

A11102 577728

NAT'L INST OF STANDARDS & TECH R.I.C.

A1102577728

/Guidance on software package selection
QC100 .U57 NO.500-144 1986 V19 C.1 NBS-P

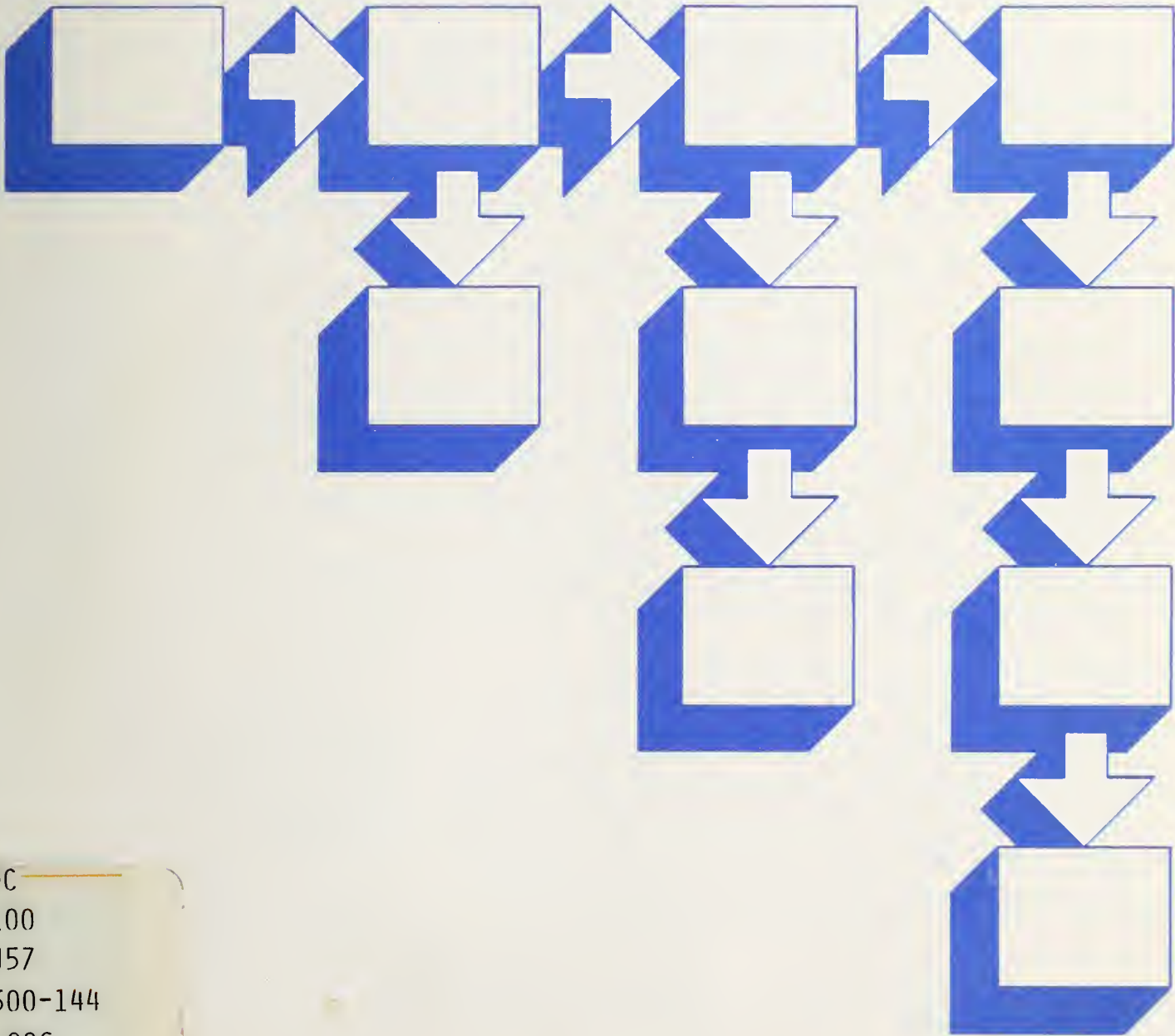
Computer Science and Technology

NBS
PUBLICATIONS

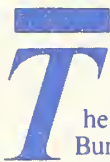
NBS Special Publication 500-144

Guidance on Software Package Selection

S. Frankel, Editor



QC
100
.U57
500-144
1986
C. 2



The National Bureau of Standards¹ was established by an act of Congress on March 3, 1901. The Bureau's overall goal is to strengthen and advance the nation's science and technology and facilitate their effective application for public benefit. To this end, the Bureau conducts research and provides: (1) a basis for the nation's physical measurement system, (2) scientific and technological services for industry and government, (3) a technical basis for equity in trade, and (4) technical services to promote public safety. The Bureau's technical work is performed by the National Measurement Laboratory, the National Engineering Laboratory, the Institute for Computer Sciences and Technology, and the Institute for Materials Science and Engineering.

The National Measurement Laboratory

Provides the national system of physical and chemical measurement; coordinates the system with measurement systems of other nations and furnishes essential services leading to accurate and uniform physical and chemical measurement throughout the Nation's scientific community, industry, and commerce; provides advisory and research services to other Government agencies; conducts physical and chemical research; develops, produces, and distributes Standard Reference Materials; and provides calibration services. The Laboratory consists of the following centers:

- Basic Standards²
- Radiation Research
- Chemical Physics
- Analytical Chemistry

The National Engineering Laboratory

Provides technology and technical services to the public and private sectors to address national needs and to solve national problems; conducts research in engineering and applied science in support of these efforts; builds and maintains competence in the necessary disciplines required to carry out this research and technical service; develops engineering data and measurement capabilities; provides engineering measurement traceability services; develops test methods and proposes engineering standards and code changes; develops and proposes new engineering practices; and develops and improves mechanisms to transfer results of its research to the ultimate user. The Laboratory consists of the following centers:

- Applied Mathematics
- Electronics and Electrical Engineering²
- Manufacturing Engineering
- Building Technology
- Fire Research
- Chemical Engineering²

The Institute for Computer Sciences and Technology

Conducts research and provides scientific and technical services to aid Federal agencies in the selection, acquisition, application, and use of computer technology to improve effectiveness and economy in Government operations in accordance with Public Law 89-306 (40 U.S.C. 759), relevant Executive Orders, and other directives; carries out this mission by managing the Federal Information Processing Standards Program, developing Federal ADP standards guidelines, and managing Federal participation in ADP voluntary standardization activities; provides scientific and technological advisory services and assistance to Federal agencies; and provides the technical foundation for computer-related policies of the Federal Government. The Institute consists of the following centers:

- Programming Science and Technology
- Computer Systems Engineering

The Institute for Materials Science and Engineering

Conducts research and provides measurements, data, standards, reference materials, quantitative understanding and other technical information fundamental to the processing, structure, properties and performance of materials; addresses the scientific basis for new advanced materials technologies; plans research around cross-country scientific themes such as nondestructive evaluation and phase diagram development; oversees Bureau-wide technical programs in nuclear reactor radiation research and nondestructive evaluation; and broadly disseminates generic technical information resulting from its programs. The Institute consists of the following Divisions:

- Ceramics
- Fracture and Deformation³
- Polymers
- Metallurgy
- Reactor Radiation

¹Headquarters and Laboratories at Gaithersburg, MD, unless otherwise noted; mailing address Gaithersburg, MD 20899.

²Some divisions within the center are located at Boulder, CO 80303.

³Located at Boulder, CO, with some elements at Gaithersburg, MD.

Computer Science and Technology

NBS Special Publication 500-144

Guidance on Software Package Selection

S. Frankel, Editor

Center for Programming Science and Technology
Institute for Computer Sciences and Technology
National Bureau of Standards
Gaithersburg, Maryland 20899

November 1986



U.S. DEPARTMENT OF COMMERCE
Malcolm Baldrige, Secretary

National Bureau of Standards
Ernest Ambler, Director

Reports on Computer Science and Technology

The National Bureau of Standards has a special responsibility within the Federal Government for computer science and technology activities. The programs of the NBS Institute for Computer Sciences and Technology are designed to provide ADP standards, guidelines, and technical advisory services to improve the effectiveness of computer utilization in the Federal sector, and to perform appropriate research and development efforts as foundation for such activities and programs. This publication series will report these NBS efforts to the Federal computer community as well as to interested specialists in the academic and private sectors. Those wishing to receive notices of publications in this series should complete and return the form at the end of this publication.

National Bureau of Standards Special Publication 500-144
Natl. Bur. Stand. (U.S.), Spec. Publ. 500-144, 121 pages (Nov. 1986)
CODEN: XNBSAV

Library of Congress Catalog Card Number: 86-600593

U.S. GOVERNMENT PRINTING OFFICE
WASHINGTON: 1986

TABLE OF CONTENTS

	Page
EXECUTIVE SUMMARY	3
0.0 INTRODUCTION	5
0.1 Overview	5
0.2 Background	5
0.3 Recommended Procedure	5
STEP ONE : REQUIREMENTS ANALYSIS	9
1.0 REQUIREMENTS ANALYSIS	10
1.1 The Requirements Analysis Team	10
1.2 The Project File	11
1.3 Defining Current Procedures	11
1.3.1 Seeking organizational input	11
1.3.2 Diagramming	12
1.3.2.1 Data flow diagrams	13
1.3.2.2 Logical flow diagrams	13
1.3.2.3 Functional diagrams	18
1.4 Improving Current Procedures	17
1.5 Constraints	18
1.5.1 Hardware environment	18
1.5.2 Cost constraints	20
1.5.3 Other constraints	20
1.6 Estimating Package Life	21
STEP TWO : THE REQUIREMENTS DOCUMENT	22
2.0 THE REQUIREMENTS DOCUMENT	23
2.1 Purpose Of The Requirements Document	23
2.2 Components Of The Requirements Document	24
2.2.1 Functions	24
2.2.2 Outputs	25
2.2.3 Inputs	28
2.2.4 User interface	28
2.2.5 Technical characteristics	29
2.2.6 Documentation	31
2.2.7 Training services	31
2.2.8 Installation services	32
2.2.9 Maintenance services	32

	Page
2.3 Requirements Document Format And Content	32
2.3.1 System purpose	33
2.3.2 Current procedures	33
2.3.3 Hardware and software configuration	33
2.3.4 System scope	34
2.3.5 Functional requirements	36
2.3.6 Reports	37
2.3.7 Data dictionary	38
STEP THREE : IDENTIFICATION OF CANDIDATE PACKAGES	40
3.0 IDENTIFICATION OF CANDIDATE PACKAGES	41
3.1 The Selection Team	41
3.2 Identification Of Candidate Packages	42
3.3 Narrowing The Field	43
3.4 The Make-or-Buy Decision	43
3.4.1 Cost accounting	44
3.4.2 Summary	46
STEP FOUR : ASSESSMENT OF SUPPORT NEEDS	48
4.0 ASSESSMENT OF SUPPORT NEEDS	49
4.1 Documentation Review	49
4.1.1 Characteristics of documentation	49
4.1.2 Documentation need evaluation	51
4.2 Modifications	51
4.2.1 Simple modifications	52
4.2.2 Software design features	52
4.2.3 Other considerations	53
4.3 Installation Review	53
4.3.1 Installation support options	54
4.3.2 Installation concerns	55
4.4 Training Review	55
4.4.1 Special considerations	55
4.4.2 Evaluation of training needs	57
4.5 Maintenance Review	57

	Page
STEP FIVE : PACKAGE SELECTION	60
5.0 PACKAGE SELECTION	61
5.1 Solicitation Of Proposals	61
5.1.1 The cover letter	61
5.1.2 The requirements statement	63
5.2 Proposal Evaluation	63
5.2.1 Proposal review	63
5.2.2 Proposal summation	63
5.3 System Walk-Throughs	64
5.4 Vendor Evaluation	65
5.4.1 Vendor stability evaluation	65
5.4.2 The vendor interview	66
5.4.3 Vendor presentations	66
5.4.4 On-site visits to package users	67
5.5 Support Evaluation	67
5.5.1 Installation	67
5.5.2 Documentation	68
5.5.3 Training	68
5.5.4 Maintenance	69
5.6 Cost-benefit Analysis	69
5.6.1 Opportunity costs	70
5.6.2 Feature accounting	70
5.7 Final Package Selection	71
5.8 Decision Approval	71
STEP SIX : CONTRACT NEGOTIATION	73
6.0 CONTRACT NEGOTIATION	74
6.1 The Statement Of Work	75
6.2 Performance Guarantees	76
6.3 Compensation	77
6.4 Other Issues	77
6.4.1 Cancellation	77
6.4.2 Condition waivers	78
6.4.3 Package ownership	78
6.4.4 Nondisclosure of proprietary information	79
6.4.5 Future performance/contract renewal	79
6.4.6 Contingencies	80

	Page
STEP SEVEN : PACKAGE INSTALLATION	81
7.0 PACKAGE INSTALLATION	82
7.1 Formation Of The Installation Team	82
7.1.1 The team leader	83
7.1.2 User representatives	83
7.1.3 The data processing expert	84
7.2 Installation Considerations	84
7.2.1 File conversion	85
7.2.2 Procedural changes	86
7.2.3 Back-up procedures	86
7.2.4 Training	87
7.2.5 Human considerations	87
7.3 Installation Planning	89
7.3.1 Preparing the environment	90
7.3.2 Operational planning	90
7.3.3 Package customization	91
7.3.4 The transition period	91
7.4 Performance Evaluation	92
7.4.1 Evaluation criteria	92
7.4.2 Performance monitoring	93
7.4.2.1 Performance milestones	93
7.4.2.2 Employee feedback	93
7.4.2.3 Maintenance contract evaluation	94
7.4.3 Insuring conformance to needs	95
STEP EIGHT : PACKAGE TESTING	98
8.0 PACKAGE TESTING	99
8.1 Testing Issues And Goals	99
8.2 Pre-installation Testing	100
8.2.1 System walk-throughs	101
8.2.2 Benchmark testing	101
8.2.3 Trial use	102
8.2.4 File conversion program validation	103
8.3 Installation Testing	103
8.3.1 Specifications matching	103
8.3.2 Sample data testing	103
8.4 Parallel Operations	104
8.5 Package Acceptance	105
8.6 Conclusion	105

	Page
APPENDIX I STEPS IN THE PACKAGE SELECTION PROCESS	106
APPENDIX II RELATED ICST DOCUMENTS	108
APPENDIX III REFERENCES AND RELATED READING	109

LIST OF TABLES

	Page
Table 1	Functions of a Requirements Document 23
Table 2	Information Found in Requirements Document . . . 25
Table 3	Sections of the Requirements Document 32
Table 4	Sources for Candidate Package Identification . . 42
Table 5	Cost Comparisons 47
Table 6	Documentation Types 50
Table 7	Issues Addressed by the Contract 74
Table 8	Installation Team Duties 83
Table 9	Components of Installation 85
Table 10	Performance Evaluation Criteria 94
Table 11	Software Package Testing 100

LIST OF FIGURES

	Page
Figure 1 Selection Process Flowchart	7
Figure 2 Example for Diagramming	12
Figure 3 Data Flow Diagram	14
Figure 4 Data Flow Diagrams	15
Figure 5 Logical Flow Diagrams	16
Figure 6 Combined Logical Flow Diagram	17
Figure 7 Symbols Used in Functional Diagrams	18
Figure 8 Functional Diagram	19
Figure 9 Sample Function and Subfunction Description . . .	26
Figure 10 Sample Input and Output Tables	27
Figure 11 File Scope Chart	35
Figure 12 Estimate of Number of Operations	35
Figure 13 Sample Functional Requirement	37
Figure 14 Example of Output-Input Listing	39
Figure 15 Sample Installation Milestone Timeline	96

GUIDANCE
ON
SOFTWARE PACKAGE SELECTION

S. Frankel, Editor

This report describes a systematic procedure for identifying and evaluating off-the-shelf software packages, and for incorporating the selected package into the organizational environment. Its purpose is to enable the layperson to choose and implement software packages with a minimum of dependence on technical personnel. The report provides guidance on each phase of the package selection and implementation process.

