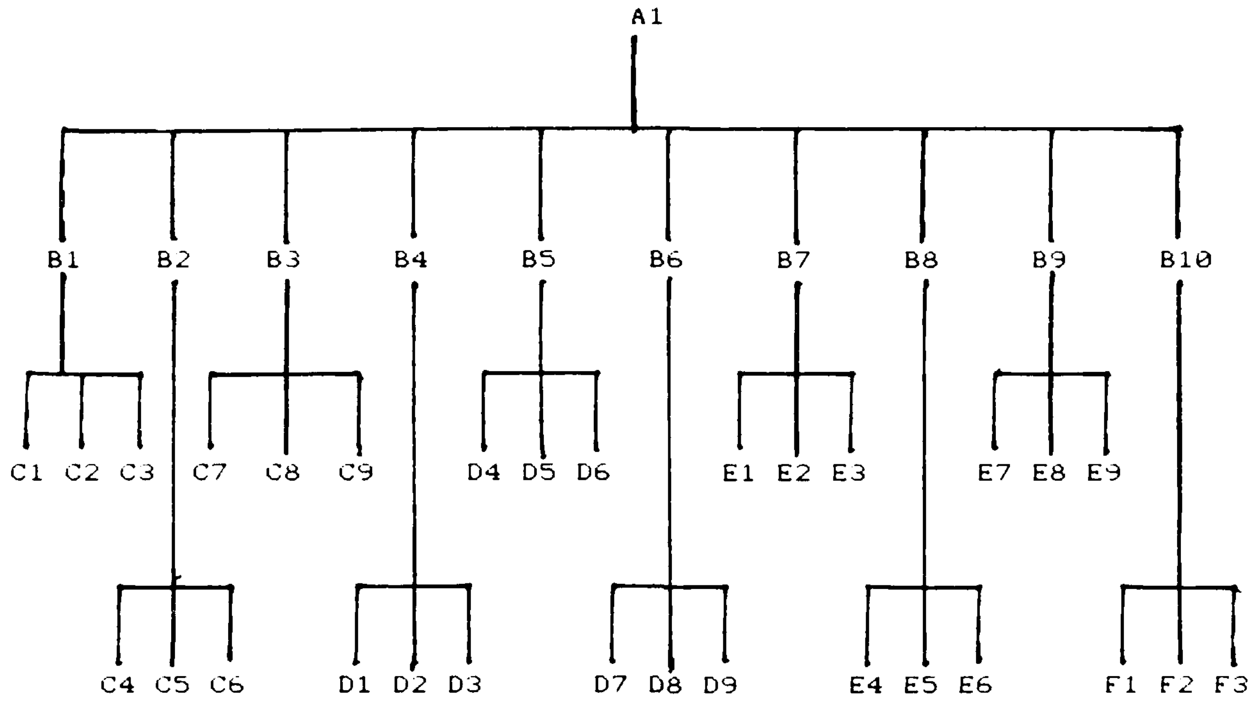
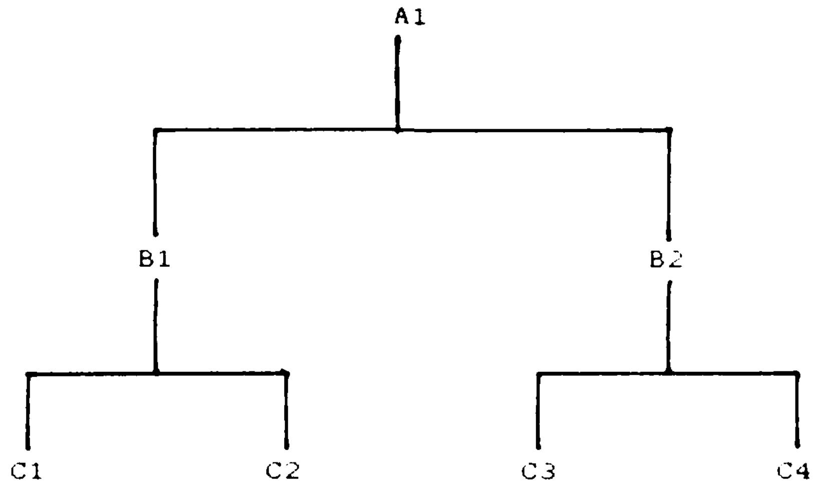


FIGURE 4-3

MINIMAL HIERARCHY



If the overall relative weight (ORW) of an attribute is greater than .10, then message 1 is displayed. If ORW is not greater than .10, then the maximum, equal, and minimal weights are calculated. If the difference between the minimum and equal weights is greater than .02 message 2 is displayed. If the difference between the maximum and equal weights is greater than .04 message 3 is displayed. If neither is true, then message 4 is displayed. It is important to remember that these are merely messages to aid the user, and do not effect system execution. It remains the user's responsibility to elect to change the pairwise comparisons.

