SOS

DSS Software Selection: A Multiple Criteria Decision Methodology

Louis A. Le Blanc
School of Public and Environmental Affairs, Indiana University, Bloomington, IN 47405, USA

M. Tawfik Jelassi
Technology Management Area, INSEAD, 77305 Fontainebleau, France

This paper illustrates an evaluation methodology for DSS software selection. The methodology incorporates three stages: (1) DSS software screening; (2) generator evaluation; and, (3) specific DSS design. Initially, developing a short list through screening of DSS software determines whether an appropriate package exists and narrows the field of available generator products for detailed consideration. The second stage determines which of the remaining generators (the finalists) best meets the needs of the organization, from both functional and technical perspectives. The final stage compares user requirements with the features of the selected DSS software by defining how these requirements will be satisfied by specific DSS applications built using the DSS generator. The methodology also controls for the possibility that no generator product is suitable and that specific DSS must be constructed from DSS tools. No other reported evaluation and selection approach offers this device. A case example demonstrating the applicability of the suggested methodology is given, and the impact of DSS software on the development of specific DSS is discussed.

Keywords: Software evaluation and selection, Decision support systems, Specific DSS design, DSS generators, DSS tools.

North-Holland
Information & Management 17 (1989) 49–65

1. Introduction

1.1. The DSS Software Selection Problem

Recent publications devoted to evaluating decision support systems (DSS) software [19,21,22,28,

Louis A. Le Blanc is Associate Professor of Information Systems in the School of Public and Environmental Affairs at Indiana University, where he is affiliated with its Transportation Research Center. Professor Le Blanc earned his Ph.D. from Texas A&M University in 1978. Dr. Le Blanc completed postdoctoral study in management information systems at the University of Minnesota's Graduate School of Management in 1984 and at Indiana University's Graduate School of Business in 1987. In 1985, Dr. Le Blanc served as a systems consultant to AT&T Technologies. He was selected as a Faculty Resident with the Management Information Consulting Division of Arthur Andersen & Co. in 1986. Professor Le Blanc has published in the areas of software evaluation and the impact of information technology on maritime safety. Current research programs focus on the selection of expert system shells as well as the evaluation of group decision support systems performance. Recent teaching assignments include systems analysis and design, decision support systems, and telecommunications.

M. Tawfik Jelassi is Associate Professor of Information Systems at the European Institute of Business Administration (INSEAD). Prior to that, he was on the faculty of the Indiana University School of Business. Dr. Jelassi received a Ph.D. in Computer Applications and Information Systems from New York University, and Diplomas in Computer Science and Business Administration from the Université de Paris-Dauphine and the Université de Tunis. His research interests include Group Decision Support Systems, Computer-Assisted Negotiation, Multiple Criteria Decision Making, and Database Applications. He has recently contributed articles to the *European Journal of Operational Research, Journal of Management Information Systems, Decision Support Systems: The International Journal, Information and Management*, and the European journal *Interfaces*. Professor Jelassi also serves as a consultant to business and government in several countries.

and 32] have identified criteria, especially user-related ones, which are critical in selecting a suitable DSS generator. However, these authors have not suggested how to incorporate multiple user criteria, as well as technical attributes, into a complete and thorough evaluation and selection process. Furthermore, Lynch [16 and 17] suggests that inadequate examination of prospective software packages leads to serious difficulties if not failures when implementing information systems.

Although a number of approaches to selecting application software for transaction processing and MIS have been proposed [3,4,6,9,18,25,33], some critical factors were omitted. These factors include assuring that the selected software package is superior to a custom alternative, or that a screening process is provided to reduce the number of packages subjected to detailed evaluation.

1.2. DSS Terminology

A number of key terms and expressions that will be used throughout the paper are now defined. A *DSS Generator* is a "package of related hardware and software which provides a set of capabilities to build specific DSS quickly and easily" [26]. Examples of such generators include IFPS [8], Prefcalc [15], Expert Choice [7], and Lightyear [29]. A DSS generator constitutes one of the three technological levels that make up the DSS development framework suggested by Sprague [26]. The other two levels are: *Specific DSS*, which are systems that actually support the manager (user) to solve specific sets of related decision problems; and *DSS Tools*, which are hardware and software elements built by a toolsmith to facilitate the development of both specific DSS and DSS generators (see respectively (1) and (2) in *Fig. 1*). Examples of such DSS tools include procedural programming languages, graphics and color subroutines, and other dialog-handling software.

Fig. 1 (adapted from [26]), shows the three technological levels defined above, the relationships between them, and the manager/technician roles associated with each level. Notice that specific DSS can be developed either directly from tools or by adapting the DSS generator to satisfy the application requirements. In the latter case, the DSS builder may use the *iterative design approach* [5] to add capabilities to the ones available in the DSS generator (or delete unnecessary features) as needed by the specific DSS. This approach can be represented by the iterative cycling between the DSS generator and the specific DSS (see (3) in Fig. 1).

To emphasize the importance of evaluation and selection of DSS software (as compared to that of other information systems), it should be noted that DSS generators are used to develop *multiple*

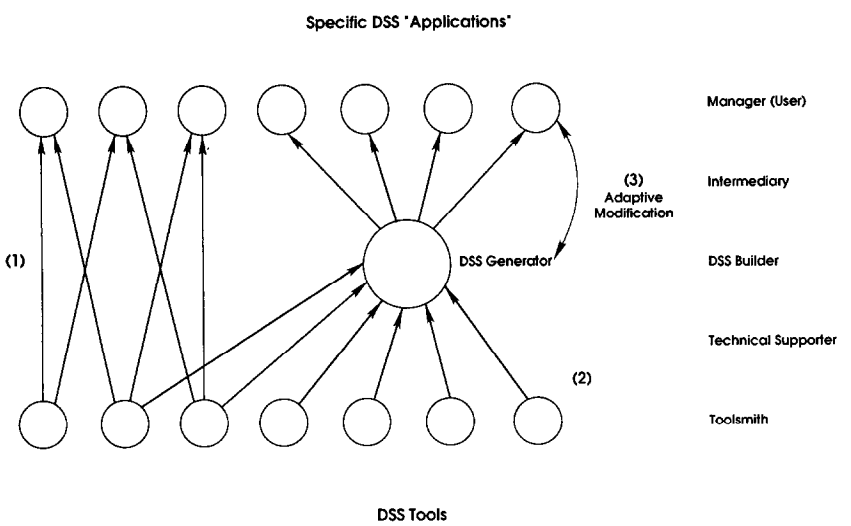

Fig. 1. DSS Technology Levels and Development Framework.

application systems, while MIS software is employed only for a *single* application. To efficiently develop specific DSS using the iterative design approach, a generator needs to be available. Hogue and Watson [10] reported that 50 percent of the firms that they studied developed specific DSS applications with a generator. The very critical software evaluation and selection process for DSS generators should take place prior to any systems analysis and design efforts.