Key words: applications software packages; off-the-shelf software; packages; software applications; software packages; software package evaluation; software package selection.

ACKNOWLEDGMENTS

This report was funded by the National Bureau of Standards' Institute for Computer Sciences and Technology under U.S. Department of Commerce Contract NB82SBCA1634. The contributors to the report, as submitted by Science Applications International, Inc. were J.A. McCall, L.J.B. Browning, L.J. Metzger, C.A. Mayers, H.E. Campbell, and K.K. Frandson.

NOTE

In no case does identification of commercial products, publications or services imply recommendation or endorsement by the National Bureau of Standards, nor does it imply that the product, publication or service identified is necessarily the best available for the purpose.

EXECUTIVE SUMMARY

Software packages are often a cost-effective alternative to the development of custom software systems. There is a need for a framework to locate suitable packages; conduct a comparative evaluation; select the package that is the overall "best fit" for the job; and introduce the selected package smoothly and efficiently into the organization. This report provides a step-by-step description of this framework, the tasks involved, activities to be performed, and decisions to be made.

The package selection process consists of 8 component tasks:

- o **REQUIREMENTS ANALYSIS**
Defining the needs that the package should satisfy and the constraints under which it must function.
- o **THE REQUIREMENTS DOCUMENT**
Recording those needs and constraints in a complete and unambiguous manner.
- o **IDENTIFICATION OF CANDIDATE PACKAGES**
Identifying packages that meet the requirements.
- o **ASSESSMENT OF SUPPORT NEEDS**
Evaluating support services necessary to implement the package.
- o **PACKAGE SELECTION**
Selecting the package, based on the information gathered in the previous steps.
- o **CONTRACT NEGOTIATION**
Negotiating contracts with the package vendor and support services providers.
- o **PACKAGE INSTALLATION**
Installing the package in a planned and systematic manner.
- o **PACKAGE TESTING**
Testing the total system in which the package has been installed.

This report is intended for anyone who expects to be involved in the package selection and evaluation process, including managers, technical personnel and users. Managers should be actively involved in the process, planning and co-ordinating all of the decisions and activities. The

participation of technical personnel is important at each step of the process, from translating user needs into package requirements to insuring that the package is thoroughly tested. Involvement of procurement personnel is necessary to insure that the conditions of purchase and agreements with support personnel conform to the actual requirements. Thus, it is recommended that each of the people involved in the selection process at least scan this report, to acquaint themselves with the key issues, and read in depth those sections concerning issues which will require their particular expertise.

The process described in this report is generic, and does not address itself to project size or budget. In trying to address all the problems and issues involved, this report may be somewhat biased towards larger projects. However, even in selecting a low-cost microcomputer package, the considerations and component tasks are the same. The following procedure is recommended for microcomputer package selection:

- o Read the whole report to become familiar with each task, its particular problems and decisions.
- o Decide which parts of the process can be streamlined, through combining tasks and eliminating or minimizing some of the paperwork.

An important consideration in the selection of microcomputer packages is the impact the package will have on the organization. Although the package itself may be inexpensive, other factors, such as its strategic importance to running the organization; the additional costs of training personnel and revising organizational procedures; and the upheaval caused by an ill-suited selection dictate that the selection process be carefully planned and executed.

0.0 INTRODUCTION

0.1 Overview

In recent years, many organizations have discovered a practical alternative to in-house software development: the purchase of software packages that were designed to perform standard business applications, yet have the flexibility to be readily adapted to a particular organization's needs. Although numerous software packages are now available, selection is the responsibility of the customer. Many potential users find this situation intimidating; they may be experts in the application area, but few have technical backgrounds in systems analysis.

Selecting the best available package requires both a systematic approach and an awareness of the potential pitfalls of package selection and implementation. This report presents a step-by-step procedure for buying and implementing a software package.

0.2 Background

The National Bureau of Standards' Institute for Computer Sciences and Technology (ICST) has an ongoing effort to develop guidance for use by government personnel who are considering the purchase of off-the-shelf software packages.

An earlier report, "Introduction to Software Packages" (NBS Special Publication 500-114, April 1984), can be used as a reference document or directory. It contains information about the types of software packages which are available, the reasons for considering the purchase of a software package, methods for locating sources of information which describe software packages, and ways to effectively use the available information.

0.3 Recommended Procedure

The process of package specification, search, selection, and implementation should be divided into a series of tasks (Figure 1). These tasks are summarized in Appendix I. Package selection is accomplished through a process of successive elimination. A detailed specification of requirements is developed and used to filter candidate packages. Characteristics of different packages are compared to the specifications and the least appropriate candidates are identified early and eliminated from evaluation.

Subsequent evaluation narrows the list of candidates to a manageable number of final candidates--no more than four or five--which are then subjected to intensive scrutiny.

The selection process is designed to facilitate package implementation. Before a package is selected, the process provides for consideration of the necessity and difficulty of package modification, the quality of documentation, and the availability of necessary support. The major strengths and weaknesses of the available packages are evaluated and compared before the final selection is made.

The process presented in this report is a generic process, that attempts to address the whole range of packages, regardless of the size of the organization, the complexity of the application, the size of the computer system, and the budget. Where appropriate, the process can be tailored to adjust its scope to that of the project at hand. Steps can be combined for a smaller project; the project log can be less detailed for a smaller organization. It is important, however, to first understand each step of the process, its goals and possible pitfalls. Only then can an informed decision be made about how to best adjust the selection process.

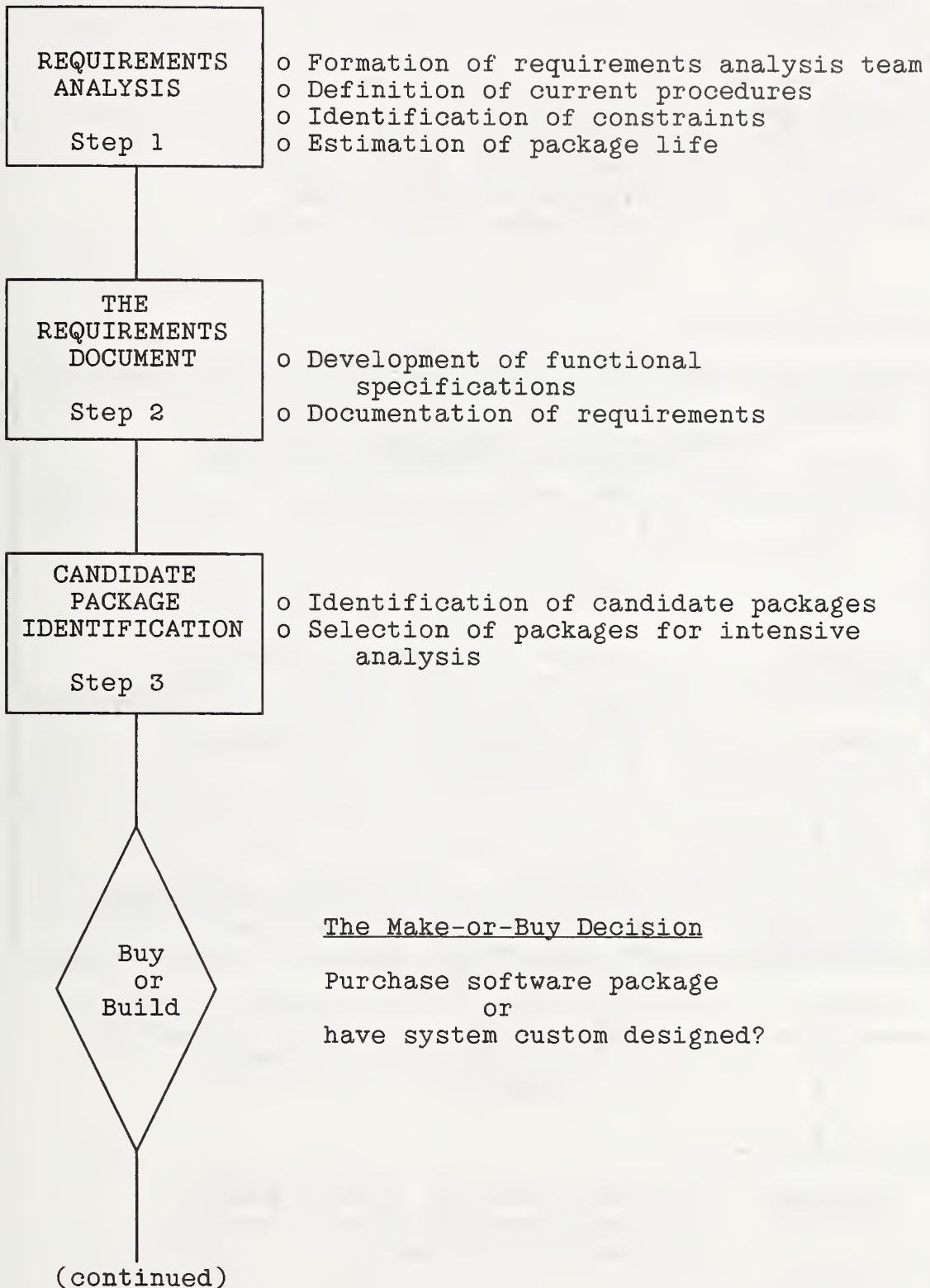


Figure 1 (page 1 of 2) - Selection Process Flowchart

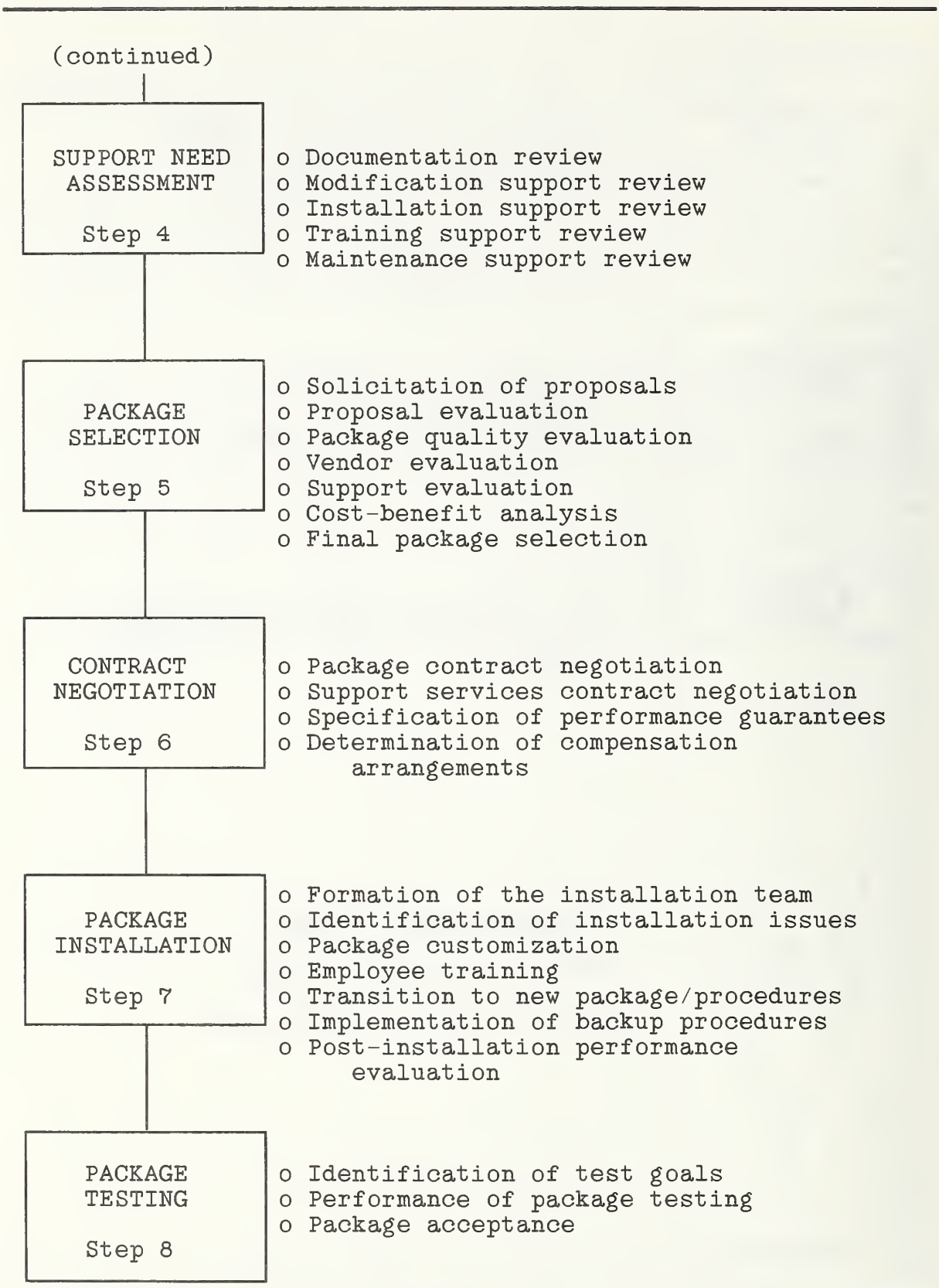


Figure 1 (page 2 of 2) - Selection Process Flowchart

STEP ONE : REQUIREMENTS ANALYSIS

1. Formation of the requirements analysis team
2. Definition of current procedures
3. Identification of constraints
4. Estimation of package life

Before the search for a package begins, organizational needs should be assessed. Evaluation of the flow of data and paperwork within an organization can help identify points of inefficiency, where a package can be most helpful. Constraints on available package options, including hardware limitations, software interfaces, and organizational procedures, should be identified.

1.0 REQUIREMENTS ANALYSIS

The largest impediment to buying appropriate software packages is that many times customers don't know precisely what they need. In order to operate efficiently, every piece of information needed should be clearly defined. Such information should be gathered and listed in the requirements document, along with a description of all required capabilities and functions.

Producing a clear and comprehensive requirements document is a critical part of the software package selection process. The requirements document becomes the basis for evaluating and comparing package options. A clear, comprehensive definition of requirements can prevent technical problems from occurring later on.

Before requirements can be committed to paper, however, they should be defined. That is the purpose of this chapter. The requirements analysis effort encompasses many tasks: researching literature for information on determining requirements for specific application areas, understanding current procedures, identifying desired changes and improvements to current procedures, and writing the requirements document.

1.1 The Requirements Analysis Team

The diverse elements of the requirements effort necessitates that the people who work on it should collectively have the following knowledge:

- o An understanding of organizational policies and procedures, and experience in making policy and procedural decisions.
- o An understanding of current hardware and software capabilities.
- o A complete understanding of the application-area tasks that are to be computerized.
- o An understanding of the perspectives, needs, and capabilities of those who will actually use the new software package.