Example Scenario

Using the data from the pilot study (Appendix A) and the hierarchy of Figure 3-1, here is an example which contains all four cases. The pilot study shows cost, performance, and hardware requirements have relative weights of .265, .565, and .170 respectively. Since all three attributes have weights greater than .10, message 1 would be displayed for all three attributes. The cost and hardware requirements attributes only have one more level, so on those two branches of the hierarchy all pairwise comparisons would be performed.

This leaves performance as the only remaining branch of the hierarchy. Table 4-5 provides an input matrix of pairwise comparisons for performance.

TABLE 4-5

EXAMPLE INPUT MATRIX FOR PERFORMANCE

	Error Handling	Ease of Learning	Ease of Use	Adaptability
Error Handling	1	2	1/4	1/2
Ease of Learning	1/2	1	1/7	1/3
Ease of Use	4	7	1	3
Adaptability	2	3	1/3	1

The relative weights are .133, .074, .569, and .224 respectively. Since performance had an ORW of .565, this leads to ORW of .075, .042, .323, and .127 respectively. This completes level 2 of the hierarchy.

Next, the attributes of level 3 will be examined one at a time. Error handling has an ORW of .075 which is less than .10, so message 1 will not be displayed. Since error handling is composed of two attributes, .038 (.075/2) is the equal weight for error messages and documentation. The minimum attainable ORW is .008 (.075*.100), while .068 (.075*.900) is the maximum attainable ORW. At this point message 2 would be displayed instructing the user to continue if one of the attributes is insignificant.

Since ease of learning is composed of four attributes,

.011 (.042/4) is the equal weight for all four attributes. The minimum attainable ORW is .001 (.042*.025), while .031 (.042*.750) is the maximum attainable ORW. At this point message 4 would be displayed because the differences are inside the allowable range.

Since ease of use and adaptability have ORW of .322 and .127, both attributes would print message 1 recommending that the user continue performing comparisons. This completes level 3 of the hierarchy. Pairwise comparisons will now be performed on the attributes at level 4 that the user decided to change.

The user has decided to perform comparisons on the attributes that comprise ease of use and adaptability, but not on the attributes that comprise error handling and ease of learning. The input matrix of pairwise comparisons for ease of use and adaptability are in Tables 4-6 and 4-7.

TABLE 4-6

EASE OF USE

	Data Manipulation	QP/AP	Report Generation
Data Manipulation	1	1/3	1/5
QP/AP	3	1	1/2
Report Generation	5	2	1

TABLE 4-7

ADAPTABILITY

	Report Generation	Data Constraints
Report Generation	1	1/3
Data Constraints	3	1

Using Table 4-6, the weights for data manipulation, QP/AP, and report generation with respect to ease of use are .109, .309, and .582 respectively. This yields .035, .099, and .187 as the relative weights. Using Table 4-7, the weights for report generation and data constraints with respect to adaptability are .250, and .750 respectively. This yields .032, and .095 as the relative weights.
All of the attributes at level 4 will now be examined one at a time.

Documentation is part of error handling and ease of learning. Since pairwise comparisons were not performed on either of those attributes, documentation has an ORW of .049 (.038+.011) which is less than .10 so message 1 will not be displayed. Since documentation is composed of four attributes, .012 (.049/4) is the equal weight. The minimum attainable weight is .001 (.049*.025), while .037 (.049*.750) is the maximum attainable weight. At this point message 4 would be displayed because the differences are inside the allowable range.

Data manipulation is part of ease of learning and ease of use, and has an ORW of .046 (.035+.011) which is less than .10, so message 1 will not be displayed. Since data manipulation is composed of nine attributes, .005 (.046/9) is the equal weight. The minimum attainable ORW is .0004 (.046*.008), while .024 (.046*.529) is the maximum attainable ORW. At this point message 4 would be displayed because the differences are inside the allowable range.