This paper illustrates a method to select the most appropriate DSS generator where multiple criteria exist not only from functional requirements but also from technical and vendor-support perspectives. As an essential part of the methodology, an initial stage determines whether a DSS software product is even suitable for a particular enterprise, or should specific DSS be developed from available tools.

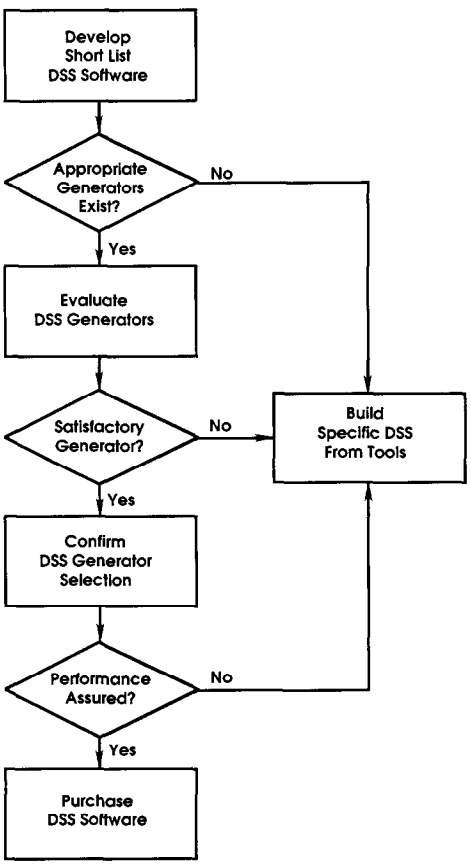

Fig. 2. A Multiple Criteria Decision Methodology for DSS Software Selection.

The proposed selection process in this paper also ensures that, at each successive stage of the methodology, a DSS generator is superior to a DSS application custom-built from tools. It continually reduces the number of DSS software products under consideration until a final selection of a generator is made or constructing a specific DSS from tools is chosen as the best alternative (see *Fig. 2*).

1.3. Structure of the Paper

This paper is primarily addressed to academics interested in software selection methodologies as well as practitioners faced with DSS-related problems. Section 2 outlines DSS developments affecting information systems (IS) planning. In particular, the enterprise software policy and the implications that DSS generators might have are discussed. Section 3 suggests a multiple criteria methodology for DSS software selection. The three stages of this methodology – DSS software screening, DSS generator evaluation, and specific DSS design – are described. Then, Section 4 presents a case example that demonstrates the applicability of the proposed methodology. Section 5 covers the impact of DSS software on specific DSS development from systems analysis and design, installation, and operating support viewpoints. Section 6 concludes the paper with some final remarks.

2. DSS Developments Affecting IS Planning

The increased use and availability of DSS software has greatly influenced IS planning [34]. Questions such as the following ones have been raised. What is the organization's strategy concerning the use of packaged software? What types of criteria should be used? Are packages easy to maintain?

These questions and others must be directly addressed by IS management, since many applications requirements can be effectively satisfied by DSS software packages. One major consideration is the degree to which DSS generators are compatible with the enterprise's technical architecture for information processing. For example, if multiple DSS generators and vendors are used, how effectively can a common database be employed throughout the organization?

2.1. Implications of DSS generators

With the greater availability of DSS generator software and its improved quality, it appears that, for most organizations, DSS packages may be preferred over custom development from DSS tools. While there are obvious benefits to using generator packages, they are not necessarily "off-the-shelf" solutions. Generator selection should be a careful and well-organized process to satisfy user requirements and meet generally-accepted information processing standards for quality and performance.

The selection of a DSS generator needs to be a disciplined process of matching package options with operating procedures, and reconciling any differences. Modifications to the DSS software should be carefully analyzed before they are made to consider risks and jeopardizing longer-term vendor support.

The use of DSS generators may conflict with the major benefit of adopting a corporate database – that is, sharing data among several users while enforcing a unique way to define it and manipulate it in addition to minimizing and controlling data redundancy. Since generator packages often create and manage their own data, this may lead to having multiple versions of the same data and can be a source of inconsistencies.

2.2. Enterprise Software Policy

The importance of an enterprise-wide policy regarding the use of application software cannot be overemphasized. This means a stated "going-in" position concerning the desirability of using DSS generators and the manner in which it should be used. Such a policy statement guides a project team as it considers the compromises that users might have to make to employ DSS technology.

The hardware and software policy of the firm has a direct relationship to the choice of DSS products. Normally, such a policy will have been determined by the enterprise IS strategy. The evaluators of DSS generators will then restrict their search to vendors offering software that will operate in the given technical environment.

The organization needs to determine overall vendor and market criteria for DSS software evaluation. For example, each package must meet 85 percent of the application's functional requirements; and each DSS generator must have been previously installed in at least five organizations.

3. A Decision Methodology for DSS Software Selection

There are three principal stages in the proposed DSS evaluation and selection methodology: (1) screening of prospective candidates and development of a short list of DSS software packages; (2) selecting a DSS generator, if any, which best suits the application requirements; and, (3) matching user requirements to the features of the selected generator and describing how these requirements will be satisfied through the building of prototypes for specific DSS. The detailed procedures involved in each state of the selection process are described in the following sections.

3.1. DSS Software Screening

During this first stage of the evaluation and selection methodology, three key issues must be addressed: (1) Is there DSS software that can be used or should a specific DSS be developed from tools?; (2) What DSS generators are available?; and, (3) Which DSS software packages should be seriously considered and evaluated in detail? (Examples of commercially available mainframe- and

Table 1
Representative DSS Software Products.

Product	Vendor
Mainframe Packages	
EXPRESS	Management Decisions, Inc.
IFPS	Execucom Systems, Corp.
SYSTEM W	Comshare, Inc.
SIMPLAN	Simplan Systems, Inc.
INSIGHT	Insight Software
PLATO	OR/MS Dialogue, Inc.
Microcomputer Packages	
FOCUS/PC	Information Builders, Inc.
IFPS/PERSONAL	Execucom Systems, Corp.
NOMAD 2 PC	D&B Computing Services
ENCORE	Ferox Microsystems
PC ANALECT	Dialogue, Inc.
PC EXPRESS	Information Resource, Inc.
PREFCALC	Euro-Decision
ENABLE	Software Publishing Group
SYMPHONY	Lotus Development
FRAMEWORK	Ashton-Tate

microcomputer-based DSS software packages are given in *Table 1*.)

The purpose of developing a short list of generator products is to narrow the field of available DSS software for consideration during Generator Evaluation. A short list of candidate DSS software (two or three) eliminates any unnecessary effort or confusion which might result because too many alternative DSS products are evaluated.