One member of the team should be responsible for the overall requirements analysis effort. This person should also serve as the leader of the ongoing selection process. Because the choice of a software package will have many diverse effects upon organizational operations and will require a number of important operational decisions, the

leader of the requirements team should be the manager of the function(s) being computerized, or a managerial designee.

Once the requirements analysis team has been created, work can begin on software specification. The following sections are designed to assist the team in reaching a clear understanding of the organization's needs and constraints and, eventually, in producing the requirements document.

1.2 The Project File

All records, documents, and memos relating to software package specification, selection, and implementation should be maintained in a project file. This file should be accessible to all members of the team. It should contain copies of the most recent version of all materials pertinent to the requirements effort, including completed worksheets and checklists, related memos, and the requirements document. An historical project file, containing all superseded documents pertaining to the requirements effort, should also be maintained.

One member of the requirements analysis team should be appointed to take notes at each meeting. These notes should then be included in the project file. Minutes should record task assignments and the due date for each.

1.3 Defining Current Procedures

The first step in determining and specifying requirements is to develop a thorough understanding of current procedures. The process outlined here entails two basic stages. The first seeks the input of employees currently working in the application area(s). The second step involves reaching a thorough understanding of organizational operations and computerization needs.

1.3.1 Seeking organizational input

Each employee and manager who will be affected by the new package should be surveyed. One member of the requirements analysis team should be assigned to interview top management and review business plans in order to determine budgetary constraints, long-range organizational growth goals, and the amount of time the package should be expected to be useful. The results of this survey are the foundation of the entire requirements effort and should be stored in the project file.

1.3.2 Diagramming

Organizational procedures should be diagrammed in detail. The procedure described in this document involves successive creation of three types of diagrams: data flow, logical flow, and functional diagrams. Completion of the last diagram in the series, the functional diagram, provides a clear listing of inputs, outputs, and processing steps. This is helpful in writing the requirements document and will also help identify areas in which procedures could be streamlined or improved. Many managers routinely use such diagrams to gain insight into the workings of their organizations.

The following sections, which describe the diagramming procedure, are all based upon the example in Figure 2.

EXAMPLE: Current Procedures, Payroll Department

The overall tasks of a payroll department include:

1. Compensating employees for time worked/earned.
2. Maintaining internal support documentation.
3. Complying with Federal and local tax requirements.

Figure 2 - Example for Diagramming

In order to simplify the process of diagramming business procedures, each task should be divided into sub-tasks. Using the example provided in Figure 2, the following subtasks might be diagrammed:

- o Collection of timecards.
- o Computation, preparation and distribution of paychecks.
- o Preparation of payroll journal.
- o Preparation of labor analysis.
- o Preparation and filing of payroll tax returns.
- o Preparation and distribution of employee W-2 forms (at the end of the calendar year).

After each of these subtasks has been diagrammed, they should be combined to create a single diagram of the overall task.

1.3.2.1 Data flow diagrams

Data flow diagrams should identify all potential sources of package input and output and delineate the kinds of processing the data undergoes. They are created by identifying organizational outputs, delineating every input required for each output, and defining the different stages of processing required.

Data flow diagrams should use the job titles (or names) of the people in the office and the actual names or numbers of forms that are used. The procedural flow should be detailed with respect to data.

When creating data flow diagrams, use arrows and either circles or boxes. Each circle should be filled in with a person's title or name. The arrows between the circles should be labeled with the tasks that must be completed before the person in the circle to which the arrow points can begin his/her task. Arrows pointing into circles but not coming from another circle should be labeled with those outside products (for example, wage and deduction information) necessary for performing given tasks. Be careful to list all information that is prerequisite to completion of diagrammed tasks.

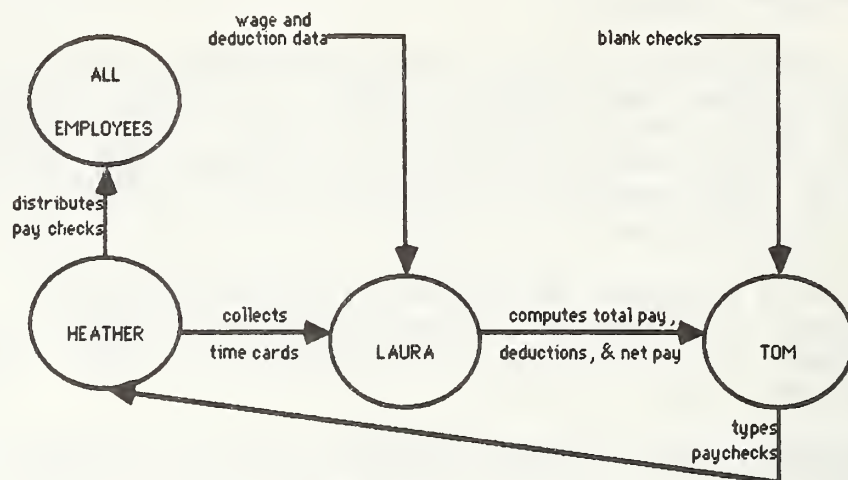
Figures 3 and 4 present three sample data flow diagrams; they represent a hypothetical procedural flow for the Payroll Department used in the Figure 2 example.

1.3.2.2 Logical flow diagrams

After data flow diagrams have been completed, they should be used to help produce logical flow diagrams. Logical flow diagrams eliminate extraneous data and show only the information required to perform a function, the processing required, and the end product.

Logical flow diagrams are particularly helpful in producing a requirements document because they detail all system inputs and outputs and describe the necessary processing. Logical flow diagrams place tasks (which were noted above the connecting arrows in the data flow diagrams) in separate circles. Original inputs and final outputs are emphasized by being placed at the beginning and end of the diagrams, respectively.

Three examples of logical flow diagrams are presented in Figure 5.



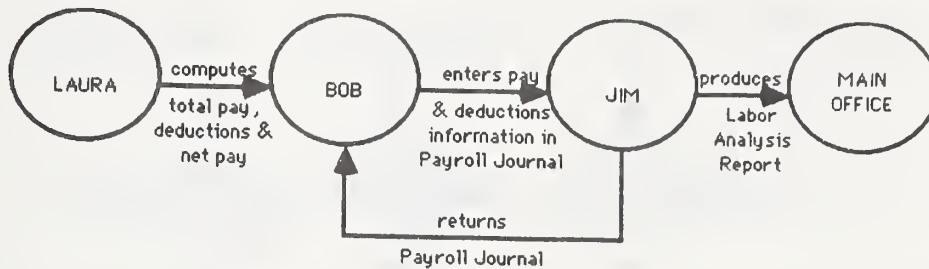
This represents a procedure in which Heather collects the time cards and gives them to Laura. Laura, using outside information on wages and deductions, computes total pay, deductions, and net pay, and then gives her computations to Tom. Tom, using blank checks, types the paychecks and gives them to Heather, who then distributes them.

Figure 3 - Data Flow Diagram

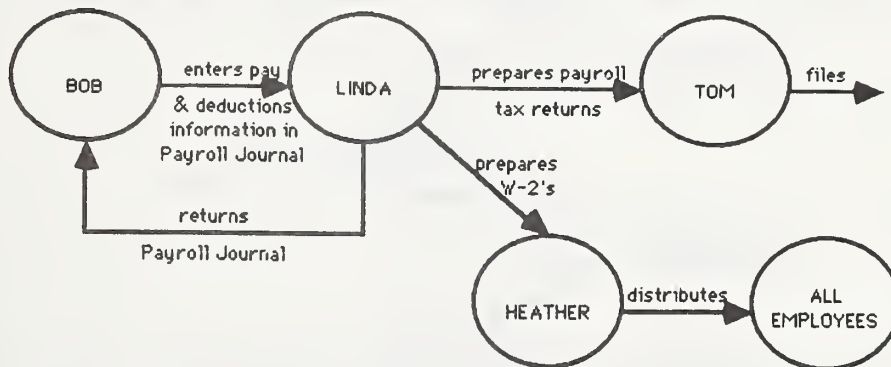
After each subtask has been diagrammed, the corresponding logical flow diagrams should be integrated to create one overall logical flow diagram for the entire application. Figure 6 is a sample overall logical flow diagram.

1.3.2.3 Functional diagrams

The integrated logical flow diagram should be used to create a functional diagram to sort out all inputs, outputs, and processing, and to define the methods of distribution. For each function in which the software package will have a place, the functional diagram will delineate the inputs and processing required for its accomplishment, the processing required to turn inputs into outputs, what actual physical output there is, and how it is distributed. In functional diagrams, different symbols should be used to differentiate distinct types of events. Standard symbols are suggested in Figure 7.



This example overlaps slightly with the one in Figure 3. After Laura computes total pay, deductions, and net pay, she also gives the information to Bob, who enters it into the payroll journal. Bob then gives the payroll journal to Jim, who uses it in producing the labor analysis report, which he sends to the main office. Jim returns the payroll journal to Bob.



This overlaps with the example above. Bob not only gives the payroll journal to Jim, he gives it to Linda, who, using other forms and instructions, prepares both the payroll tax returns and the employee W-2s. She returns the payroll journal to Bob, gives the payroll tax returns to Tom, who files them, and gives the W-2s to Heather, who then distributes them to all employees.

Figure 4 - Data Flow Diagrams

Dependent processes should be placed under those which must precede them. Indicate dependent relationships by the use of downward arrows. Arrows indicating inputs should always point into the left side of figures, and arrows representing outputs should always come from the right side.

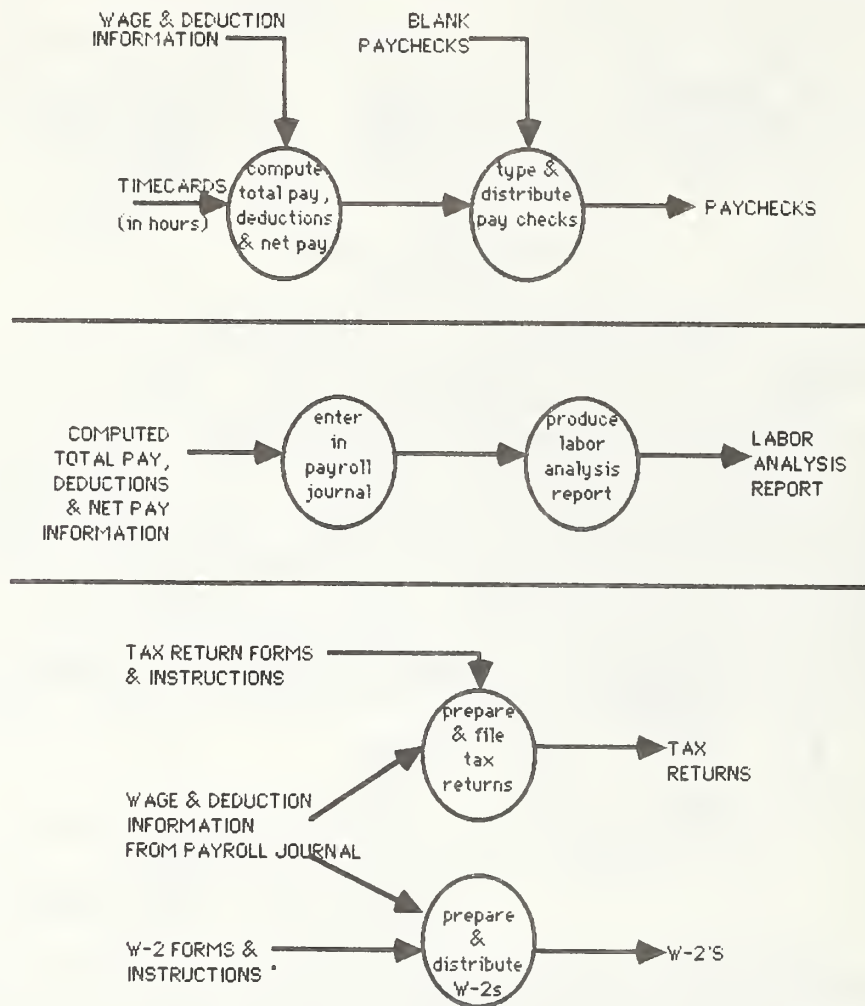


Figure 5 - Logical Flow Diagrams

Figure 8, a functional diagram of the payroll example, illustrates the usefulness of such diagrams. They simplify identification of those areas that can most profit from automation. Suppose the data from the timecards were entered into a computer package that includes printers capable of producing the W-2, payroll tax return, and required blank check form. Standard software package capabilities include computation of total pay, withholdings, and net pay (if wage and withholding information is in the package data base); preparing and printing checks; payroll journal maintenance; labor analysis; production of payroll tax returns and W-2 preparation; storing all of these types of information; and sending the labor analysis report to the main office via electronic mail.

The completed data flow, logical flow, and functional diagrams should be stored in the project file.

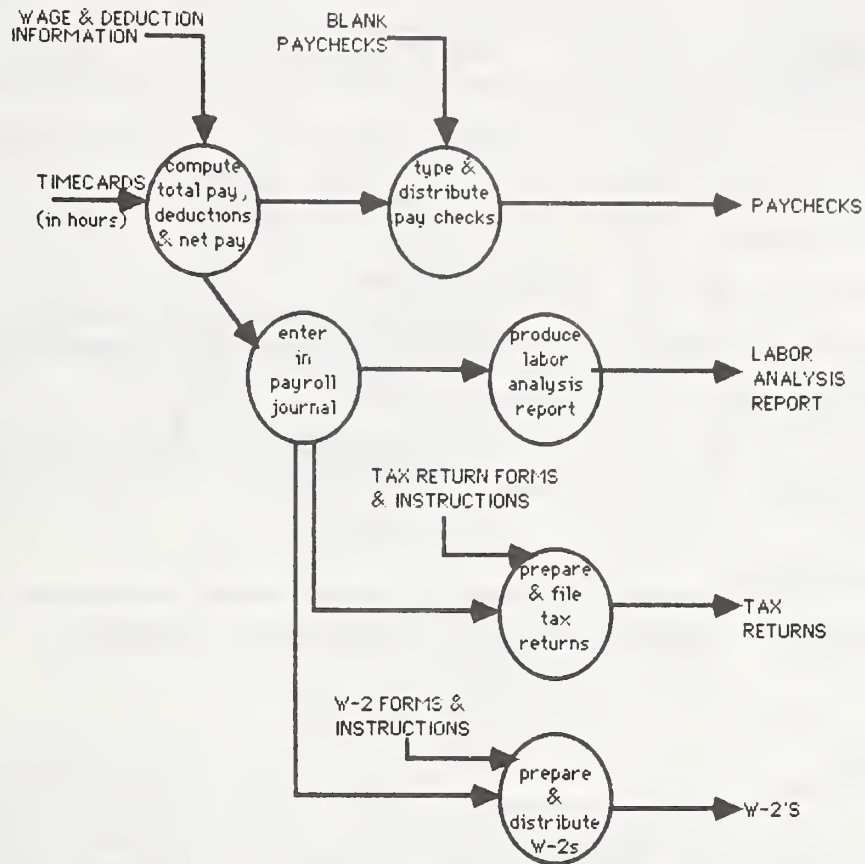


Figure 6 - Combined Logical Flow Diagram

1.4 Improving Current Procedures

Requirements definition can be a prime opportunity to improve the efficiency of office procedures. The functional diagram can be analyzed to identify potential areas of improvement. Improvements can be achieved by reducing redundancy in current processes or by the addition of efficiency-enhancing package functions, such as:

- o Automatic file sorting.
- o Global file updating.
- o Automatic data validation.
- o Query capabilities.
- o Data sharing between functions or applications.