QP/AP is part of ease of learning and ease of use, and has an ORW of .110 which is greater than .10, so message 1 will be displayed recommending continuation of comparisons.

Report generation is part of ease of learning, ease of use and adaptability, and has an ORW of .230 (.187+.011+.032), so message 1 would be displayed to recommend performing more comparisons.

Data constraints is part of adaptability and has an ORW of .095 which is less than .10, so message 1 will not be displayed. Since data constraints is composed of seven attributes, .014 is the equal weight. The minimum attainable ORW is .001, while .057 is the maximum attainable ORW. At this point message 3 would be displayed instructing the user to continue performing comparisons if one of the attributes is dominant.

Table 4-8 is a summary of the ORW for the attributes in this example. Where an attribute was on the lowest level of the hierarchy, input matrices were not constructed and the weights are not listed.

TABLE 4-8
ORW FOR ATTRIBUTES

Attribute	Upper level attribute it is part of	Weight
Cost	Purchase of a DBMS	.265
Performance	Purchase of a DBMS	.565
H/W Rqmnts	Purchase of a DBMS	.170
Error Handling	Performance	.075
Ease of Learning	Performance	.042
Ease of Use	Performance	.323
Adaptability	Performance	.127
Error Messages	Error Handling	.038
Documentation	Error Handling	.038
Documentation	Ease of Learning	.011
Data Manipulation	Ease of Learning	.011
QP/AP	Ease of Learning	.011
Report Generation	Ease of Learning	.011
Data Manipulation	Ease of Use	.035
QP/AP	Ease of Use	.099
Report Generation	Ease of Use	.187
Report Generation	Adaptability	.032
Data Constraints	Adaptability	.095

Evaluation

A preliminary evaluation of the DSSDS software, using the evaluation form in Appendix C, showed two significant trends. Those users who had a working knowledge of databases and database terminology were able to concentrate on the comparisons and hurdle rates, and they produced meaningful scenarios and rarely struggled with inconsistent inputs. On the other hand, user's who were not comfortable with database terminology tended to get so confused by the different comparisons required that they rarely input a consistent matrix on the first try. This led to frustration with the system and eventual dissatisfaction with the results.

This does not imply that the system is only useful to database experts. It merely demonstrates that it is in fact a decision support system and not an expert system. Those users who do not have a working knowledge of databases need to review the glossary of terms (Appendix B) and user's manual (Available with the software) in depth before attempting to operate the system.

This completes the description of the software that was developed to implement the DSSDS and the preliminary evaluation. The next chapter will explain the conclusions that have been drawn, and the recommendations for improvements in the system.

V. Conclusions and Recommendations

The DSSDS has been designed to assist users in the purchase of a DBMS. It has combined a six level hierarchy of DBMS attributes with a file of information on numerous DBMS, to allow the user to select the DBMS that best meets his requirements.

Nonlinear Utility Functions

The implementation of the DSSDS has used only linear utility functions. As was shown in Figure 1-1 and explained in Chapter 1, clearly there are cases where the evaluation of a DBMS attribute does not fit a linear scale. The problem is twofold: 1) Can the appropriate nonlinear utility function for an attribute be determined a priori, or does it depend on the user's application, and 2) how would the utility functions be implemented into the DSSDS.

The initial answer to question 1 is that the utility function could probably be determined, but the critical values are application dependent. Therefore, if the utility functions are determined, it would be incumbent upon the user to evaluate his application and determine the critical values. However, it has already been shown in the discussion of Levels of Detail in Chapter 4 that the number of inputs required by the system are already reaching an excessive amount. One very challenging area of future research would be to devise a method for assisting the user in determining critical values, and to implement it such that the number of user inputs is minimized.

The answer to question 2 is more concrete than the answer to question 1. If the user was to input the critical values for an attribute, these critical values would be different for each scenario. Therefore, it would make sense to place the selection of nonlinear utility functions at the end of scenario creation. The user would create the scenario based on the importance of the attributes and the nonlinear utility functions. This is favorable for several reasons, but is most appealing for use in nonlinear utility function research. Since the user is allowed to do "what if" analysis on scenarios, it will allow for increased analysis and experimentation with the utility functions.