3.1.1. Identity Candidate Software

The project team must first identify available DSS products that operate within the enterprise's specific computer hardware and are compatible with its operating system and database management system (DBMS). To accomplish this task, there are several publications (e.g., Datapro Directory and ICP Directory) which provide profiles of DSS software vendors and the products they offer.

3.1.2. Screening Criteria

At this point in the process, since a detailed analysis of user requirements has not likely been performed, screening criteria should be kept to a rather high level. Otherwise, these criteria will become so specific that it might become impossible to meet them with any commercially-available DSS generator. The list of criteria will contain relatively few items and should concentrate on functional requirements not commonly provided by DSS packages and which are very specific to the organization evaluating DSS software.

Some of the screening criteria are requirements that cannot be compromised and are easy to define objectively, such as compatibility with a particular operating system. However, other criteria are less definite, such as a vendor's ability to adequately support the software. Screening criteria can be categorized into four major types: (1) technical requirements; (2) functional requirements; (3) documentation and training; and (4) vendor information.

Technical Requirements – An organization's hardware and software strategy will likely dictate the high-level criteria in the technical area. To be considered, a package must fit the framework of the proposed system; it must be compatible with the hardware and software direction already identified (usually IS planning). The operating system is clearly a strict technical requirement. Others could include programming languages, peripherals, memory needs, or data communication capabilities. If relatively high transaction and report volumes are required, then the technical architecture of the DSS software package must support efficient processing.

Functional Requirements – The functional requirements of a DSS generator can be classified according to the following system components (see *Fig. 3* adapted from [2]): (1) Dialog Management; (2) Data Management; and (3) Model Management. The functional requirements associated with each of these three system components readily distinguish DSS generator evaluation and selection from other software appraisal efforts [27], where functional requirements are less unique to the IS type.

The dialog component of a DSS is the software and hardware that provides the user interface for the system. It presents the process outputs to the users and collects the inputs to the DSS. Building a DSS without databases and associated DBMS will be extremely difficult, since this component provides the data needed for decision making. The modeling component gives decision makers the

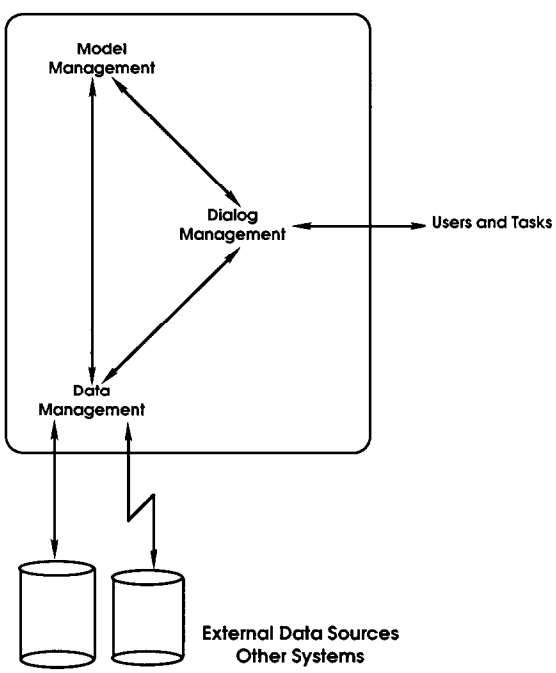

Fig. 3. Functional Components of A Decision Support System.

Table 2
High Level Screening Criteria of DSS Components.

Dialog Management	Data Management	Model Management
Multiple Dialog Styles	Variety of Logical Data Views	Library of Optimization Models
Command Language	Relational DBMS	Linear Programming
Menu	Hierarchical DBMS	Dynamic Programming
Question/Answer	Network DBMS	Integer Programming
Object Oriented	File Management System	

ability to analyze a problem by developing and comparing alternative solutions.

Table 2 lists several examples of functional criteria to conduct the first-cut screening according to the DSS components. High-level criteria for the dialog management component would be that the DSS generator offers several dialog (e.g., command language, menu, question/answer, and object oriented) to accommodate different cognitive styles of various users. The data management component could call for both relational and hierarchical DBMS. Model management criteria might require the availability of multiple optimization models, such as linear, dynamic, and integer programming models.

Documentation and Training – DSS software packages normally include the documentation required to install and support the DSS generator. It should be detailed, complete, and easy to understand. Poor documentation makes it more difficult for personnel to understand the package, and would also increase the time to modify it, if necessary. The availability of vendor-developed training sessions and materials may be very important, especially when the organization's personnel are inexperienced in implementing software.

Vendor Information – A vendor's ability to support its package through training, consultation, installation, and maintenance assistance is an important consideration in evaluating DSS software packages. Whenever the extent of a vendor's support for a generator package is unclear, the vendor should be contacted so the point can be clarified. He should also be able to refer an evaluation team to a user who is willing to talk to them about the DSS package and the accompanying support.

The financial stability of a vendor can also be an important consideration. Financially successful vendors that have been in existence for more than a few years are more likely to adequately support their packages initially and in the future. Such vendors attract and retain competent personnel, so that, in addition to having the funds available for support, they also have the personnel.

It is important to remember, however, that financial success alone does not ensure adequate, continued support. Vendor image, package reputation, the unit price, and the number of installations are also important considerations. Either the vendors themselves or the users to whom they directed the prospective buyer should be able to provide the needed information in these areas. Vendor support should always be carefully investigated.

3.1.3. Pick Finalists

The matching of the screening criteria against the list of DSS software and their capabilities will cause the elimination of many (but hopefully not all) generators. The following are typical reasons to eliminate potential DSS software candidates: (1) a vendor has only three employees and has been in business less than a year; (2) operating systems software and hardware is not supported by a vendor; and, (3) system documentation is inadequate.

By reducing the number of DSS software packages under consideration from as many as twenty to two or three, a project team can more effectively devote its attention to the critical details that can make the difference between selecting an adequate DSS and selecting a superior one. Moreover, by determining what DSS software packages are available for the application, the screening process also determines whether a DSS generator can be used or if a specific decision support system should be constructed from DSS tools (see Fig. 2).

3.2. DSS Generator Evaluation

This second stage focuses on the two or three DSS generators that were identified in the screening of DSS software. The objective is to evaluate

in detail the DSS generator finalists and select the one software product that best meets the needs of the organization. The primary tasks of DSS software selection are: (1) to further define the detailed evaluation criteria; (2) obtain generator product information; and, (3) evaluate the DSS software finalists and pick one as the best alternative.

3.2.1. Expand Evaluation Criteria

The screening criteria are expanded in more detail and fall into the same four categories: (1) technical requirements; (2) functional requirements; (3) documentation and training; and (4) vendor information. Although all categories are expanded during generator evaluation, the *functional requirements* receive the majority of attention and are related to the dialog, data, and model management components of the DSS generator.