SYMBOL	EVENT
	Inputs.
	Processes. If a process is manual, a star should be placed within or near the box.
	Outputs (paper and information).

Figure 7 - Symbols Used in Functional Diagrams

1.5 Constraints

Once organizational procedures have been defined, constraints on the prospective software packages should be identified. Software packages should conform to the hardware and software environment in which they will operate and should accommodate any organizational circumstances or restrictions that may apply.

1.5.1 Hardware environment

Hardware components should be considered when purchasing software packages because they impose potential constraints upon the software package. Each of the following hardware components can limit appropriate package options:

- o Central Processing Unit (CPU)--the main computer
- o Disk drives
- o Tape drives
- o Terminals (CRTs and keyboards)
- o Printers
- o Total available computer memory.

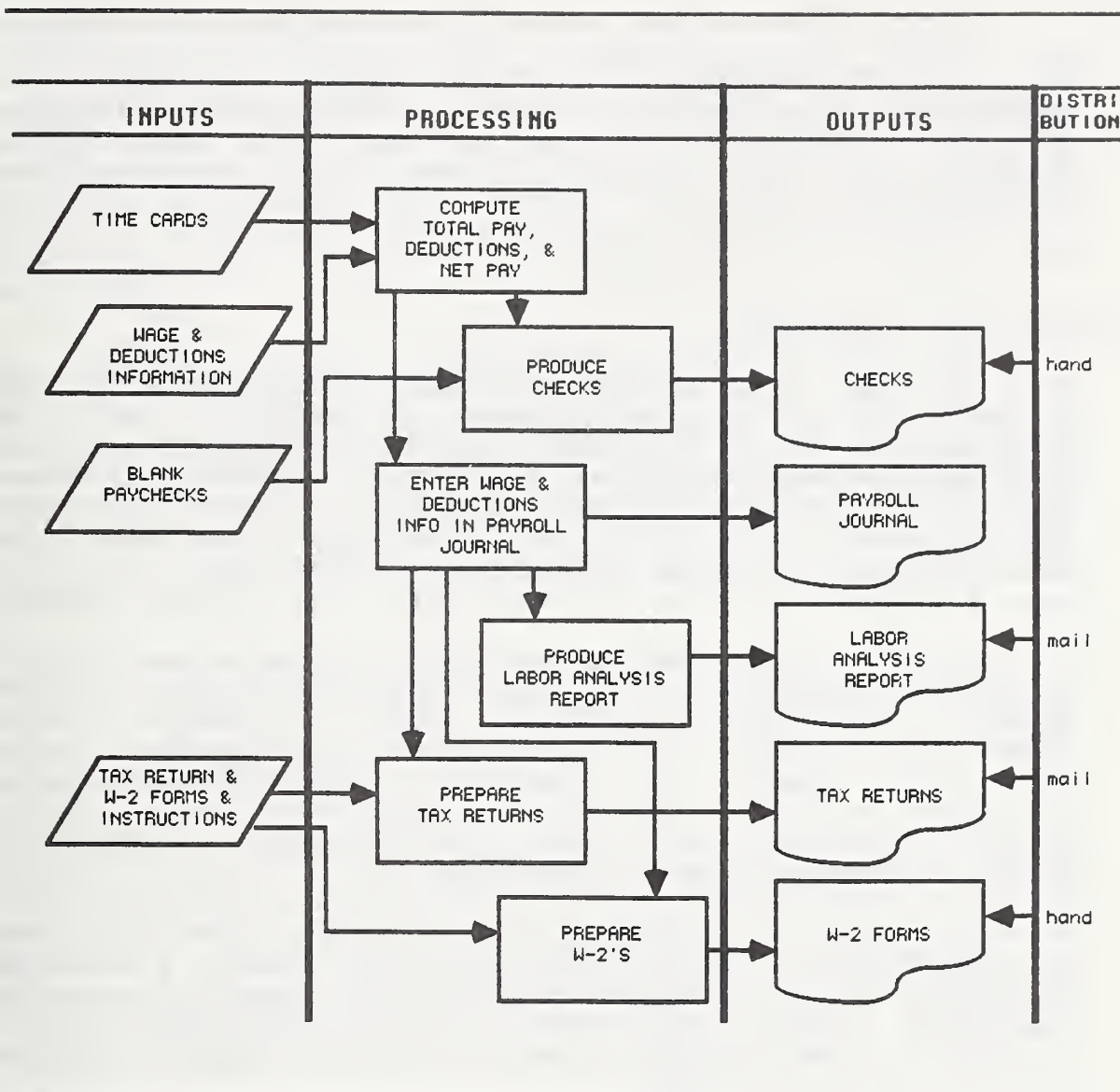


Figure 8 - Functional Diagram

If computer resources will be shared with other groups, this should be taken into account.

When considering whether to buy new hardware, it is important to remember that--although it may be expensive--buying new hardware can be more cost-effective in the long run than using outdated or ill-designed hardware. In most applications, the cost of hardware is far exceeded by software costs over the system's life. The cost of any new hardware required by candidate packages will be accounted for in the final cost-benefit analysis guiding package purchase.

If both hardware and software will be purchased (as in an initial computerization or total upgrade), it is usually best to find an appropriate software package first, and then purchase hardware that can run it. Hardware and software act as mutual constraints on one another; but the software comprises the most important portion of a data processing system.

1.5.2 Cost constraints

In most organizations, a major constraint on the purchase of an appropriate software package is cost. The most useful way of assembling a computer system is to determine the maximum budget beforehand. Software, hardware, and ancillary budgets can then be set at levels that make optimal use of the available resources. Sometimes, it is less expensive to buy an entirely new system than to try to make a software package run on an old, inappropriate system.

Section 5.6 on cost-benefit analysis can be used to make a rough determination of the cost savings that can be expected from the software package purchase. When calculating package cost, the costs of anticipated benefits from the package (for example, increased efficiency, increased accuracy, increased productivity, etc.), and the costs of maintenance, training, modification, and other necessary support should be included.

The goal of the package search is to find the most efficient way to satisfy organizational needs. A realistic assessment of the costs and benefits of computerization is essential. The assessment should also include consideration of budget, organizational resources, and the impact of growth on data processing needs.

1.5.3 Other constraints

Constraints such as the delivery time of a software package, its expected life, and the availability of in-house maintenance personnel all help determine the type of package that is most appropriate. Although these constraints are not functional requirements, they do belong in the requirements document and will serve as evaluation criteria when selecting a software package.

1.6 Estimating Package Life

In order to choose a package appropriately, it is important to estimate how long it will be expected to remain useful. Although it is not subject to wear and tear in the way that capital equipment is, a software package can become obsolete as a result of changes in the application it supports, changes in the software and hardware with which it interfaces, or changes in organizational needs and procedures.

Knowing how long a package will be needed is essential to making a cost-effective comparison of different data processing alternatives. The benefits of including a function, or group of functions, will be spread over the time a software package is used. Comparing different options includes comparing the costs and benefits a package will yield after it is purchased, as well as estimating the long-term efficiency savings and maintenance and training costs.

The primary factor influencing the useful life of a software package is the anticipated change in the application it will support. A certain degree of flexibility is designed into most packages, but there is some point at which change cannot be economically accommodated.

Organizational growth is one example. A package is usually selected with some growth potential in mind. Data handling capacities and response times are degraded as system load increases, and the memory capacity of a system can be exceeded. In fact, this is the most common major maintenance problem associated with software packages. By estimating how long a package is likely to be used, the maximum amount of required growth potential can be estimated.

If anticipated policy or procedural changes will have major effects on an application, the life of the package may be shortened. The estimated life of the package should not exceed the period that such changes can be reasonably predicted. The likelihood of significant changes in the duties, values, or capacities of the organization will serve as constraints on the package selection.

The projected life of a software package should be long enough to justify the initial expenditure. If the efficiency savings from the package will not be enough to justify the purchase price, it is probably not an appropriate purchase. If major organizational changes are likely in the near future, it is often best to postpone selection of a new software package until the changes have taken place.

STEP TWO : THE REQUIREMENTS DOCUMENT

1. Development of functional specifications
2. Documentation of requirements

The organizational and functional requirements should be precisely stated in terms of specific package characteristics. A format is presented for the requirements document, which will be used to communicate requirements to vendors and to help evaluate different package options.

2.0 THE REQUIREMENTS DOCUMENT

2.1 Purpose Of The Requirements Document

The next step in selecting a package is producing a comprehensive requirements document that describes, in detail, the characteristics that are required from the package. This document should define what inputs and outputs the package must handle, what kinds of processing are required, what reports the package is expected to produce, and all other important details of package performance.

Production of a requirements document is a crucial part of package selection. It allows software package needs to be precisely considered and identified and relates those needs clearly and completely to potential package vendors and developers. The additional expense and trouble caused by installation problems of an inappropriate package can be mitigated or eliminated by using a carefully drafted requirements document.

The requirements document serves a number of important functions, which are described below. Table 1 lists these functions.

Table 1 - Functions of a Requirements Document

- o Basis for comparison of different packages
 - o Clarity of requirements
 - o Achieving internal agreement on package needs
 - o Program design specification
 - o Identification of the criticality of different aspects of the application
-

Basis for comparison of different packages. Because the requirements document describes the needed software package completely, it can be used as a standard to compare different packages. The requirements listed in the document can be used as package selection criteria. The relative strengths and deficiencies of different off-the-shelf packages can be compared by evaluating each with respect to the requirements document.

Clarity of requirements. The requirements should be clearly, concisely and completely described, in order to evaluate different package options.

Achieving internal agreement on package needs. Very often the choice of a software package will help determine future organizational methods and priorities. Thus, all major characteristics of the package need to be understood. The requirements document can facilitate this understanding. In producing a document that comprehensively describes application needs, organizational priorities and needs should be defined as well.

Program design specification. If a package needs to be modified or if software must be custom-designed to meet application needs, the requirements document can serve as a definitive guide for programmers. Functional needs are clearly set out in a way that minimizes the chance of misunderstanding, reducing the organizational costs associated with inappropriate design or repeated rounds of package use and modification.

Identification of the criticality of different aspects of the application. The methods of needs analysis described in the previous chapter are designed to facilitate understanding of the ways that different application tasks relate to one another. Dependent processes are identified so they will not be neglected and the inter-dependencies of different application tasks will be clearly understood.

2.2 Components Of The Requirements Document

The requirements document is an orderly presentation of all essential information about the required software package. It should be written in such a way that it may be easily used as a reference guide. It should state which requirements are vital, which are important, and which are simply preferred. Table 2 lists the types of information that should be included in the requirements document.

The following sections discuss those aspects of package performance and functionality that the requirements document addresses. For the sake of illustration, a payroll software package is partially described. The examples are not complete, and are offered only as illustrations of package requirements specifications.

2.2.1 Functions

The requirements document should describe in detail every function the package should perform. The diagrams made during requirements analysis should prove helpful. Interconnections between functions should be identified.

Table 2 - Types of Information Found in Requirements Document

Functions
Outputs
Inputs
User interface
Technical characteristics
Documentation
Training Services
Installation Services
Maintenance services

For every computerized function, each component task should be identified. These descriptions should not prescribe a single way of performing or displaying something, unless there is a genuine application-related rationale why they can be done only that way. Instead, they should provide detail on what the package should do, rather than how it will be done. For instance, if payroll is to be done by the package, each related subfunction should be briefly described. Figure 9 is an example of such a description.

2.2.2 Outputs

Required output formats should be stated unambiguously. Each output should be described in terms of its length, format, minimum and maximum values, and whether it is to be externally displayed. This information should be compiled into table form. Figure 10 is an example of such a table. Any abbreviations or designations should be explained. The description of each output item should include:

Length. The maximum length (in number of characters) of every output should be listed.

Description. The format and possible values of every output should be detailed.

Internal or external. Not all outputs will be directly available to package users. Some outputs will be inputs to other computerized processes. Even though they are "unseen," it is important to describe intermediate outputs, and to designate them as such. This is particularly important if a software package is to be integrated with other software, whether it is pre-existing or anticipated.

Paycheck Production

- o Collect employee time reports
- o Tabulate total hours
- o Wage lookups
 - Determine overtime eligibility
 - Calculate overtime
- o Deduct sick time and vacation time
 - Maintain eligible hours log
 - Disallow if ineligible
- o Calculate base pay
- o Determine appropriate deductions and subtract from base pay:
 - Voluntary deductions (savings bonds, etc.)
 - Stock ownership
 - Medical plan
 - Life insurance
 - Social Security
 - FICA
 - Federal income tax withholding
 - State income tax withholding
 - Local, commuter, or other taxes
 - Disability/unemployment insurance
- o Deduct pay amount and any other expenses from company ledger
 - Social Security contribution
 - Unemployment insurance
 - Company contribution to insurance plan(s)
 - Company contribution to unemployment insurance
- o Print paychecks
- o Print ledger

Figure 9 - Sample Function and Subfunction Description

INPUT NAME	# CHAR	FORMAT	LO VAL	HI VAL	SOURCE	COMMENTS
Employee number	5	I-nnnn	A-0001	Z-9999	Operator	
Employee name	40	(40) I	A	(40) Z	Operator	Last name, first name, MI
Wage rate	9	nnnnn.nnl	0000.01H 0185.00S	0035.00H 1250.00S	Software	H designates hourly wage S designates weekly salary
Federal wh rate	4	.nnn	.000	.999	Software	Rate table must be revisable
State wh rate	4	.nnn	.000	.250	Software	Rate table must be revisable
Social security tax	5	.nnnn	.0000	.0129	Software	Rate must be revisable
Job classification	10		-no text-		Software	Must be existing classifictn
Voluntary wh allowance	4	nnn.	000.	350.	Software	Default value = zero
Federal tax wh to date	10	nnnnnn.nn	000000.00	65000.00	Software	
Total wages to date	10	nnnnnn.nn	000000.00	65000.00	Software	
OUTPUT NAME	# CHAR	FORMAT	LO VAL	HI VAL	EXTRNL?	COMMENTS
Social Security Number	11	nnn-nn-nnnn	000-00-0000	999-99-9999	Y	
Employee number	6	I-nnnn	A-0001	Z-9999	Y	Letter used to desig. dept.
Net wage this period	8	\$nnnn.nn	\$0000.00	\$1250.00	Y	Leading zeros not printed
Federal withholding	8	\$nnnn.nn	\$0000.00	\$1250.00	Y	Leading zeros not printed
Fed. withholding rate	6	nnn.nn	100.00	0.00	N	Shared with acctng software
Overtime allowed?	1	I	see comment	see comment	N	Values must be "Y" or "N"
YTD gross earnings	10	\$nnnnnn.nn	\$0	\$65,000.00	Y	Leading zeros/sps not prntd
Cum. hrs vacation time	6	nnn.nn	0.00	360.00	Y	Leading zeros not printed
Vacation hours used	5	nn.nn	0.00	40.00	Y	Leading zeros not printed
Cum. hrs sick leave	6	nnn.nn	0.00	80.00	Y	Leading zeros not printed
Sick hours used	5	nn.nn	0.00	40.00	Y	leading zeros not printed
Total overtime hours	5	nn.nn	0.00	99.99	N	Used for acctng software

n = Integer number
I = Alphabetic character

Figure 10 - Sample Input and Output Tables

Minimum and maximum values. The minimum and maximum value that each output could take should be precisely stated in the requirements document. Potential organizational growth, future inflation and the like should be taken into consideration when calculating maximum and minimum values.