Group Decision Making

The current implementation allows a single user to evaluate a DBMS with respect to his requirements. The only way for a group to use the current design would be for each member of the group to run a scenario and then compare the final ratings that each one produced. It would be extremely beneficial to allow each user to create a scenario, and then have the scenarios merged together before the DBMS are evaluated. This would produce a rank ordered list that reflected the requirements of the group rather than having multiple lists of individual objectives.

There are numerous possibilities for how group inputs could be implemented. Different variations would have to include allowing for any number of members in a group, and the possiblity of weighting group members.

Mainframe DBMS Systems

The current file of DBMS contains data on numerous micro-computer based DBMS and the hierarchy is definitely slanted toward these types of databases. Further research into the different requirements for mainframe DBMS systems would also be useful.

Evaluation Conclusions

In addition to the evaluation trends discussed in Chapter 4, there were numerous suggestions from the evaluators on ways to improve the software. The majority of the suggestions were requesting more background information for the novice user such as detailed definitions and an explanation of the scales used to rate the database attributes. This was most profound when users tried to include hurdle rates and were not sure what the values should be.

APPENDIX A

Pilot Study

A pilot study was conducted to show that users need assistance in determining the relative importance of attributes. The intent was to show that while user's know what they want, they have trouble converting that preference into numbers.

The study asked users to evaluate their preferences based on level 2 of the DSSDS hierarchy of Figure 3-2. The users were told that they had been selected to purchase a DBMS to maintain a student/faculty database. The database was to contain information relative to schedules, grades, advisors, thesis informatio, and any other applicable data.

The user was instructed to perform the evaluation in two steps. Step 1 was to evaluate the attributes of cost, performance, and hardware requirements against the overall objective of purchasing a DBMS. The user could divide a total of 100 points among the three attributes. Step 2 was a pairwise evaluation of the attributes using the scale of Table 2-1.

The participants in the study included four advanced database students, three beginning database students, two instructors, and one electrical engineer. The inputs of the ten user's are shown in Tables A-1, A-2 and A-3.

TABLE A-1

DIRECT VS PAIRWISE ASSIGNMENT (COST)

	DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	DELTA
	40	15.8	24.2
	40	27.9	12.1
	40	48.7	- 8.7
	30	21.8	8.2
	30	14.3	15.7
	20	20.7	- 0.7
	20	11.7	8.3
	20	11.1	8.9
	20	5.8	14.2
	5	6.2	- 1.2
AVE	26.5	18.4	8.1

TABLE A-2

DIRECT VS PAIRWISE ASSIGNMENT (PERFORMANCE)

	DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	DELTA
	80	80.8	- 0.8
	70	73.5	- 3.5
	65	76.6	-11.6
	60	77.8	-17.8
	60	71.4	-11.4
	50	76.6	-26.6
	50	68.3	-18.3
	50	64.9	-14.9
	40	57.1	-17.1
	40	43.5	- 3.5
AVE	56.5	69.05	-12.55

TABLE A-3

DIRECT VS PAIRWISE ASSIGNMENT (HARDWARE REQUIREMENTS)

	DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	DELTA
	30	28.6	1.4
	30	20.0	10.0
	20	11.1	8.9
	20	7.8	8.9
	15	13.0	2.0
	15	7.6	7.4
	10	7.6	2.4
	10	7.2	2.8
	10	6.7	3.3
	10	5.8	4.2
AVE	17.0	11.54	5.46

Using the data in Tables A-1, A-2 and A-3, a paired t test was performed to determine whether the means of the two methods were equal. The first step was to compute the difference between the means (x), and the standard deviations (sigma). Table A-4 provides these values.

TABLE A-4

MEANS AND STANDARD DEVIATIONS

	Difference Between Means	Standard Deviation
COST	8.1	9.53
PERFORMANCE	-12.55	7.67
HARDWARE REQUIREMENTS	5.46	3.84

These values were then used to calculate the t statistic using t=x/(sigma/n**(1/2)) . The t values and the average weights are shown in Table A-5.

TABLE A-5

COMPARISON OF RELATIVE WEIGHTS

	DIRECT ASSIGNMENT	PAIRWISE ASSIGNMENT	t VALUE
COST	26.5	18.4	2.687
PERFORMANCE	56.5	69.1	-4.906
HARDWARE REQUIREMENTS	17.0	11.5	4.490

Using a table of t values for a smple size of ten, a hypothesis can be rejected at the 0.95 level if the t value calculated is greater than 2.262 or less than -2.262, and all of the values in Table A-5 are outside that range. Therefore, the hypothesis that the means are equal can be rejected, implying that the DSSDS is needed to assist the user in weighting his requirements.