The purpose of this task is to develop a rather comprehensive functional view of the proposed system and to summarize the requirements that must be satisfied by the DSS. The definition of functional requirements must be detailed enough to provide users and management with a complete view of the proposed system. This task should emphasize how the new system will work in the business environment. As the project team defines the functional criteria, they should also document the levels of importance and need to the user. The following functional requirements for DSS software, identified in [19,22 and 27], are those for a hypothetical firm: (1) user friendliness; (2) hardware and operating system considerations; (3) variety of dialog styles; (4) data handling functions; (5) management of internal and external databases; (6) logical data models; (7) analysis; (8) forecasting and statistics; and, (9) graphics. These categories represent an outline for the extended functional criteria of the DSS generator. *Table 3* exhibits an expansion of the above summary list of functional generator requirements categorized by the DSS as dialog, data, and model management components.

Obviously, some criteria are more important or critical to users than others. To reflect the relative importance of each criterion, the users must weight or assign a level of importance (such as "3" for essential, "2" for important, or "1" for optional) to each criterion. To demonstrate this element of the evaluation and selection methodology, an example of the weighting procedure is given below using the "Analysis" criteria from Table 3 for a hypothetical enterprise.

Table 3
Detailed Functional Criteria for DSS Software.

Dialog Management Component
User Friendliness
Consistent, natural language commands
"Help" command and error messages
Novice and expert modes
Hardware and Operating System Elements
Printer and plotter support
Variety of input device support
Variety of Dialog Styles
Menu
Command language
Object oriented
Question/Answer
Graphics
Basic plots and charts
Multicolor support
Previewing of output
Data Management Component
Data Handling Functions
Dictionary
Creation, deletion, update, and query
Management of Internal and External Databases
Extraction
Capture
Integration of data sources
Logical Data Views
Record
Relational, Hierarchical, and Network DBMS
Rule
Model Management Component
Analysis
What-if and goal seeking
Monte Carlo
Mathematical optimization
Forecasting and Statistics
Basic statistical functions
Time series with seasonal adjustment
Multivariate statistics

Analysis	Weight
What-If	1
Goal Seeking	3
Monte Carlo	2
Optimization	2

The purpose of this weighting process will be discussed in more detail in a later section (Evaluate DSS Generators).

Criticism levied against weighting schemes for software selection decisions [14] can be minimized through the screening process and development of a short list. Naumann and Palvia [20] successfully applied weighting and scoring measures to select a systems development tool from a relatively short list (4) of candidate techniques. By weighting criteria for only two or three packages rather than for a dozen (in which case the aforementioned criticism is probably valid), the proposed evaluation process allows for a very detailed and focused inspection of just the few best alternative DSS software products.

The proposed approach is an efficient, pragmatic, and managerially-oriented evaluation and selection procedure. The advantages of using a DSS generator may be reduced significantly by a lengthy evaluation and selection process which often delays the prompt installation of the software and postpones the benefits available from rapidly producing a specific DSS from a generator.

3.2.2. Obtain Package Information

Once the system requirements have been established, and the criteria have been reviewed and weighted, the capability of each DSS generator to satisfy the requirements must be measured. Several techniques may be used to gather enough information to determine how well each package meets the requirements.

In many cases, the project team can meet directly with the vendor sales and support personnel and discuss each requirement. But if requirements are so comprehensive and detailed that a more formal procedure should be followed, a request for proposal (RFP) can be submitted to vendors. In situations where requirements are less detailed and complex, the RFP can be replaced by a less formal and more direct procedure, for example a basic letter of request.

3.2.3. Evaluate DSS Generators

Once the vendors' responses to requirements have been received, the actual evaluation process can begin. The review is very detailed at this point, since the project team is looking for specific strengths and weaknesses of each package.

The project team is searching for deciding factors – not only *what* DSS software packages have and how well they provide it, but also what they *don't* have. Detailed information is desired on the functions of the DSS software and its related processing, including if and how functions that are not included in the DSS generator could be implemented.

Evaluation matrix – An evaluation matrix should be constructed to organize and assimilate all necessary information. The first step in constructing this matrix is to set up a rating scale that indicates, for each evaluation criterion (i.e., technical, functional, documentation, and vendor-related), how easily each package is able to meet that specific criterion. These rating scores are then multiplied by the weight factor for that criterion. The weights reflect the relative importance of each of the criteria, while the rating scores show how well a given package meets each user criterion.

Using the previous example of the "Analysis" criteria, a project team might employ a scale of 0-3: "3" if the DSS package totally meets the requirements; "2" when the product does not meet the requirement completely but enough so that tailoring is not warranted; "1" if the criterion would be met with some tailoring; and, "0" when the package does not meet the criterion at all. (Other rating, scoring, or evaluation methods (e.g., [23,24, and 31]) could also be appropriate depending on the particular user requirements.)

Using this scale, two prospective DSS software packages, ABC and XYZ, were scored:

Analysis	ABC	XYZ
What-If	1	3
Goal Seeking	3	3
Monte Carlo	3	2
Optimization	2	3
Subtotal	9	11

These results would indicate that XYZ meets the hypothetical requirements better. However, these may not be accurate! The detailed requirements for "Analysis" (an element of the model management component in the DSS generator) are probably not equal. The weighting factors, which were established earlier, are absent from these calculations.

If the scores are adjusted by multiplying the rating score for each "Analysis" criterion by the

corresponding weight factor, then the figures would appear as the following:

Analysis	ABC Generator			XYZ Generator		
	Weight	Rating	Total	Weight	Rating	Total
What-If	1 ×	1 =	1	1 ×	3 =	3
Goal Seeking	3 ×	3 =	9	3 ×	3 =	9
Monte Carlo	2 ×	3 =	6	2 ×	2 =	4
Optimization	2 ×	2 =	4	2 ×	3 =	6
Total			20			22

Note that the weight scores for each "Analysis" criterion are the same for both DSS generators, reflecting the user's decision about the relative importance of each of the criteria. On the other hand, the rating scores indicate how well each DSS software package meets each criterion established by the users.

A "total possible points" column could represent an ideal package meeting 100 percent of requirements. The "constant" weight factor would be multiplied by the highest possible rating score for each of the "Analysis" criteria. An ideal package would have all of the analysis features (i.e., what-if, goal seeking, Monte Carlo, and optimization) as standard. This would mean a score of "3" in the example.

This scoring by matrix is a small part of the DSS software evaluation process; but this exercise or calculation must be completed for each criterion, such as those listed in Table 3. The total points of each criterion for each DSS software package would then be recorded in a large matrix. A partially completed matrix for the hypothetical example is given below.