2.2.3 Inputs

As with package outputs, package inputs should also be described precisely. Field formats and minimum/maximum values should be designated. Internal inputs should also be included. These include inputs that are the output of other software, as well as values that the package will have to look up in order to complete a calculation or operation (e.g., withholding rates or year-to-date totals).

A separate listing should be made of each package output, and all inputs required to generate it. This will help to insure that the input and output listings will be comprehensive and that none are neglected. It will also help guarantee that the package calculates data in the same way(s) as are organizationally required. The data flow diagrams described in the previous chapter should help in making this listing. Figure 10 has an example of an input listing.

2.2.4 User interface

The ease with which non-technical users can learn and operate a computer package differs greatly from one package to another. Many software packages, particularly those of more recent design, are designed to be user friendly, or relatively easy to operate. Many of these packages make it more difficult to perform irreversible operations (for instance, deletion of a file), and attempt to mitigate the results of many operator errors that do occur.

Some packages are menu driven, allowing users to operate the package by choosing from among a number of options presented. Selection of one option will result in the display of appropriate sub-options, and so on. Users do not have to memorize a great number of commands, and training needs are thus reduced. Menu-driven packages are recommended in cases where there will be a large number of infrequent package users, the package is highly complicated, or employee turnover rates are high.

Many packages offer on-line help to users in performing computerized operations. These help facilities may indicate command formats, explain computer features, or may provide line-by-line response prompting when requested. They can be very helpful in performing unfamiliar operations.

Some packages offer error-checking capabilities for data that is entered. Depending on the level of a package's sophistication, error-checking can take a number of different forms:

- o Input values can be checked against established minimum and maximum values.
- o Input values can be checked to insure that they follow established formats.
- o Input values can be checked against historical data to insure that the entered value is reasonable.
- o Input values can be checked against partially redundant information already entered (e.g., Employee Name, and Employee ID Number) to confirm that they are logically consistent.

Such user-friendliness is not without its price, however. User-friendly features usually consume computer processing time, so package response time may be reduced. Often, user-friendly features can become time-consuming and tedious to a heavy user who must perform a number of similar operations: a user may have to go through several menus to accomplish what could have been done with a single command on a less "friendly" package.

Requirements analysis will have to evaluate the usefulness of such user-friendly features. Desired user-friendly features should be specified in the requirements document. Other than those general features described above, most are application-dependent. Review of package literature and advertisements in the application area should provide guidance as to what features are available. User-friendly features should not be stated as absolute requirements unless they are essential. However, it can be helpful to indicate which features would be particularly useful.

2.2.5 Technical characteristics

Some minimal technical package characteristics should be included in the requirements document. Technical specifications that are appropriate in a functional requirements document include reliability rates and response times.

Reliability is usually expressed in two ways. Mean-Time-Between-Failure (MTBF) is the average time a system is operational between malfunctions or errors. This includes

one-time spurious errors. Mean-Time-To-Repair (MTTR) is the mean time between the identification of a failure and its repair. It is best to check software literature in the application area to determine reasonable values for these two characteristics.

Response time, another useful package characteristic, is the time the package takes to execute a command or procedure. It is usually measured under a controlled set of circumstances, which differ depending upon the hardware, how many users will be on the computer at once (and what they will be doing), what kind of computerized operation is being considered, etc. It is best to state the response time required under worst-case conditions so the vendor or package developer will understand what is unacceptable. Otherwise, there are too many variables to consider, and real package use will seldom duplicate the precise set of circumstances detailed under the response time characteristics.

Maximum acceptable response times should be defined for the following operations (given a specific hardware configuration):

File Inquiry. How long will it take for the package to find the information contained in a particular file once an inquiry has been made by a user?

Terminal Echo. When a character is typed on the keyboard, how long will it take for the package to acknowledge the character and print it on the screen? (If display lags much behind entry, user mistake rates will be higher, and mistakes will take more time to correct.)

Report Generation. How long will it take to generate a required report?

File Updating. How long does it take to revise a file? Sometimes, to change a single character in a single file, all related files must be copied into an internal working space and back. If there are many files, and minor, occasional file revision is a norm, this can be a very important consideration.

File Sorting. (if applicable) If files must be arranged in a particular order, or in different orders, the package may have to sort them. If the number of files to be sorted is high, this may be extremely time consuming. The maximum acceptable sorting time should be included in the requirements document. A reasonable value can be determined from a review of software literature, and consideration of the number of files (and kinds of information) that must be sorted. If other computerized operations must be able to take place at the same time, this should be noted.

File Searching. Computer packages are often able to search a file, or group of files, for the occurrence of a certain string of characters or a particular set of values. The time required to search may be important, if this is done often. Again, software literature from the application area is the best guide to selecting appropriate response time values.

2.2.6 Documentation

All required package documentation should be listed and briefly described. Section 4.1 discusses the different kinds of documentation.

Documentation should:

- o be comprehensively indexed.
- o be written at an appropriate level of technical complexity for intended readers.
- o discuss, in adequate detail, every command and feature that will relate to application tasks.
- o be systematically organized.
- o be updated to reflect any customizations or changes made to the package.

2.2.7 Training services

Every software package requires some degree of training to be properly used. For small, simple packages, a training manual is often all that is needed. For larger ones, training becomes a more critical consideration. An excellent and sophisticated software package may be no more useful than a mediocre one unless users know how to operate it efficiently. Training should be appropriately geared to the complexity of the package and the computer proficiency of the users. Chapter 4 discusses training need evaluation in more extensive detail.

The requirements document should indicate which user groups will need to be trained, their current degree of computer familiarity and sophistication, and what operations each user group will have to perform on the computer. It should also indicate whether training will be needed for technical personnel. Required training manuals should be listed by functional type (see Chapter 4). If organizational changes are to be introduced when the new package is installed, this should be included as part of the training.

2.2.8 Installation services

In general, the vendor or developer who provides the new package will also take responsibility for installing it. If the new package is to interface with other software or package customizations are to be made, this may no longer be true. The requirements document should have a brief section identifying who will perform the installation. Chapter 4 has an extensive discussion of installation issues. If an installation will be nonstandard in any way, the reasons should be explained in the requirements document. If data conversion services will be required, the volume of data requiring conversion and its present format should be specified.

2.2.9 Maintenance services

The requirements document should indicate whether package maintenance will be done using in-house resources, by the package provider, or by some other party. If it will be done in-house, and source code is required to do it, this should be indicated. Different kinds of maintenance services are discussed in Chapter 4.

2.3 Requirements Document Format And Content

The requirements document should be logically organized and easy to reference. This section suggests a coherent format for a requirements document, but if another layout is more appropriate, it may be used. The document should include all the information described in this chapter. Table 3 summarizes the suggested format.

Illustrations and examples should be used, particularly to illustrate output and report formats. Required forms should be appended to the document.

Table 3 - Sections of the Requirements Document

System purpose
Current procedures
Hardware and software configuration
System scope
Functional requirements
Reports
Data dictionary

2.3.1 System purpose

The requirements document should be prefaced with a short section describing the organizational application needs that are to be addressed; identifying if the package will be required to interface with other software; and briefly stating organizational data processing goals.

This section need be no more than a couple of paragraphs long. It should simply relate the basic reasons for the package search.

2.3.2 Current procedures

Current application procedures should be described in this section. Organizational information regarding the data flows and logical flows of application area procedures should be included. The point at which data should be intercepted or approved by organizational staff members should be identified. Data flow and logical flow diagrams made during the needs analysis stage may be included.

This section should include a description of all functional areas that the package will incorporate. If previous procedures were also computerized, a brief description of computer system capabilities should also be included.

2.3.3 Hardware and software configuration

The current hardware and software configurations should be described and diagrammed, if they will be used with the new package. The following information should be included in this section of the requirements document:

Present hardware components. Each hardware component of the present computer system, or system that is to be used with the package, should be listed. The brand name and model number of each component should be specified:

- o Computer
- o Disk drives (hard and floppy) and their capacities
- o Tape drives and their availabilities
- o Terminals
- o Printers

Configuration diagram. The ways that the hardware components are connected together should be graphically represented in a configuration diagram. The diagram should indicate which components have dedicated uses and which are always on line, what components can be accessed by what other components, and the brand and model names of each component.

Possible hardware changes. If the purchase of new equipment to help support the package is under consideration, those contemplated changes should be described. It is often best to leave this open-ended, where possible, so the hardware chosen can be optimally suited to the capabilities and characteristics of the package. If this is not possible (because the hardware must be selected to suit other software or for any other reason), the brand names and model numbers of new hardware should be specified. If the configuration of new hardware has already been determined, an additional hardware configuration diagram should be included.

The current software configuration should be represented in much the same way. If the new package will have no interfaces and share no data with other software on the computer, the other software may simply be listed. The dedicated memory required by each piece of software on the system should also be indicated. The operating system, and its version number, should be specified.

If information will be shared between different software on the computer system, an interface control document will have to be included for those items of information that will pass between them. It is beyond the scope of this document to describe how to construct such a document; a computer consultant or data processing expert should write it. If the same files will be processed by more than one software system, and a data base management system accesses those files, the brand, name, and version number of the data base management system should be included, along with its system parameters. A data processing expert will usually be needed for this as well.

2.3.4 System scope

In order to insure that the software package will be of appropriate capacity to meet application needs, the number of files or information of each class that will be handled should be specified. This should be done for each separate type of file that will be maintained. Figure 11 suggests a format for relating such information.

A separate section should estimate the maximum number of times each kind of package operation will be done per week or month. Figure 12 is an example of such an estimate.

FILE TYPE	# FILES	FILE COMPONENTS
Payroll	800	Employee number Employee classification Base rate of pay Date of last pay raise Hire date Paycheck routing address Employee division number Address Phone number YTD gross earnings YTD net earnings YTD Federal income tax paid YTD State income tax paid YTD voluntary withholding Accumulated sick leave Accumulated vacation time
Accounting	20	Accounts receivable (10,000) Accounts payable (10,000) Payroll expenses (21 divisions x 52 weeks, plus total) Capital expenditure (20 divisions x 52 weeks, plus total) Capital plant maintenance (x 52 weeks, plus total) Liquid assets (x 52 weeks)

Figure 11 - File Scope Chart

OPERATION	# OF TIMES/MONTH
Paycheck production	3,500
General ledger update	20
Account inquiry	40,000
File search	200
Management analysis	40

Figure 12 - Estimate of Number of Operations

2.3.5 Functional requirements

The functional requirements section is normally the longest and most important part of the requirements document. It should describe each subtask in every application area that will be processed by the package; the manner in which data is to be processed, explicitly if it is at all nonstandard or ambiguous; and all required or desired package features. Figure 13 is a sample of part of a functional requirements description.

In describing functional requirements, particular attention should be paid to the following:

Who has access to what information. The requirements document should specify what kinds of information, and what functions, may be accessed by different classes of users. Private file information should remain private. Most software packages have provisions for restricting access to certain files or kinds of information.

All interconnections between functions. If data entered when performing one function must be accessible to another function this should be specified.

Look-up values. If an input for an operation will be "looked up" from registers or tables maintained within the package, it should be indicated whether these values need to be easily accessed or changed by the application manager. For instance, tax rate tables, employee overtime eligibility status tables, and the like, will probably be changed frequently enough that it would be overly burdensome to require a programmer to change these values when required.

Historical data. Specify how long historical data should remain accessible on-line.

Additional features. Each feature that is not directly related to a particular application function should be listed separately. These include such considerations as the degree of user-friendliness required, ease of package start-up, and other incidental features.

Audit trail requirements. If transaction records need to be maintained, the level of detail required, and the relevant data, should be specified.

Package Security

Access to payroll information shall be restricted to managers and payroll data entry clerks only. Access may be restricted through use of a password or by other means. Access to internal accounting information shall be restricted to only those computer terminals in the accounting office. Login to the computer package will be allowed only to specifically authorized employees. Passwords will be available only to the department manager, and will not be echoed back to the terminal when entered. These are absolute requirements.

A record will be kept of which employee has revised each file, and the date of revision. This data will be retained for three pay periods. Records will also be maintained of which files have been printed into hardcopy form or copied onto another data storage medium, either partially or wholly, and by whom. These requirements are strongly preferred.

The package will allow employees to change their own passwords at will, without changing their access privileges. This is a preferred requirement.

Figure 13 - Sample Functional Requirement

2.3.6 Reports

This section should contain a description or facsimile of each report that the package will produce. This should include both printed and on-line reports. If format is important, a facsimile report should be included. If the package should have the capability of using current forms, a full-sized copy of each form should also be included.

The sample forms (or form descriptions) should indicate every data field that should be included, as well as the format and position of each.

2.3.7 Data dictionary

This section should detail, as described in Sections 2.2.2 and 2.2.3, each package input and output. For each input and output, the following should be specified:

- o The length of the data field.
- o The format of the data field, including:
 - Character type.
 - Whether the decimal point (if any) is floating or fixed.
 - Whether leading zeros, and those trailing the decimal point, will be displayed (outputs only).
- o Whether an input value comes from a user input or from calculations performed by the package.
- o Whether an output value will be displayed or simply become an input for another operation.

Figure 14 presents samples of sections from input and output listings. Every functional input and functional output should be accounted for. A legend should accompany each listing, detailing the symbology used.

For the input listings, the source of the input data should be indicated. Inputs can be directly entered by the user; taken from internal values stored by the package; generated by another software process or subprocess; or taken from stored, historical information on disk, tape, or other storage medium. Inputs should be traceable to one of these sources.

OUTPUT	INPUTS
Social Security number	Employee name (o) Employee number (o) Social security number (sw)
Employee number	Employee number (o)
Gross weekly pay	Wage rate (sw) Employee type (sw) Overtime eligibility (sw) # of hours worked (o) Sick pay.. (sw) Vacation pay.. (sw)
Net weekly pay	Gross weekly pay.. (sw) Federal tax paid.. (sw) State tax paid.. (sw) Voluntary withholding (sw) Social security tax.. (sw)
Federal tax paid	Gross weekly pay.. (sw) Federal tax rate.. (sw)
Federal tax rate	Gross weekly pay.. (sw) Federal w/h allowance..(sw) Taxable entity check (sw)
YTD Federal taxes paid	YTD Federal taxes paid (sw) Federal taxes paid
Voluntary withholding	Voluntary withholding (sw)
YTD voluntary withholding	YTD voluntary w/h (sw) Voluntary withholding (sw)
Accumulated sick time	YTD sick time (sw) Sick time earned this period..(sw) Sick time used this period (o)
Sick time used this period	Sick time used this period (o)
State tax paid	Gross weekly pay.. (sw) State tax rate.. (sw)
State tax rate	Gross weekly pay.. (sw) State w/h allowance.. (sw) Taxable entity check (sw)

o=operator-entered input;sw=software-defined; ..=secondary input

Figure 14 - Example of Output-Input Listing

STEP THREE : IDENTIFICATION OF CANDIDATE PACKAGES

1. Identification of candidate packages
2. Selection of packages for intensive analysis
3. The make-or-buy decision

The first step in package selection is to identify appropriate commercially available software packages and determine whether they are able to fulfill all requirements. If appropriate off-the-shelf packages are available, a limited number are chosen for intensive analysis. A decision should then be made: whether to purchase one of the packages or to develop custom software.

3.0 IDENTIFICATION OF CANDIDATE PACKAGES

The decision whether to buy a software package or to have software custom developed can be a difficult one. First, it should be determined whether appropriate packages are available. Then, potential candidates should be identified.