In fact, the values for all ten user's could have been rejected at the 0.975 level, but there were some subsets of user's that could not reject at that higher level. The 0.95 confidence level was chosen for that reason.

APPENDIX B

Glossary of Terms in the Hierarchy

Accuracy -- decimal values and maximum digits per numeric field.

Adaptability -- usability of the DBMS for storing and outputting the user's specific data requirements.

Backup and recovery -- availability of a complete backup, audit files, or log files, and rollback capabilities.

Bulk load -- ability to load data files into the database.

Classes -- frequency and level of classes which are offered.

Color -- how effectively color is used to enhance the operation of the DBMS.

Cost -- the initial purchase price of the DBMS and any additional costs the user might incur.

Database description methods -- description of fields, format of data in fields, and relationships between fields.

Data constraints -- any restrictions on how data can be entered into the system, and the types of data that the system will accept.

Data entry -- ease with which additions, deletions, and updates can be accomplished.

Data manipulation -- how the user can get data into the DBMS, the speed at which this can be performed, and the ability of the DBMS to adjust changes in both content and format.

Data types -- availability of alphnumerics, integer, fixed point, floating point, date, time, boolean, etc.

Default values -- automatic input of constant values during initialization.

Documentation -- quantity and quality of printed material, and other learning aids.

Ease of learning -- how much introductory support is provided.

Ease of use -- approximation of the skill level required to effectively operate the DBMS.

Error handling -- capability to recover from errors.

Error messages -- quantity and quality of error messages.

Flexibility -- ability of an application program to reference arbitrarily located units of data independent of placement within the database.

Formats -- ability to save report formats for further use.

Format modifications -- ability to convert between data types.

Growth -- ability to modify and extend the database without requiring modifications to existing software.

Hardware requirements -- type of hardware the DBMS requires and any special hardware features it can utilize.

Hard disk -- ability to run the DBMS from a hard disk.

Headers -- ability to print report and page headers.

Help line -- availability and cost of a telephone help line.

Independence -- amount of difference permitted between logical and phvsical data definitions.

Initial cost -- purchase price of the DBMS.

Integrity -- protecting data from invalid updates by authorize users. Protection against loss of data or erroreous updates caused by system failures. This also includes the ability to validate fields.

Justifying fields -- right justifying, left justifying, or centering fields.

Labels -- ability to print mailing labels and maximum number of labels across a page.

Languages -- capabilities of the DBMS specification language.

Machine -- capability of the DBMS to run on a specific computer.

Maintenance cost -- any fees which must be paid for supplier support.

Materials -- quantity and quality of printed materials that are supplied with the DBMS.

Memory -- minimum amount of memory required to operate the DBMS.

Mouse -- ability of the DBMS to use a mouse.

Operations -- manipulation and retrieval operations, including specification statements and vendor supplied macro routines.

Paging -- controlled page breaks, printing page numbers, and printing a specified number of records per page.

Performance -- how the DBMS interfaces with the user after it starts to execute, including how easy it is to learn and use, and how easily it is adapted to the user's data.

Processing programming (query processing and applications programming or QP AP) -- how the user can interface with the DBMS through other software, either written by the user or supplied by the vendor.

Redundancy -- Amount of duplication required to support multiple users.

Report generation -- capability of providing printed output in a format that is convenient for the user.

Searches -- ability to search based on logical conditions, values, and non key fields.

Security -- protecting data from overt or inadvertent change by unauthorized users. Ability to password protect entries.

Size -- Maximum characters per record, fields per record, records per table, and tables per database.

Skill level -- amount of programming experience required to effectively use the embedded language.

Sorting -- Maximum sorting levels, maximum characters per sort key, and ascending and descending sorts.

Specialized fields -- Monetary fields, system date and time, and automatically calculated fields.

Speed -- how long it takes to make updates, run programs, or execute queries.

Tutorials -- availability of on-line tutorials or help functions.

Upgrade cost -- how much the supplier charges to provide new version releases.

User Defined screens -- ability to custom design screens for data entry.

Width -- maximum report width.