Functional Requirements	ABC	XYZ	Total Possible Points
1. User Friendliness	23	21	24
2. Hardware and Operating System Elements	13	13	16
3. Variety of Dialog Styles	20	22	24
4. Data Handling Functions	15	14	18
5. Extraction from Internal or External Data Base	22	20	22
6. Logical Data Views	18	17	20
7. Analysis	20	22	24
8. Forecasting and Statistics	12	10	15
9. Graphics	26	21	30
Subtotal	169	160	193

As established in an enterprise software policy, hurdle scores ensure that DSS generators provide adequate coverage of requirements. A policy for software selection might be that all DSS generators must satisfy at least 80 percent of the requirements. In the prior partial matrix, the "ABC Generator" satisfied 88 percent of the criteria, while "XYZ" covered only 83 percent. Both DSS software packages met the minimum hurdle.

In this instance, where both packages exceed the hurdle percentage scores, an index could be constructed by dividing the cost of each respective package by its score, giving the price in dollars per requirement point. For example, if the ABC package sold for $100,000 and XYZ costs $75,000, the respective indices would be $591.72 and $468.75. This indicates that the XYZ package would provide more than the minimum functional requirements and almost equal coverage of these requirements as the alternative DSS software package but at considerably less cost per requirement.

3.2.4. Additional Selection Requirements

The matrix scores are not necessarily the determining factor for selecting a particular DSS generator. The matrix should be used as a decision tool – a means for organizing and summarizing the significant quantity of information that the project team has collected. The highest score on the evaluation matrix may not always indicate the best DSS generator. The matrix scores may not accurately reflect certain intangible factors such as the cosmetic appearance of reports and screens, how easy it will be to use the DSS software, etc.

Tailoring – The matrix may not indicate how much time or the level of technical expertise needed to "tailor" the DSS generator. Tailoring can be either costly if it is relatively extensive, or difficult if the internal structure of the software is complex. The importance of the technical processing architecture will depend on how much tailoring is anticipated. Furthermore, the architecture of the DSS generator also determines how much modification is even possible.

Documentation – A decision to use a particular DSS generator should not be made on the basis of functional requirements alone. The DSS software's

documentation is a very important non-functional factor. Its accuracy and level of detail can affect the time it will take to evaluate and modify the package.

Comparing documentation is sometimes very difficult at this stage of the evaluation process due to differences in format, style, etc. Still, it is important to review the vendors' documentation and to reconfirm that the information collected on maintenance and support, for instance, is accurate and correct. At this stage, the vendor should be able to refer the project team to current users of his software. The comments of these customers should prove invaluable. Site visits and demonstrations of the DSS software in operation may be helpful.

3.3. Specific DSS Design

Assuming that DSS software which is anticipated to provide satisfactory performance has been selected (see Fig. 2), the project team is ready to confirm the selection by developing some specific DSS prototypes based on the chosen generator. The primary reasons for this stage are to ensure that the DSS package can be used effectively and to provide one last chance to reconsider the DSS software decision.

It is often difficult to determine the degree of user satisfaction until the design process has begun for specific applications utilizing the selected DSS software. Therefore, this stage involves the design of demonstration prototypes of specific DSS built from the DSS generator [11 and 19]. Such prototypes can provide significant benefits before finalizing the selection decision [1,11,12 and 13]. These benefits afford much information for the evaluation and selection process [19], including: (1) estimates of programmer productivity; (2) measures of computer resource utilization; (3) personnel requirements for the DSS software; (4) performance of the documentation under actual working conditions; and, (5) experience with the iterative development process using the DSS generator for building specific DSS applications.

In addition to prototyping specific DSS with the selected DSS software before actual purchase, Meador and Mezger [19] suggests conducting "benchmark evaluations" which would be undertaken during this stage of the proposed evaluation and selection process. A benchmark evaluation is a series of simulated tests for a comprehensive set of the DSS software's features. The simulations attempt to determine the level of computer system resources utilized by the various capabilities of the DSS package. Resources include CPU cycles, main memory, input/output activity, and response time. The programs or models to be tested are specifically designed to execute the features or capabilities of the DSS generator, rather than to solve specific DSS application problems.

3.3.1. Alter Functional Requirements

Based on the capabilities of the selected DSS generator as experienced in the prototyping exercise of specific DSS and benchmark testing, the definition of user requirements might be altered to include package features not previously considered, or to change or eliminate others. The modified requirements should be reviewed with the users. The effect of DSS software deficiencies perhaps can be minimized by altering user procedures or postponing the implementation of some requirements until generator enhancements could be made.

3.3.2. DSS Software Modifications and Supporting Programs

Typically, the specific DSS being developed requires certain functions and interfaces not provided by the software. If a DSS generator does not meet all the functional requirements of a system, the following alternatives should be considered: (1) persuade the vendor to include additional features; (2) develop supplemental software; and, (3) modify the vendor's software. The chosen alternative will depend on the extent of the DSS generator's deficiencies, the potential costs and benefits of altering the software, and the size and technical skills of the programming staff.

Vendor-Supplied Enhancements – If possible, the vendor should be persuaded to do the modification for the purchaser. This is often the best alternative, since the vendor will usually update and maintain the software on a routine basis.

Supporting Programs – Developing software to supplement the vendor's DSS package is often the most practical alternative. The vendor will normally continue to service the DSS generator; but if this alternative is selected, the supplemental

software should conform to the standards used by the vendor in developing the DSS generator.

Alter Code – Modifying a DSS generator is usually not recommended. If the software is modified, the vendor may be reluctant or may even refuse to service the package. Updates to the software may not be compatible with the modifications effected.

In some cases, this may not even be an option, since the purchaser of the DSS generator does not have (or cannot get at any price) a copy of the source code. In this instance, all that the purchaser can do is to build a front-end or back-end to the software package.

3.3.3. Finalize DSS Generator Selection

It is not unheard of for an organization to complete the last stage of the evaluation and selection process for DSS software, only to realize that the DSS generator selected is not the best choice. Perhaps too many compromises have been made and users are no longer satisfied. Possibly, the tailoring effort has become so extensive that a custom DSS (i.e., specific DSS application built from tools) would be a better choice (see Fig. 2). Therefore, a final commitment to using a particular DSS generator should be avoided until the design of specific DSS using the potential software package has progressed to the point where user satisfaction is ensured.

The following section provides an illustrative example of how the DSS software evaluation and selection methodology works. It uses a real-world case, the Wildlife and Fisheries Department, to demonstrate the applicability of the proposed methodology.

4. Case Example: The Wildlife and Fisheries Department

The Wildlife and fisheries Department (WFD) is a state government office responsible for developing a strategy to manage its state's deer population. Each year, the Department chooses to either maintain, increase, or decrease the deer population in each county. Population regulation may be achieved through the selective issuance of hunting permits. Therefore, it is essential that WFD be able to accurately predict deer population levels within each of the state's counties.

4.1. Case Background

Deer hunting regulations require that each successful hunter reports his kill to a check station within 24 hours. Within the state of Indiana, for example, there are approximately 240 such check stations. When a hunter brings his deer to a check station, the deer is tagged and the hunter is required to fill out a form listing the county of kill, date of harvest, and the sex of the deer taken. A copy of this form is then forwarded to the WFD on a weekly basis for the duration of the hunting season.