3.1 The Selection Team

Careful package evaluation requires a diversity of perspectives. Final selection should include analysis of technical, organizational, and human interface issues to make sure that the software package selected is satisfactory. A selection team should be formed that incorporates people who can fulfill each of the following roles:

Selection team leader. The head of the selection team should be the functional manager of the application. If a number of different functions will be computerized at once with an integrated package, the manager of each function involved should be included on the team, with one chosen to be the team leader. The team leader will organize the package evaluation, assign personnel to perform various evaluation tasks, and oversee scheduling.

Technical advisor. Technical issues should be reviewed by someone familiar with both data processing and the application area. The technical advisor will evaluate package documentation, review requirements, check specifications of candidate packages, and advise other team members on technical issues.

User personnel. The selection team should include people from each class of potential users of the new package. If clerks, record management personnel, secretaries, and managers will all be using the new package, at least one person from each category should be included on the selection team. Users have an intimate understanding of day-to-day workings of their departments and can often offer astute insights about how well a particular package can be integrated.

Requirements effort leader. Generally, the head of the requirements analysis team should lead the selection team as well. If not, the leader of the requirements analysis team should be included, to help resolve selection issues. If a package does not meet a particular requirement, for instance, the selection team cannot determine whether an alternative feature is a fully satisfactory substitute unless the rationale behind the original requirement is fully understood. The leader of the requirements effort should be able to illuminate such issues.

Procurements/contracts personnel. Software packages are usually licensed, not sold, and are subject to copyright laws. Licensing contracts are often very complicated and technically involved. They should be thoroughly evaluated by legal and procurements specialists.

3.2 Identification Of Candidate Packages

The next step in the selection process is the compilation of a list of candidate packages, which tentatively satisfy the hardware requirements and the most important functional requirements. "Introduction to Software Packages", NBS Special Publication 500-114, discusses candidate package identification in greater detail. Table 4 lists the major sources of information about software packages.

Table 4 - Sources for Candidate Package Identification

Application area journals
Data processing magazines
Software package compendiums
Other application-area package users
Software package search firms
Computer consultant

In evaluating the options, all available software packages within the broad applications category should first be identified, using any resources that prove helpful. Many data processing and application area magazines offer comparisons and user ratings of different packages. If other divisions of an organization have the same application, they may be valuable--and frank--sources of information about the package they use.

Simple identification of a package does not provide enough information for evaluation. The next step is to obtain literature from the developer or distributor of each package that has been identified. The usefulness of package literature varies widely: some brochures are quite specific about package features and capabilities, while others are not. However, the literature should provide at least one very valuable piece of information: the name of a local vendor of that package.

3.3 Narrowing The Field

Before intensive analysis and comparison of candidate packages can begin, the number of packages should be narrowed down to a manageable number: no more than three to five candidates, which will then be closely analyzed. For common applications, there is a wide variety of software packages available--far too many to consider closely. Even for those application areas in which there are only a few appropriate packages, a brief evaluation can help to identify packages that do not merit closer analysis.

The first step is to verify that each package fulfills the essential requirements; any package that does not, should be eliminated from further consideration. The following criteria should also be used to eliminate clearly inappropriate packages:

- o Hardware incompatibility.
- o Operating system incompatibility.
- o Requires too much dedicated memory.

The remaining requirements should then be used to further narrow the field. This is done using less and less critical requirements until all but three to five software packages have been eliminated. The literature for each package that has been rejected should be retained in the project file, along with a short explanation of why that package was not suitable. This can be particularly helpful if intensive analysis proves that none of the finalists are appropriate, and previously rejected packages have to be reconsidered.

3.4 The Make-or-Buy Decision

Generally speaking, there are three ways of meeting software needs: a package can be purchased, software can be custom developed, or a hybrid "customized" off-the-shelf package can be developed. Of the three alternatives, off-the-shelf packages are often the least expensive, the most reliable, and require the least time to implement. Their performance and the quality of their documentation and training programs can be known beforehand, from vendor information and other package users, so that there are less likely to be any significant surprises after implementation.

Custom development may be unavoidable if an application is unique or uncommon capabilities or features are required. It has the advantage of being able to meet all requirements,

but often requires extensive debugging and requires a long lead time to implement--a complex program can take years to develop. Custom software is generally more expensive than an off-the-shelf package.

Customization of packaged software has some of the advantages and disadvantages of each of the other approaches. It is most appropriate in cases where input or output formats of a packaged program are not appropriate to application requirements, or additional reports need to be generated from data stored by the package. A rule-of-thumb offered by many people with software package experience is: if more than 10 - 15% of the package needs to be modified, it is generally more cost-effective to develop the system in-house.

Commercial software packages are copyrighted, and firms often place restrictions on the packages they sell or lease, including restrictions on modifications. If a package developer is unwilling to release the source code, internal modification may not be possible. If consideration of a package is contingent upon modification, check with the vendor of that package to make sure that modifications may be made. It is important to note whether changes made to the source code of a package voids or alters any package warranties or service agreements.

3.4.1 Cost accounting

A cost-benefit analysis should be performed to determine whether the purchase of a software package is the most cost-effective option. The following costs should be taken into consideration:

Personnel costs. Personnel costs are incurred by an organization through expenditure of employee time. This includes the time it takes to train employees to use the new package and the time to negotiate the purchase or development contract. If software is developed in house, there are additional personnel expenses. The costs for development of the software, its training program, and documentation should all be accounted for. If an organization is billed by an in-house data processing unit, expenses for development, documentation, and training should be computed as though they were purchasing costs and the task was contracted. If the software is purchased, or developed outside, these expenses would be detailed under "purchasing costs."

Personnel costs are not simply the wages paid to employees, but also all overhead costs. Personnel costs should be carefully considered.

Purchasing costs. If a package is bought or leased, the purchasing cost is simply the price of the software, documentation, training, and modifications to the program. Vendors may offer all these services together or "unbundle" documentation and training from the software and sell services separately. If multiple copies of the package are needed, the vendor may offer site licenses as an alternative. Vendor support services are discussed in detail in Chapter 4. If the program is custom developed, purchasing costs include purchases ancillary to development and implementation, such as supplies, consulting services, testing equipment, or any additional hardware or software required for development.

Implementation costs. Implementation costs are those expenses incurred when installing the new software. They include the time an organization's computer must be inoperative while the new software is being installed; the costs associated with testing and parallel operations until the reliability of the new system is assured; and training employees to use the system. In most cases, initial debugging will be far more extensive with custom-developed software and will often exceed the cost of development itself.

Another major implementation expense is file conversion, which involves incorporating old records into the new system's data base. If current records are not computerized, this requires manually entering records into the new system. If records are already computerized, they will have to be converted into the new system's data format. This usually requires development of a program to convert old computerized files into a form that the new system can use. If there is a need for continuous record-keeping (as, for instance, with year-to-date earnings in a payroll program), all relevant file data will have to be entered into the new system. Chapter 7 discusses installation issues in detail.

Costs of software errors. These are the costs of errors discovered after the initial integration of a system. Software errors discovered after integration are usually quite expensive: the principal cost may not be correcting the programming error, but rather undoing the damage that the error caused. For example, a software error may result in a large number of records containing incorrect information, or organizational decisions having been based on erroneous information. The impact of these kinds of errors can be quite significant.

If software is purchased or contracted for, the maintenance provider will often correct software errors for a specified warranty period without additional charge; other costs consequent to software errors must be borne by the purchaser. Maintenance contracts vary in the kinds of software errors that they cover; all errors may be repaired

or only certain kinds, such as those consequent to implementation. Programs developed in-house are generally corrected on a programmer per-hour rate, and correction costs are therefore less predictable.

Maintenance costs. These are the costs to update and adapt software to match changing organizational needs. As an example, a payroll program might have to be modified to reflect a change in FICA tax rates. The maintenance costs of a system will vary widely, depending upon such factors as the application, the complexity of the system, and the need for periodic updates. Maintenance costs for packaged software may be included in a maintenance contract.

Opportunity costs. Opportunity costs are those costs inherent in foreclosing one option in favor of another. If a program is developed in house, whatever the organizational data processing unit could have accomplished during the time it spent developing the program must be given up. Conversely, adoption of a sub-optimal off-the-shelf package means that the benefits of having a "perfect" package will be lost; the opportunity cost would be an estimate of those benefits lost over the period the software package is to be used, minus the additional expense involved in developing a "perfect" package.

Opportunity costs are hard to quantify precisely, but can be among the most important factors in software selection. Wherever a significant benefit can be estimated, it should be done. There may still be some degree of uncertainty, but careful thought and analysis can identify many important costs and assign them reasonable values. Section 5.6 on cost benefit analysis can help in the identification of significant opportunity costs.

Table 5 provides a synopsis of the relative costs of packaged versus developed software.

3.4.2 Summary

If no packages can be found that satisfy the essential requirements, custom development may be necessary. If a package will be acceptable only after some degree of customization, the cost of any required package modifications should be determined. It is not always possible or practical to tailor features into packaged software: doing so may void software or vendor warranties and reduce the reliability of the entire package. If a package is sold only in object code, tailoring may be impossible.

On the other hand, if one or more software packages have been found that are suitable for the application and conform closely to requirements, an off-the-shelf package is appropriate.

Table 5 - Cost Comparisons

COST COMPARISONS -CUSTOM DEVELOPMENT VS. SOFTWARE PACKAGE-			
TYPE OF COST	GENERAL COST	SPECIFIC COSTS FOR CUSTOM DEVELOPMENT	SPECIFIC COSTS FOR SOFTWARE PACKAGE
PERSONNEL COSTS	Time to train personnel on new system Time to draft requirements document Time to negotiate development or purchase contract	Cost of: Program development Training program development Documentation development	
PURCHASING COSTS	Cost of additional hardware needed for system	Supplies ancillary to development	Cost of software package
INSTALLATION COSTS	Cost of file conversion Downtime of computer during installation Cost of initial debugging	Initial debugging expenses often high	Initial debugging expenses often nominal
COSTS OF SOFTWARE ERRORS	Large portion of cost is often that of correcting software errors	Risk of significant software errors often high	Risk of significant software errors often low
MAINTENANCE COSTS	Costs of program updates Cost of implementing program improvements		
OPPORTUNITY COSTS		Costs due to delayed system implementation Inherent costs of other ADP projects being deferred during program development	Inherent costs of unsatisfied requirements

STEP FOUR : ASSESSMENT OF SUPPORT NEEDS

1. Documentation review
2. Modification support review
3. Installation support review
4. Training support review
5. Maintenance support review

Each software package under consideration will require a certain level of expert support. Insuring that package support needs will be met is an important step in the selection process. This requires a separate consideration of documentation, modification, installation, training, and maintenance needs for each package. Then, available services can be compared to these needs.

4.0 ASSESSMENT OF SUPPORT NEEDS

Proper support is essential to insure satisfactory package installation and its ongoing maintenance. Consideration of the support needs associated with each package is, therefore, a vital part of the selection process. Support needs should be determined for every candidate package. A package that suits organizational needs well, but cannot be adequately supported, can be much less useful in the long run than a less well designed package that is well supported.

4.1 Documentation Review

Documentation is essential to the efficient use of any software package. While the training program acquaints users with a package and its basic operations, documentation is needed as a long-term comprehensive reference guide to package use, features, and design. It is generally the only guide to infrequently used package features. It is also necessary as a diagnostic aid when the package doesn't work as expected.

Documentation is oriented to several classes of readers: management, computer operators, users, and maintenance personnel. Sufficient documentation should be purchased for each of these personnel to do their jobs properly. With most packages, documentation is divided into a number of functional categories. They are described in Table 6.

Some packages offer on-line documentation that can be accessed from computer terminals while using the package. This capacity is often quite helpful, but is never a complete substitute for standard documentation. The information contained in such on-line facilities is usually quite condensed and is intended primarily as a ready summary reference guide. If the system malfunctions and documentation is needed so that users or operators can recover files or restart the system, on-line documentation is of no use because it is inaccessible.

4.1.1 Characteristics of documentation

The quality of documentation is among a package's most important characteristics. If good, it can help the user make the most of a software package. If it is poor, important package functions and capabilities may never be used because they are not understood. Maintenance and modification without good documentation is very difficult; if technical documentation is poor, package life may be shortened and efficiency diminished.

Table 6 - Documentation Types

TYPE	FUNCTION
User Manual	<p>Guide to package use for a particular application or job type. Generally includes:</p> <ul style="list-style-type: none">o Data entry instructionso Descriptions of report formatso Error codes and responseso File entry and exit procedures
Operator Manual	<p>Describes system-level commands and procedures, and a comprehensive listing of all system commands and functions. Generally includes:</p> <ul style="list-style-type: none">o System security procedureso File recovery instructionso System start-up instructionso File back-up procedures
System Manual	<p>Explains, in non-technical terms, system functions and capabilities. Helpful to managers in developing workflows and making job assignments.</p>
Training Manual	<p>Explains package functions and procedures in a stepwise, tutorial style. Usually presents basic package usage skills first and then builds on them. Generally includes:</p> <ul style="list-style-type: none">o Pictorial representation of package screenso Precise data entry formatso Listing of commands and protocols
Program Maintenance Manual	<p>Description of system design, functional modules, system ports, etc., so data processing personnel can maintain and update the system. Generally includes:</p> <ul style="list-style-type: none">o System flow chartso Data base configuration informationo Program source code

Good documentation is clear and easy to read. It is geared to the level of technical sophistication of its intended audience. User documentation provides guidance on package operation, while program maintenance documentation includes the technical information necessary to modify the package or package parameters. Both should be clearly written and complete.

Documentation should also be complete. All package functions, capabilities, and limitations should be described fully. Good documentation includes troubleshooting guides that list common errors and appropriate responses. Logical organization is important. If all of the information is there but can't be found easily because of its presentation, the usefulness of documentation is impaired.

A good index is also important. It should reference the location of every significant cluster of information in the document and should include cross-references under other subject headings so that required information is easy to locate. Readers should not have to check a number of synonyms to find the subject heading they are looking for.

4.1.2 Documentation need evaluation

The documentation necessary for efficient use of each package should be determined. If maintenance will not be performed in house, technical information will probably not be needed, although if technical documents are inexpensive it is often good insurance to buy them.

After the necessary information for each package has been specified, the corresponding system documents should be identified and listed. The documents identified on this list will be reviewed during package evaluation.

4.2 Modifications

Despite the flexibility designed into packages, organizational needs are different enough that tailoring is sometimes necessary to make a software package fully appropriate. Modifications should be made to a package only to the degree that they make it more efficient. Section 5.6, on cost-benefit analysis, can help determine which features and capabilities are worth adding.

4.2.1 Simple modifications

Modifications are easiest if they are not internal to the package. Pre- and post-processing programs can condition the input or output of a program to make it more appropriate to organizational needs.

Pre-processing can include error checking of data to make sure that values fall within a certain range or that a number is consistent with a check digit. Prompts may be made more explicit or more consistent with organizational terminology or procedures, or the order in which data is input can be changed to match the forms from which the information will be copied.

A post-processing program may reformat package outputs so they fit current organizational forms. If a package is designed so that it can share data base information with other software, new reports can sometimes be added that use this additional information.

4.2.2 Software design features

Modern software packages are being designed with greater flexibility, which simplifies modifications and enhancements.

User ports are designed into some packages. These are the locations of information within the program that can be referenced by programmers who perform modifications. Processing can be diverted at a certain point to incorporate an additional program module or to eliminate a component part of processing by knowing where to interrupt the program stream. Software designs that incorporate user porting also facilitate sharing of data with other software.