APPENDIX C

DSSDS Evaluation

This evaluation is designed to rate the effectiveness of the DSSDS. Evaluators should complete this questionaire only after they have finished entering and running at least one scenario. Evaluations can range from a value of 1 if the DSSDS was completely unsatisfactory in that area, to a value of 5 if the DSSDS completely satisfied a criteria. Values 2, 3 and 4 should be used for partial satisfaction. Please circle the number which best describes your satisfaction with that criteria.

	Unsatisfactory		Average		Satisfactory
1. Were the menus clear?	1	2	3	4	5
2. Were the DSSDS prompts helpful?	1	2	3	4	5
3. Was the system forgiving when errors occured?	1	2	3	4	5
4. Did the option to only rate portions of the hierarchy help?	1	2	3	4	5
5. Format of the final results?	1	2	3	4	5
6. Overall reaction to the final results the DSSDS produced?	1	2	3	4	5
7. Did you understand what the system was trying to accomplish?	1	2	3	4	5
8. Overall reaction to the system?	1	2	3	4	5

9. Amount of time it took for you to run one complete scenario. ____________________ (Hours:Minutes)

BIBLIOGRAPHY

1. Bowerman, Robert. "Relational Database Systems For Micros," *Datamation*: 128-134 (September 1983).

2. Cohen, Leo J. *Database Management Systems*. Wellesley Ma Q.E.D. Information Sciences, Inc., 1978.

3. Curtice, Robert M., and Paul E. Jones Jr. "Database The Bedrock of Business," *Datamation*. 163-166 (June 1984).

4. Hartrum, Thomas C. *AFIT/ENG Software Development Documentation Guidelines and Standards*. Unpublished text. AFIT, Wright-Patterson AFB, Ohio, 1986.

5. Johnson, Charles R. "Constructive Critique of a Hierarchical Prioritization Scheme Employing Paired Comparisons," *Proceedings of the International Conference of Cybernetics and Society of the IEEE*. 373-378. Institute of Electrical Engineers, Cambridge, MA, 1980.

6. Jones, Marilyn S., Charles J. Malmborg, and Marvin Agee. "Decision Support System Used for Robot Selection," *IE* 66-73 (September 1985).

7. Katz, Randy H. "Managing the Chip Design Database," *IEEE*: 26-35 (December 1983).

8. Keeney, Ralph L. and Howard Raiffa. *Decisions with Multiple Objectives*. New York: John Wiley & Sons, 1976.

9. Nierenberg, Nicolas. "Pitfalls Can Plague Unwary Benchmarking Teams," *Government Computer News*: 34-35 (February 1986).

10. Palmer, Ian. *Database Systems: A Practical Reference*. Wellesley Ma Q.E.D. Information Sciences, Inc., 1975.

11. Saaty, Thomas L. "Axiomatic Foundation of the Analytical Hierarchy Process," *Management Science: Journal of the Institute of Management Sciences*, *32 (7)*: 841-855 (July 1986).

12. Saaty, Thomas L. *Decision Making for Leaders: The Analytical Hierarchy Process for Decisions in a Complex World*. Belmont CA: Lifetime Learning Publications, 1982.

13. Software Digest. Ratings Newsletter. Philadelphia: Software Digest, Inc., 1986.

14. Sussman, Philip. "Evaluating Decision Support Software," Datamation: 171-172 (October 1984).

15. Ullman, Jeffrey D. Principles of Database Systems. Rockville MD: Computer Science Press, 1982.

16. Zahedi, Fatemah. "The Analytical Hierarchy Process: A Survey of the Method and its Applications," Interfaces, 16 (4): 96-108 (July-August 1986).

17. Zahedi, Fatemah. "Database Management System Evaluation and Selection Decision," Decision Sciences, 16 (1): 91-116 (Winter 1985).

VITA

Captain Dennis R. Davidson was born on 25 September 1960 in Kittery, Maine. He graduated from high school in Rochester, New Hampshire, in 1978 and attended Norwich University from which he recieved the degree of Bachelor of Science in Mathematics in May 1982. Upon graduation, he received a commission in the USAF through the ROTC program. He served as the Space Shuttle Data Officer at the Johnson Space Center, Houston, Texas, until entering the School of Engineering, Air Force Institute of Technology, in June 1985.

Permanent address: 12115 Palmcroft
Houston, Texas 77058

END
5-87
DTIC