Once the WFD receives the data, it must sort it by county. Certain statistical analysis, such as the percentage of yearling, must also be calculated. Once the data is adequately prepared, it is ready for use in a predictive model. With data, such as population fecundity and survivorship by age class, a spreadsheet model could produce estimates of the state's deer population. The harvest number can then be varied to show its effects on the deer population within each county of interest. In this way, the WFD can determine the number of permits to issue within each county. The spreadsheet output along with recommendations for management are then incorporated into an annual report which is presented to the WFD's "administrator" who makes the final decision.

The professional constructuion and presentation of the report may also influence the "administrator." Therefore, it is to the advantage of the WFD staff to have its report neatly processed. For illustrating the calculated trends in the deer population, graphical and tabular summaries should also be incorporated into the report.

4.2. DSS Generator Screening

While practically any spreadsheet program could perform the mathematical requirements of the population forecasting model, the production of the complete report requires additional software capabilities. A functional and technical analysis of the procedures to be used by the WFD revealed the following list of requirements (i.e., screening criteria).

IBM-PC Compatibility – This was essential for interfacing with the corresponding systems of neighboring states which would provide relevant

input data. As a base level operating system, DOS 3.0 was identified.

Database – The database must be large enough to incorporate all of the data from the approximately 50,000 individual kill reports received each year. One hundred characters were needed for each record. Also, the maintenance of a five-year database was recommended.

Statistical Analysis – Functions such as mean, standard deviation, relative percentages and to a lesser extent regression analysis, were necessary for the data analysis.

Spreadsheet – The dimensions of the spreadsheet should be large enough to accept data from all 92 state counties and perform the necessary computations. In addition, the spreadsheet function must link individual county spreadsheet models into a comprehensive statewide summary.

Word Processor – This component required both spelling and grammatical checks to assist the biologists in preparing their reports.

Graphics – The software had to produce good quality line, grouped line, and bar charts for showing trends and supporting quick information assimilation by the WFD users.

File Import / Export – In addition to exchanging files from other computer programs, the chosen software package must be able to accept data from existing files which contain a significant amount of needed historical data. The acceptance of this material by the new system without major modification would allow for considerable savings in time and money.

Documentation – The documentation had to be detailed, organized and precise. Both external documentation (books, manuals, videotapes) and internal (on-screen help) should be available. Online documentation should be context sensitive, and external documentation might include videotape sessions.

Two classes of potential "commercial" DSS software, namely basic and integrated spreadsheets, were identified by the WFD. It should be noted here that these packages are commercially-available products which might differ from the "ideal" DSS generator defined in Section 1.0 of this paper. The list of spreadsheet packages available in the market is very long, ranging from the most simplistic to the very sophisticated. A preliminary list, based on the two potential classes of DSS software, is given below. From this roster, a short list of three DSS generators was developed by applying the aforementioned screening criteria (i.e., functional and technical criteria, etc.) and eliminating those packages which were not considered adequate for more detailed inspection.

Basic Spreadsheets	Integrated Spreadsheets
Lotus 123	Enable
Visicalc	Symphony
Quattro	Framework
Excel	Smart
Multiplan	Electric Desk
Supercalc	Get Organized

A brief evaluation of the main characteristics of the basic spreadsheets uncovered the following attributes: (1) good features in terms of mathematical abilities; (2) lack of graphical components (except in Lotus and Quattro); (3) none of them has a word processing component; and, (4) none of them can handle the necessary database size.

The functional and technical environment described earlier in this section documented the crucial nature of the graphical, word processing, and database capabilities. Consequently, the basic spreadsheet packages were eliminated and would not be included in the final evaluation. While separate software products may be combined into one DSS to perform the necessary functions, the number of possible combinations was too large to consider.

The further matching of screening criteria against the list of integrated spreadsheet packages eliminated all but the following potential DSS generators: Symphony (Lotus Development), Framework (Ashton-Tate), and Enable (Software Group). The other potential DSS generators failed to be included in the short list since they did not meet one or more of the screening criteria (i.e., database size or spreadsheet linking functions).

4.3. DSS Generator Evaluation and Selection

Based on the aforementioned list of screening criteria, a scheme for the detailed evaluation and

selection of the generator software was based upon the following categories: (1) technical criteria; (2) functional criteria; (3) documentation criteria; and, (4) vendor criteria.

Technical Criteria – The primary technical requirement for the WFD computing environment is IBM and DOS compatibility. A version of DOS 3.0 or higher was also considered a necessity.

Functional Criteria – The functional requirements of the WFD application include the database, statistical, spreadsheet, word processing, graphics creating, file handling and exchange, and documentation criteria which was listed in the initial phase (DSS Generator Screening).

For purposes of evaluating the three commercial DSS generators listed above, the preceding criteria were weighted according to their relative importance to the WFD operations. The following scale was used: "3" expressing a crucial function; "2" meaning significant; and "1" noting optional. *Table 4* describes the relative importance assigned to each criterion.

In addition to the weighting scale, a rating system was used to indicate the respective software's performance on each of the criteria. This rating was based on a four-point scale (3 = good, 2 = fair, 1 = poor, and 0 = not available).

Table 4 depicts both the weights and the total scores (weights multiplied by rating scores) developed for each DSS generator. The matrix totals give Enable the edge over the other DSS software packages evaluated. However, the tabulated scores were rather close (i.e., Enable = 91, Framework = 85, Symphony = 76).

While scores were close, Enable was the best performer in the most critical areas. Of the criteria considered to be the most crucial, Enable received a perfect (3) rating in four of the them. Perhaps the most important criterion was database size. Enable was the only DSS generator capable of handling the required 50,000 records. Enable was also superior in both internal and external documentation, which was critical since non-computer personnel would be directly involved in the operation of the DSS software. However, Enable was relatively inferior in two areas, namely its spreadsheet linking function and its vendor reputation (i.e., lack of market prominence compared to the other vendors).

Table 4
WFD Case: Evaluation Matrix for DSS Software.

Criteria	Weight	DSS Generator Scores		
		Sym-phony	Frame-work	Enable
Technical				
IBM Compatibility	3	9	9	9
Functional				
Database Size	3	3	3	9
Basic Statistics	3	9	9	9
Regression Analysis	1	0	0	0
Spreadsheet Size	3	3	6	6
Spelling check	2	0	6	6
Grammar Check	1	0	0	0
Graphics	3	6	6	6
Spreadsheet Linking	2	6	6	4
File Import/Export	2	2	6	6
Combine Graphics and Text	2	2	4	6
Menu Dialog	2	6	6	6
Command Dialog	2	4	2	6
Documentation				
External Documentation	2	2	4	6
On-Line Help in Context	3	9	9	9
Vendor				
Reputation	3	9	9	3
Total		76	85	91

Overall, Framework had the best word processing features, Enable offered the most useful database module, and Symphony's spreadsheet capabilities were exceptional. However, the superior package at providing comprehensive functionality in spreadsheet, database, and word processing in a single package was Enable.