Software packages with highly modularized design are also easier to modify internally. Small program modules can be adapted or added to reflect special organizational needs, and repercussions of changes are usually easier to identify and contain. Maintenance is easier because the modified modules are known, and the effect of updates or enhancements on the changed portions can be more easily determined.

High modularity of software design can also make it easier to change the auditing characteristics of a package. Since different functions are separated in modular design, monitoring a specific set of internal processes can be done more easily.

The design of the data base can also affect how easy it is to change recordkeeping characteristics and share information with other software systems. Some packages are

designed to keep information in a common organizational data base. Others may be able to reconfigure data. This can be helpful when different software systems will be partially integrated.

4.2.3 Other considerations

Tailoring a package may void all or some vendor or developer warranties, especially if modification is performed by in-house data processing staff. If extensive modification is required for any package under consideration, the warranty and maintenance policies of the vendor should be checked beforehand.

If internal modification of a package is required, whoever tailors it will almost certainly need access to the package's source code. If modifications to the source code are made by the vendor, and the source code is not made available, the purchaser will remain dependent on that vendor for the remainder of the life of the package. In this case, extra care should be taken to insure that the vendor is responsible and financially stable.

It is important to package maintenance, diagnosis, and future package modification to carefully document changes made to a package. Whether tailoring is done in-house or by the vendor, documentation standards for all changes should be made clear. Changes in the source code should be identified and closely commented in the software. Documentation should include a listing of all modules and routines that have been changed, and all changes in interfaces between software components, hardware, and data bases.

4.3 Installation Review

Package installation involves a number of issues. The software must be loaded into the computer's memory; hardware installed or modified; modifications made to the package; and package output carefully checked.

For a small unmodified package, installation can often be performed by a layperson familiar with the computer system. However, if modifications to hardware, interfacing software, or the package are involved, installation personnel should have the technical knowledge and experience to evaluate every significant aspect of installation.

Installation personnel should therefore have experience with other package installations. They should have some experience with software system testing, as well, so that

appropriate tests can be devised to assess whether the package has been installed and is working correctly. If installation involves interfaces to other software or data bases, installation personnel should also have a basic understanding of the whole system's application area(s) so that all interfaces reflect application, as well as software, requirements.

4.3.1 Installation support options

Installation can be performed by in-house personnel if they have the appropriate technical knowledge and experience. If in-house personnel are primarily responsible for maintaining other software with which the new package will be integrated, they probably have a technical knowledge of the entire system that will be difficult for outside personnel to match. Before relying upon in-house personnel for installation, make sure that they are able to provide required installation services. The reliability and technical proficiency of in-house data processing staff should be given the same scrutiny as that of a vendor.

Installation services are usually available from the vendor and are often included in the price of the package, particularly with larger systems. The vendor is usually (but not always) technically familiar with the package; has installed it before; and usually has access to the source code and the technical personnel who developed the package. If in-house personnel will install the package, make sure that the vendor or developer is willing to provide whatever technical information is required to perform installation.

If neither the vendor or in-house personnel are qualified or able to perform installation, a consultant may be retained to do it. Use of an installation consultant is usually an option of last resort, especially if an installation is complicated. The consultant must spend time to become familiar with the package, its environment, and any interfacing software or data bases.

There are a number of ways that a package can be installed, depending partly upon technical considerations. In most cases, the entire package, with whatever modifications have been made to it, is installed at once. If not, installation tasks can be split between more than one party, (e.g., the vendor can install the package and in-house data processing personnel can install and test modifications.) If package tailoring requires that software be modified internally, this approach is usually precluded. In those cases where the vendor will not install the package after it has been modified by another party, incremental installation is a common installation option.

4.3.2 Installation concerns

The time around installation is usually busy and closely scheduled. Problems and delays in package installation can force other implementation concerns to be rescheduled. Package testing, training and file conversion all require substantial advance planning and scheduling, and usually require that the new system be operational. If installation requires computer downtime, delays can have impact on all other uses of the computer system as well.

If installation is to be performed by a single party, it is usually best that whoever designed the package modifications also install the package. Modifications are usually the most troublesome part of installation. Since modification of a package requires intensive review of its design, whoever modified the software usually has an extensive understanding of both the package and its modifications, and should be aware of important installation issues.

When the scheduling constraints and technical requirements to installation of a particular package have been specified, potential installation options can be assessed according to how well they satisfy these needs.

4.4 Training Review

Package training should introduce employees to the new software package and teach them the basic commands and procedures they will need to know. It should also delineate the impact of the new package on the workplace, covering changes in procedures, job assignments and duties.

An effective training program includes practice time on the package, has regular follow-up training sessions, and integrates procedural and package training. It should take place during a slack period if possible, so that employees can become fully comfortable with the package before critical work is done.

4.4.1 Special considerations

The form and intensity a training program should take depends on a number of factors:

Complexity of operation. If the new package is complex to operate or requires that users know a large number of commands and input formats, training is necessarily going to take longer and be more complex. For complicated packages,

training should be broken up over a number of sessions, and reference documentation that outlines the most important information about package operation should be made available to each user.

Number of user types. Effective training relates only relevant information to trainees. If there are a number of kinds of employees, each of whom will have different uses of the package, training will have to address the different duties and job skills held by each group. This could entail separate training sessions for each type of user, or selective review of user and training documentation by each group.

Computer sophistication of trainees. If employees are conversant with computers, training will be simplified, particularly if the current system is similar to the new package. If trainees are not used to working with computers, they will have to be taught how to work with terminals, how to enter data, and a variety of other information that a more computer-sophisticated group of trainees would take for granted.

Quality of documentation. If a package has good documentation that is well organized and easy to read and follow, training will be simplified considerably. Computer-assisted training can be extremely useful in teaching employees to use a new package, and in some instances can supplant instructor-led training.

Other changes accompanying the new system. The more employees must learn, the longer and more carefully planned training should be. This is particularly true when major procedural changes will accompany use of the new package. Changes in workflow, deadlines, procedures, and duties will be disruptive. Effective training can mitigate the disruption. If changes are major, training should treat procedures and package operation simultaneously. If different equipment will be used with the new package, training should cover its use.

Employees should have a formalized introduction to a new package even if training is to be accomplished primarily through documentation. The introduction should present an overview of the package and its functions and should review any procedural changes that the new package will require.

Unless a package is quite simple, the system manager(s) and in-house maintenance personnel should attend package technical training. This will give them an overview of package operation and design and answer any questions they have about the system.

4.4.2 Evaluation of training needs

Evaluation of training needs is complicated because organizational circumstances and applications differ so greatly. A general procedure for analyzing training needs is suggested below.

- o Employees should be separated into training groups for each package. Employees in each group should have essentially similar job duties and responsibilities. If an employee has a unique set of responsibilities, he or she should be regarded as a separate training group.
- o The training needs of each group should be listed. The information each group will have to learn in order to do its tasks properly should be enumerated, along with rough estimates of the depth of the training required.
- o It should be determined which aspects of training can be shared among different groups. For example, both file clerks and secretaries may need to learn new data entry formats.
- o Finally, each training need should be cataloged, and the user types that share each need should be listed. A rough estimate of the time required to satisfy each training need should be made for each group.

The anticipated number of new hires for each user type over the package's life should be estimated and appended to the estimate of training needs. If the training required of new employees will be substantially different than for current personnel, the differences should be noted.

This analysis should be done separately for each package under consideration. The analysis will be used to evaluate vendor and developer training programs and is an important criterion in package selection.

4.5 Maintenance Review

Package maintenance services are required to insure that a package will remain useful and appropriate. Organizational needs and circumstances often change over time: changing regulations, tax rates, organizational structures, recordkeeping procedures, and employee assignments can affect software needs. Maintenance services help insure that a package will keep working efficiently and be appropriate over its life.

Described below are a number of maintenance support services available for most packages. Which services should be utilized depends upon organizational needs and resources.

Warranty services. Usually a package's developer or vendor warrants that it will be substantially free of programming defects for a specified period--generally three months to a year--after it has been installed. If a programming defect is discovered during this period, the vendor or developer will generally repair it or install a modified version of the program for no additional charge. This kind of service can often be extended past the original warranty deadline for an annual maintenance charge. Such warranties cover software repair only; the vendor and developer will not assume responsibility for consequent organizational costs incurred due to software errors. The warranty is usually void if user modifications have been made.

Enhancements/upgrades. It is common for package developers to continue development of commercially offered packages. New features and capabilities are sometimes added, response time improved, or data storage capacities changed or enlarged. These upgraded packages are frequently made available to licensees of earlier versions at reduced cost. Generally, these enhancements are completely compatible with previous versions of the package and can be installed easily. If the original version of the package has been modified, installation of upgrades may be more difficult.

Package updates. Changes in laws or circumstances governing an application area will change processing needs. A simple example is a change in tax rates and laws with respect to an accounting program. In many cases the package developer will offer new versions of the program that incorporate these kinds of changes. These updates are often available at no additional cost to organizations that have a maintenance contract with the vendor or developer.

Modification services. Package vendors often offer a service to make minor changes in the package to reflect different needs. If, for instance, employee numbers must be changed from 6-digit to 7-digit numbers, package modifications might be required. Changes in data base configuration and parameter values are often included in maintenance contracts. More substantial changes in software usually have to be procured on a cost reimbursable basis.

On-site package repair. A software package can sometimes stop working because of programming errors, file space limitations, or because the values of inputs, outputs or internally computed data exceed those anticipated by the developer. Services are often available that can perform

package repair on short notice. This usually entails a qualified programmer or technician coming on site within a short time after problems are reported. The need for this kind of service depends upon the criticality of the software package to performance of organizational functions, how tightly operations are scheduled, and whether replacement services for those computerized functions can be found. (See Contingency Services, below)

Contingency services. If the software package or its supporting hardware breaks down for a protracted period of time, replacement services may be needed until repairs are effected. These services can be provided by in-house or outside resources. Manual procedures may replace computerized services on a short-term basis, but if time or resource limitations preclude this, two other options are available. A software service bureau can provide replacement services, but is very expensive to retain on a contingency basis. If the package is not highly modified, another user of the same package will sometimes allow others to use it during off-hours. This sort of arrangement is usually reciprocal and ad hoc, but if other local users of the package can be identified, mutually beneficial arrangements can often be arranged.

Consultation. Many package developers and vendors offer consultation services. These services differ in scope, but are often very useful. If users do not know how to perform a particular operation or the package is not performing as expected, they may be able to call the package vendor or developer for information. If the package is maintained in house, and source code for the package is not owned, programmer-level consultation services may be available. Consultation services are usually a part of contracted maintenance agreements with the vendor or developer.

User groups. Package users sometimes band together to share information and to suggest future improvements to the developer. These user groups may take the form of regular meetings, or simply be represented by a newsletter.

STEP FIVE : PACKAGE SELECTION

1. Solicitation of proposals
2. Proposal evaluation
3. Package quality evaluation
4. Vendor evaluation
5. Support evaluation
6. Cost-benefit analysis
7. Final package selection

Final package selection should take in a multiplicity of considerations. Package performance should be adequate, and all required forms of support should be available and reliable. A routine method of package evaluation is advocated that provides for consideration of all relevant aspects of package performance and support.

5.0 PACKAGE SELECTION

In order to make the final selection, the candidate packages should be compared in detail. One mechanism to achieve this is to solicit proposals from the vendors and compare the written vendor responses. If this procedure is not followed, the information needed to compare the candidate packages should be gathered by the selection team. This chapter assumes that the process is one of soliciting proposals, but the collected information and the evaluation process are the same in either case.

5.1 Solicitation Of Proposals

Before close evaluation can begin, proposals should be solicited for each package under consideration. If a package is offered by more than one vendor, separate proposals should be solicited from each. The proposal should include all pertinent information about organizational requirements and hardware, maintenance and training needs.

A formal request for a proposal to provide goods and services should specify exactly what is required from prospective vendors. Because of the complexity of software packages, it is important that requirements be precisely stated.

5.1.1 The cover letter

The cover letter should state that the organization is seeking proposals for a software package and briefly describe the application. The letter should request a response, by a specified date, which explicitly addresses the following:

Vendor profile. The vendor should describe the scope of its operations, including the number of employees, the number of packages it supports, and whether it has a programming staff. The vendor should provide a description of its relationship to the package developer (i.e., whether the vendor is the developer of the package, a distributor, an authorized representative, or a broker for the developer).

Developer profile. The vendor should provide a short profile of the package developer, particularly if regular updates from the developer are anticipated or the developer assumes responsibility for package support.

Training and documentation options. The vendor should lay out all training options that are available, specify where training sessions will be held, and describe the scope and price of each option.

Support options. Various support options should be detailed, and average and guaranteed response times to service calls should be specified. If user groups (see Section 4.5) are available, they should be described as well.

Requirements conformance. The vendor's proposal should include a listing of every deviation from stated requirements. If a vital requirement cannot be met, the vendor's response should also state whether the package can be modified to work adequately, and at what price.

Complete price information. All prices related to the package should be detailed, including the following:

- o Package lease or license.
- o Any package customization required.
- o Package installation.
- o Each kind of documentation offered.
- o Each training option available.
- o Each support option available.

Technical response. System security, memory limitations, speed, and basic design should be described according to standard criteria.

Lead time required. The vendor should state how long, once a proposal has been accepted, it will take to have the package ready for installation.

Preparation required. If changes or additions to hardware, other software, or the workplace will be required, they should be delineated.

Miscellaneous. The vendor should also respond to the following questions:

- o If the vendor makes substantial modifications to a package, will it grant the right of documentation approval to the customer for all package additions and modifications?
- o Is source code included in the package purchase? If not, is it available, and at what cost?
- o Will the vendor grant the customer a 30-60 day package acceptance option?

- o Will the vendor agree to contractual caps on the price of subsequent support agreements after the first has expired?

5.1.2 The requirements statement

The solicitation should feature a complete listing of requirements, including all input and output formats and required response time characteristics. It is generally acceptable to include a copy of the requirements document.

If a special statement of requirements is drafted, it should specify which requirements are vital, which are important, and which are simply preferred. It should also list all hardware currently being used.

5.2 Proposal Evaluation

Proposals should be assembled and evaluation tasks divided among the selection team. The process described in this section allows early identification of packages that may be inappropriate.

5.2.1 Proposal review

Proposals should be examined to make sure they are reasonable in all major respects. If a package cannot meet a vital requirement or necessary modifications are so expensive that they would disqualify a package, it should be dropped from further consideration at this point.

5.2.2 Proposal summation

For each package, a summary profile should be constructed that lists all deviations from requirements, extraordinary features or capabilities, all support and training options offered by each vendor (and the cost of each), the lead time required by vendors to install the package, any required hardware additions or modifications, and special preparations necessary for installation. The summaries will aid the selection team in comparing different packages.

5.3 System Walk-throughs

Using package documentation, application procedures should be "walked through" and data followed as it would be processed by each candidate package. The package should handle and store information appropriately. System flowcharts are often helpful in picturing package processes. Particular attention should be paid to the following:

- o All required data should be processed or stored by the package in the required forms. For example, if employee identification numbers are six digits, or contain alphabetic characters, the software package should be able to accommodate them.
- o All relevant file and processing categories should be represented in the package (e.g., payment arrangements or inventory categories).
- o System data storage capacities should be adequate. Keep in mind that more functions than originally anticipated may eventually be computerized. Also, any possible growth in organization size or recordkeeping requirements over the package's life should be considered.
- o If records of package transactions need to be retained, they should be kept by the package, or separate manual procedures will have to be developed to record and store them. Such redundant manual procedures are generally an inefficient use of organizational resources.
- o Package features and capabilities will have to be integrated into the working environment. Package selection should insure that they can be, and that legal and internal policy restrictions can be adhered to.
- o Data should be accessed and manipulated only by those qualified and authorized to do so. If some functions should be restricted to certain personnel, security precautions may need to be developed.