4.4. Specific DSS Applications

At this point in the evaluation and selection process, specific DSS applications were constructed with Enable, the chosen DSS generator. Representative prototypes were built to evaluate the DSS software's ability to handle not only functional requirements but also to appraise its operating efficiency. If satisfactory performance was achieved by the DSS generator, then multiple copies would be purchased or a site license

acquired. In the event that performance was less than satisfactory, the methodology prescribes construction of the specific DSS application from tools as needed to achieve the application requirements and user expectations.

4.5. Case Summary

It is important to note that no package provided a perfect fit for the WFD case. For example, none of the software offered regression analysis as a standard feature. A grammar check was also not available in any package.

As described by Sprague [26], an "ideal" DSS generator does not likely exist. The ideal generator would be developed over a long period of time in a fairly narrow problem domain. In practice, however, most specific DSS applications are being developed with general purpose DSS generators such as FOCUS, IFPS, and Lotus 1-2-3 [31, p. 205].

5. The Impact of DSS Software on Specific DSS Development

Because DSS software can reduce the costs of developing specific decision support systems, organizations should investigate the possibility of using DSS generators during the systems planning process. Obviously, this may increase the personnel requirements of systems planning. However, such an investigation is valuable even if a custom approach to developing specific DSS from tools is determined to be more appropriate.

The evaluation process will help familiarize personnel with the functional requirements of proposed DSS applications. Furthermore, the availability of good DSS packages may have a significant effect on the organization's hardware and software strategy. Although the evaluation of DSS products can increase the cost of systems planning, the use of a DSS generator can clearly reduce overall development costs for specific DSS.

The following sub-sections discuss the effects that DSS software may have on developing specific DSS applications. In particular, the impact on systems analysis and design, installation, and operating support is assessed.

5.1. DSS Software Impact on Systems Analysis and Design

DSS software selection often precedes the design of a specific DSS since the latter will be based on the chosen DSS package(s). Therefore, DSS software evaluation and selection is usually an additional effort that would not be required (or at least not to the same extent) for building specific DSS from tools.

While using DSS generators will often reduce the time and effort needed to complete the preliminary design, this reduction is frequently offset by the amount of work involved in evaluating and selecting a DSS package. Therefore, the overall effort for systems analysis and design may remain fairly constant regardless of whether users decide to use DSS generators or develop specific DSS from tools.

5.1.1. User Requirements and Application Design

The systems analysis activity for developing specific DSS determines whether or not the user's requirements are met by DSS software. While the effort needed to define these requirements is not reduced when a DSS generator is used, specific DSS design usually requires fewer personnel than when a specific system is custom developed from tools.

If user requirements are not satisfied by a DSS software package, the investment in time and effort depends on the amount of analysis and design that is necessary to meet the user's needs. This might involve developing manual procedures, interface capabilities, as well as additional software modules.

5.1.2. Technical Design

The work required for technical design is significantly reduced when DSS software is used. This is apparent because the technical architecture, database, and system processes for the DSS generator have already been defined by the vendor. However, what is necessary here is a confirmation by the project team that the architecture of the DSS package is compatible with the organizations's technical environment.

The personnel requirements for designing security and control mechanisms will vary according to the particular DSS software package being used. For many packages, however, security and

control is a weak area that requires additional work.

The operating performance of the system can be affected by the use of DSS software. It may suffer especially if many options are used, since the generalized software logic could require longer execution times than specific DSS developed from tools. Therefore, operating performance should not be overlooked during systems design just because DSS software is being used. This underscores the need for benchmark testing as previously mentioned.

5.2. DSS Software Impact on Systems Installation

Clearly, the greatest savings in developing specific DSS from a DSS generator are realized during installation. When a DSS package is installed, detailed systems design, programming, and debugging should require less effort than they would in the installation of a custom developed, specific DSS. Computer programs have already been designed, and coding and testing completed when DSS software is utilized.

5.2.1. Detailed Design and Programming

The primary purpose of using DSS software is to reduce the work performed for detailed design and programming. If the user requirements are not completely met by the DSS generator, some tailoring may be necessary. Any modification at this point needs thorough documentation which should be made available (by the system support group) for on-going maintenance of the DSS software.

Even if no program code changes to the DSS software are necessary, there is usually some detailed design, programming and testing required. Other production systems (e.g., transaction processing) might need to be changed, or interface programs be developed. Data conversion facilities are commonly needed to load the initial production (raw) files.

5.2.2. Systems Testing

The reduction of effort in detailed design and programming does not imply that system testing is less critical when DSS software is used. It is just as important as for specific DSS built from tools, if not more. A combination of conditions could be unique to a particular user and may not have been system tested by the vendor.

Some additional effort is required to perform physical installation of the DSS package. The project team should verify that the software delivered by the vendor is complete and operates in the company's technical environment. Some vendors provide a limited test case to be executed during what is often called the "acceptance test." All other segments of systems installation are still required and usually are not materially affected by DSS software.

5.3. Operating DSS Software Support

The amount of support work involved with installed DSS software usually depends, to a large degree, on the quality of the vendor support. DSS generator packages are more difficult to maintain if vendor support is poor.

The type of necessary maintenance also determines the extent of the impact DSS software has on supporting installed decision support systems. Maintenance of a DSS generator is categorized as follows: (1) maintenance of the code performed by the vendor, including new releases, temporary program fixes in response to bugs or code changes requested by the user; (2) maintenance of the code performed by in-house personnel, which refers to modifying the DSS software's program code; and, (3) maintenance of existing parameters and selected options (i.e., most parameter-driven software is designed to be maintained by the user.)

When vendor modifications are implemented to an installed DSS generator, it is very important to maintain a listing of updates made to the DSS software, and to keep track of specific modifications and who made them. It is very useful for the vendor to know the status of the software when he is asked to investigate problems.

When implementing new vendor releases, several levels of modification and testing may be necessary: (1) acceptance testing of the new release; (2) modification of the new release to reflect prior user changes and parameters; (3) testing the modified release with acceptance data; and, (4) testing the modified release with a system model. The net effect of installed generator effort varies with the particular package, the quality of vendor support, and the extent of maintenance. With a DSS generator, the number of necessary maintenance changes is often fewer than for custom-built specific DSS.

If the DSS software is well designed, it incorporates additional functions that can be activated as needed. The activation and testing effort required in this case would be far less than the effort to add the same functions to a specific DSS built from tools. If the DSS generator is well coded and tested, the number of bugs occurring immediately after conversion should be substantially fewer than with custom-developed specific DSS.

6. Concluding Remarks

Using DSS generators for the development of specific decision support systems will reduce personnel requirements and development costs. Conducting the evaluation of DSS software increases the effort necessary for developing specific DSS, but this undertaking is offset by the advantages of using a generator package. Despite the promises offered by DSS software, the performance of some DSS generators is much less than expected. Weak or non-existent selection procedures may explain much of this poor implementation record. The methodology proposed in this paper will hopefully reduce the risks associated with decision support systems software and facilitate success in developing specific DSS from generators.

The most critical phases of the methodology are the first (the development of a short list) and the third (design of specific DSS with the selected generator). Initially, the screening process determines whether a generator is feasible and reduces the number of DSS software packages to be evaluated in detail. Finally, the development of specific DSS with the selected generator ensures that the DSS software can be used effectively and provides a last chance to consider building specific DSS from tools.

As stated in the first section of this paper, while prior work provided partial guidelines for DSS software evaluation and selection, no unified and comprehensive methodology (as presented herein) was suggested. It is the authors' belief that this methodology is quite easy-to-use and pragmatic. Its intent is to efficiently choose a DSS generator that meets the application needs and user expectations from the employment of packaged software.

References

[1] Alavi, M., "An Assessment of the Prototyping Approach to Information Systems Development," *Communications of the ACM*, Volume 27, Number 6, June 1984, pp. 556–563.

[2] Ariav, G. and M.J. Ginzberg, "DSS Design: A Systemic View of Decision Support," *Communications of the ACM*, Volume 28, Number 10, October 1985. pp. 1045–1052.

[3] Berst, J., "The ABC's of Evaluating Packaged Software," *Interface Age*, February 1983, pp. 35–38.

[4] Breslin, J., *Selecting and Installing Software Packages*, Quorom Books, Westport, Connecticut, 1986.

[5] Courbon, J.C., J. Grajew and J. Tolovi, Jr., "Design and Implementation of Decision Supporting Systems by an Evolutive Approach," Unpublished Working Paper, University of Grenoble, France, 1980.

[6] Curry, J.W. and D.M. Bonner, *How to Find and Buy Good Software: A Guide for Business and Professional People*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[7] Decision Support Software, Inc., *EXPERT CHOICE User's Manual*, McLean, Virginia, 1983.

[8] Execucom Systems Corporation, *IFPS User's Manual*, Austin, Texas, 1982.

[9] Gray, C.D., *The Right Choice: A Complete Guide to Evaluting, Selecting, and Installing MRP II Software*, Oliver Wight Limited Publications, Inc., Essex Junction, Vermont, 1987.

[10] Hogue, J.T. and H.J. Watson, "Current Practices in the Development of Decision Support Systems," *Proceedings of the Fifth International Conference on Information Systems*, Houston, Texas, 1984, pp. 117–127.

[11] Janson, M., "Applying a Pilot System and Prototyping Approach to Systems Development and Implementation," *Information And Management*, Volume 10, Number 4, 1986, pp. 209–216.

[12] Keen, P.G.W., "Adaptive Design for Decision Support Systems," *Data Base*, Volume 12, Number 3, 1980, pp. 15–25.

[13] Keen, P.G.W., "Value Analysis: Justifying Decision Support Systems," *MIS Quarterly*, Volume 5, Number 2, 1981, pp. 1–15.

[14] Klein, G. and P.O. Beck, "A Decision Aid for Selecting among Information System Alternatives," *MIS Quarterly*, Volume 11, Number 2, June 1987, pp. 177–185.

[15] Lauer, T.W. and M.T. Jelassi, "PREFCALC - A Multi-Criteria Decision Support System: A User Tutorial," Indiana University Institute for Research on the Management of Information Systems, Working Paper #714, December 1987.

[16] Lynch, R.K., "Implementing Packaged Application Software: Hidden Costs and New Challenges," *Systems, Objectives, Solutions*, Volume 4, Number 4, 1984, pp. 227–234.

[17] Lynch, R.K., "Nine Pitfalls in Implementing Packaged Applications Software," *Journal of Information Systems Management*, Volume 2, Number 2, 1985, pp. 88–92.

[18] Martin, J. and C. McClure, "Buying Software off the

Rack," *Harvard Business Review*, Volume 61, Number 6, November-December 1983, pp. 32–47.

[19] Meador, G.L. and R.A. Mezger, "Selecting An End User Programming Language For DSS Development," *MIS Quarterly*, Volume 8, Number 4, December 1984, pp. 267–281.

[20] Naumann, J.D. and S. Palvia, "A Selection Model for Systems Development Tools," *MIS Quarterly*, Volume 6, Number 1, March 1982, pp. 39–48.

[21] Reimann, B.C., "Decision Support for Planners: How To Pick The Right DSS Generator Software," *Managerial Planning*, Volume 33, Number 6, May/June 1985, pp. 22–26.

[22] Reimann, B.C. and A.D. Waren, "User-Oriented Criteria for the Selection of DSS Software," *Communications of the ACM*, Volume 28, Number 2, February 1985, pp. 166–179.

[23] Saaty, T., *The Analytic Hierarchy Process*, McGraw-Hill, New York, New York, 1981.

[24] Saaty, T., *Decision Making for Leaders*, Lifetime Learning, Belmont, California, 1982.

[25] Sanders, B.L., P. Munter and R.O. Reed, "Selecting A Software Package," *Financial Executive*, Volume 50, Number 9, September 1982, pp. 38–46.

[26] Sprague, R.H., Jr., "A Framework for the Development of Decision Support Systems," *MIS Quarterly*, Volume 4, Number 4, June 1980, pp. 1–26.

[27] Sprague, R.H., Jr. and Eric D. Carlson, *Building Effective Decision Support Systems*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1982.

[28] Sussman, P.N., "Evaluating Decision Support Software," *Datamation*, Volume 30, Number 17, October 15, 1984, pp. 171–172.

[29] Thoughtware, Inc., *LIGHTYEAR User's Manual*, Coconut Grove, Florida, 1984.

[30] Timmreck, E.M. "Computer Selection Methodology," *Computing Surveys*, Volume 5, Number 4, December 1973, pp. 199–222.

[31] Turban, E., *Decision Support and Expert Systems*, Macmillan Publishing Company, New York, New York, 1988.

[32] Waren, A.D. and B.C. Reimann, "Selecting DSS Generator Software: A Participative Process," *Policy and Information*, Volume 9, Number 2, December 1985, pp. 63–76.

[33] Welke, L.A., "Buying Software," in *Systems Analysis And Design: A Foundation for the 1980's*, edited by W.W. Cotterman, J.D. Couger, N.L. Enger, and F. Harold, Elsevier North Holland, Inc., New York, New York, 1981, pp. 400–416.

[34] Young, O.F., "A Corporate Strategy for Decision Support Systems," *Journal of Information Systems Management*, Volume 1, Number 1, Winter 1984, pp. 58–62.