Having gone through each package's procedures, make a list of all problems or questions that have been identified. Discuss them with the vendor. If the vendor's response to any of these issues is inadequate or too technically involved to evaluate, the data processing expert should review them and discuss any potential impacts.

5.4 Vendor Evaluation

The software package will, in most cases, become central to conducting organizational tasks in the application area. Thus it is very important that the package continue to function reliably and in accordance with any organizational changes. A secondary dependence therefore develops upon the package supplier. In many cases the vendor will provide support services, including maintenance, repair, and enhancement services. Evaluation of a package necessarily entails evaluation of its vendor as well.

5.4.1 Vendor stability evaluation

It is important to research both package vendors and package developers to insure that they are stable and adequately managed. If maintenance and support will be performed in house, these considerations are of less importance.

The vendor should have the technical staff to insure that the package is installed (and modified, if applicable) correctly. Some vendors function only as sales representatives for the package developer, necessitating the investigation of other support options. Sometimes, in such a case, a package's developer will assume responsibility for support services. In either case, available support should be adequate for each package under consideration. If a package will be regularly updated, this is even more important.

Although no single measure is infallible, some indications of the long-term stability and managerial competence of a vendor are suggested below.

- o If a vendor has been in business for a number of years, it is generally a good indication that it will continue to remain in business. Most vendors that fail do so within their first few years.
- o A low rate of technical staff turnover is another positive indicator. High rates often suggest poor management or a poor industry reputation.
- o User interviews can provide the perspectives of those who have dealt with the vendor before. This can provide some very important insights into vendor performance and reliability.
- o Dun and Bradstreet reports can often be valuable in evaluating whether a vendor is in good financial condition. A marginal business should not be relied upon for support.

- o Another good indication of the vendor's long-term financial stability and reliability is the way it responds to potential customers. The vendor proposal should be thorough, deadlines should be met, and questions should be answered satisfactorily.

There is no substitute for careful judgement in evaluating candidate package developers and vendors. The vendor should be well managed, should be easy to deal with, and should respect any commitments made.

5.4.2 The vendor interview

The package selection team should meet with the vendors of packages under consideration. Questions raised during proposal evaluation should be addressed to the satisfaction of all team members. If a vendor's response in any area is inadequate, let the vendor know that a more appropriate answer is expected. Some questions may require research or consultation with programmers, with the full response to be presented at a later date.

Contract and procurement personnel should be present at vendor interviews. They should make sure that any vendor guarantees or representations will be reflected in a contract, should that vendor be selected.

The interview should address whether the vendor has the staff and technical expertise to perform any package modification or support that may be needed. If possible, a vendor technical staff member should be present at the interview to address these kinds of questions.

The goal of the vendor interview is to evaluate whether the vendor has the professionalism and expertise to support a package adequately. The vendor presentation, which is described in the following section, will focus on the quality of the software package itself. Both the vendor interview and the vendor presentations may take place at the same occasion, but it is important that both the vendor and the package be evaluated separately.

5.4.3 Vendor presentations

The entire selection team should attend vendor presentations for the candidate packages that are still under consideration.

The presentation should offer a package demonstration and give selection team members an opportunity to use the package. Any problems or questions that come up should be noted so they can be resolved later. If possible, vendor technical staff should attend the presentation so questions can be answered directly. All guarantees of package performance or support deadlines should be noted.

5.4.4 On-site visits to package users

The vendor should be willing to supply a number of user references. It is extremely helpful to arrange an on-site visit and see the package in actual operation. If possible, the whole package selection team should attend. If there is an opportunity for hands-on use of the package, different procedures should be performed, with particular attention paid to the simplicity of package use and response time. The opportunity should be used to ask any questions that may have been raised in earlier package evaluation and to see exactly what procedures have to be followed to effect major application tasks.

5.5 Support Evaluation

Available support options should be evaluated for each candidate package. The process is a straightforward one: needs are compared to support options. Where there are a number of adequate support options, cost can be taken into closer consideration as a decision criterion. In no case should a package be chosen unless adequate support provisions can be made.

5.5.1 Installation

If no single installer can satisfy all needed technical requirements, cooperative installation using more than one installer may be used. If this is called for, very careful planning and preparation will be necessary. The responsibility for the different installation tasks will have to be clearly and explicitly understood by all parties. Close consultation is essential; many issues will have to be resolved in a large or complicated installation. Before counting on a cooperative installation effort, all potential parties to it should have a chance to meet and determine whether they can work together and whether scheduling will be a problem.

If installation cannot be performed adequately by in-house personnel, the vendor, or the developer, a computer consultant can be retained to do it. A consultant is sometimes retained to supply needed expertise to other installation personnel; for example, when an in-house installer doesn't have the knowledge or experience to do the data base reconfiguration required for an installation.

5.5.2 Documentation

Documentation from each package should be evaluated. The vendor or developer will often make documentation (except source code) available to prospective customers for evaluation. A deposit equivalent to the purchase price of the documents borrowed is often all that is required. Even if it has to be purchased, important documentation should be reviewed. At the minimum, a user manual and a technical manual should be evaluated.

Whenever possible, a document should be reviewed by the employee who would have to use it in the event of package purchase. Each group of reviewers should list five or six operations they would have to perform on the package. They should then try to determine, using the documentation, how that operation is performed. If the document is clear and well written, this should be fairly easy. If not, this procedure helps identify weak points of the documentation.

Index entries should be checked to insure that the page numbers are accurate and that all significant discussions of the keywords are included. Similarly, it should also be checked backwards, from text to index. Using the operations chosen above as examples, check the index to make sure that it includes all important aspects of each operation.

The reviewers' ratings of each document should be compiled into a single rating, with especially strong and weak points noted. Documentation for each package should be adequate. If not, that package should be withdrawn from consideration.

5.5.3 Training

Available training options for each package should be evaluated. If, for any package, significant training needs are not met by vendor or developer training, they will have to be provided in-house. If they cannot be met with either in-house or vendor/developer resources, the package should not be considered suitable.

If more than one training program is offered for a package, evaluation should be done separately for each program considered. Evaluation of training should include consideration of the following:

- o Quality of training documentation.
- o Whether training will incorporate procedural changes and package modifications.
- o Whether computer-assisted instruction is available.
- o Whether training will be conducted on-site.
- o Whether training will be tailored to the training groups and specific needs of the user organization.
- o Cost of training.
- o Whether special training is available for the functional manager, system manager, and system operator.

The relative importance of each of these issues depends on the specific details of an application, but all of them should be considered.

5.5.4 Maintenance

Consideration of maintenance options will depend on a number of factors. Those services that must be provided should be delineated for each package and then the most appropriate parties to perform each support service determined. In an application area that changes frequently, it is often a good idea to buy enhancement services. For instance, accounting packages may change every year to reflect changes in tax laws.

5.6 Cost-benefit Analysis

Each candidate package has a different set of features and characteristics: somehow the "best" package must be chosen. Assigning monetary values to significant package characteristics provides a common basis of comparison.

This is the stage at which final package selection begins in earnest. Earlier stages of consideration served to identify and eliminate unsuitable candidates. At this point, there should be three to five suitable candidates and the objective is to make the "best" selection.

Cost-benefit analysis is the surest means of doing this. It is time consuming and requires close analysis, but can determine the ultimate organizational cost of each candidate package more accurately than any other method of analysis.

The analysis should be scaled to the cost and criticality of the package. If analysis will be limited, it should focus on the most important aspects of package performance. A comprehensive but casual analysis is not very useful. If candidate packages are significantly different, cost-benefit analysis should address the most important points of difference.

5.6.1 Opportunity costs

An opportunity cost, or "foreclosure cost," is a cost inherent in giving up one option in favor of another. For example, assume that one must choose between two software packages that are identical in every respect, except that one has an extra feature and costs \$2,000 more. It is impossible to say which is the better buy unless the value of that extra feature is known. If that feature would result in cost savings of \$50,000 over the life of the package, the feature is most certainly worthwhile. If, on the other hand, its benefits are trivial, it would be more cost effective to forego it and buy the less expensive package.

In most cases, the packages being compared are far from identical. Each package offers a different mix of features and a quantification of opportunity costs may be difficult. However, these costs are not unimportant. In most cases, the consequent costs of a package will far exceed its purchase price.

5.6.2 Feature accounting

The costs of a package or feature all relate to deviations from "perfect" package performance. If the requirements document is carefully drafted, it should include a precise list of all features that should be included in a package. Additional features are justifiable only to the degree that they enhance organizational performance.

Packages should be compared as described in vendor proposals. Only the level of package performance that the vendor is willing to guarantee is relevant in comparing packages. All changes, adaptations and limitations of each package should be taken into consideration.

Unless the value of specific features is known, the various mixes of features offered by different packages cannot be evaluated meaningfully. Each feature should be given either a calculated cost or an assigned value. A calculated cost is one that may be determined directly from an accounting of anticipated organizational savings. An assigned value is dependent upon organizational values, priorities and circumstances. The best guide for assigning a value is to ask how much would be too much to pay for the benefits offered by a single feature.

5.7 Final Package Selection

The goal of software package evaluation is to select the optimal package with respect to organizational needs. The "perfect" package probably won't be found. Commercial software packages are designed to serve a generalized application need. Organizational characteristics are unique. Even the most flexibly designed software package cannot be responsive to every particular application need.

The customer, then, wants to pick the package that is the most appropriate. The purchase of a software package is a complex process, with so many different impacts, that unless this is approached systematically, important considerations might be left out or not given the weight they deserve.

In order to perform the final package selection, the candidate packages should be "scored" and assigned values, so that a uniform basis of comparison is used. Provision should be made for adjusting the comparison to reflect organizational priorities and values. The package that results in the highest point value should represent the best choice for a particular set of organizational circumstances. Ultimate point scores should not be viewed as perfect representations of a package's "true worth," but as criteria for a final decision.

5.8 Decision Approval

When the final selection of a software package has been made, the decision often has to be approved by others before the purchase is made. It is important that the criteria and processes used by the selection team are recorded and preserved. Literature from every package considered and the reasons for dropping packages from further consideration should be included.

Such a presentation should generally include the following information:

- o The requirements document.
- o The vendor responses for all final contenders.
- o Any documentation supporting the decision to buy packaged software instead of custom development.
- o The package evaluation and cost-benefit analysis for the final contenders, along with an explanation of the way weighting variables were assigned.

In most large corporations or government offices, this kind of decision justification is largely proceduralized. Whatever the form, decision presentation should emphasize the thorough and systematic approach of the package search.

STEP SIX : CONTRACT NEGOTIATION

1. Package contract negotiation
2. Support services contract negotiation
3. Specification of performance guarantees
4. Determination of compensation arrangements

The contract(s) with the package vendor and support service providers require(s) scrupulous attention to detail and careful representation of all agreements made with the customer. Careful review of each contract, before it has been signed, is the best insurance that the package will continue to fulfill organizational needs.

6.0 CONTRACT NEGOTIATION

The contract with the vendor or developer is the definitive statement of all promises and agreements made with the customer. It is vital that it be carefully drafted and reviewed. Any misunderstanding or unwarranted assumption could lead to increased organizational costs, less than optimal package performance, or inappropriate service agreements.

Government agencies and most large companies have established procedures for procurement of large purchases. Contracts for software packages and support services should conform to all applicable policies. It is important that a legal or procurements expert be involved in the drafting, revision, and review of the contract for the software package and any ancillary services, so that organizational needs are clearly and fully represented in the contract.

Every understanding with the vendor and any other support providers should be explicitly stated. All guarantees of package or support performance will be based on the contract. Therefore, every significant promise made by the package licensor and service providers should be written in the contract.

The contract should not contain terms that are subject to interpretation. Words like "prompt," "reliable," and "flexible" are useless unless their meanings are clearly defined in writing and understood by all involved parties.

Table 7 lists the details that any contract for software package purchase or support should address.

Table 7 - Issues Addressed by the Contract

Acceptable performance criteria.
Conditions of performance.
Penalties and remedies for nonconformance.
Warranty conditions.
Promised dates for deliveries and completion of services.
Prices and payment schedules.
Lists of applicable features, reports, and system components.

6.1 The Statement Of Work

The central part of the contract will be the Statement of Work, which describes the goods or services the contractor is expected to provide. It should include the following:

System specifications. All system specifications, as agreed to by the package provider, should be delineated. This can best be done by appending the requirements document, revised to reflect any changes or new understandings, to the contract.

A copy or representation of every report that the package will be expected to produce should be included with the contract. The information to be contained in each report should be listed, and the maximum and minimum values of each item specified.

Technical specifications for the package should also be detailed. Minimum file space limits for each record and data field should be specified, as well as the maximum number of files that will be stored. Maximum acceptable response times and minimum reliability characteristics should be specified in verifiable and unambiguous terms.

All specifications listed in the contract should be mutually acceptable. If the vendor is reluctant to have any particular standard included in the contract, another standard that is mutually acceptable can usually be found.

All documentation that will be purchased with the contract should be listed by name and by stock or catalog number. Delivery of all documentation should be required prior to installation.

Module listing. The software package and each component module should be listed in the contract. The version number of the package being purchased should be specified.

Complete specification of services. All promised services should be listed and detailed on the contract. Terms should be specific. The contract should specify the circumstances covered by the service agreements, exclusions to service coverage, and the maximum acceptable response time to a request for service. If installation services are included, the installation date should be specified. It is advisable to cite the names or qualifications of the personnel that will be performing different support tasks: trainers, installers, on-site diagnosticians, etc.

6.2 Performance Guarantees

A contract for a package or package support should specify in detail all standards to which the provider will be expected to adhere. It should include:

Package fixes. Repairs or modifications to the package should be made in conformance with contractual specifications, unless otherwise agreed. Maintenance service providers should agree not only to perform necessary repairs or modifications, but to correct any undesirable repercussions. If, for instance, a repair results in the disruption of other package features, or requires reconfiguration of the data base, it should be the responsibility of the maintenance service provider to insure that the package remains fully operational.

Deadlines for package repairs and modifications should be stated clearly. If a package problem impairs day-to-day use of the system, shorter "emergency" deadlines should be in force. For those changes that only increase package efficiency or usability, more lenient deadlines can be specified.

Warranty provisions. The terms and limitations of the package warranty should be specified. If the warranty and the maintenance contract run concurrently, the contract should specify that conflicts between them will be resolved in the favor of the customer.

Package acceptance. A package acceptance clause should be included in the contract. Acceptance options typically run ten to sixty days. All acceptance criteria, if different from the list of enforceable standards, should be delineated. "Satisfaction" is not an adequate criterion for acceptance.

The acceptance period should begin after correction of the last problem with the package. If the package provider has to make changes to the package to bring it into conformance with other provisions of the contract, the clock on the acceptance period should restart. Formal acceptance should be documented and signed by both the customer and vendor. These issues are discussed in greater detail in NBS Special Publication 500-136, "An Overview of Computer Software Acceptance Testing."

Penalty provisions. If either party to a contract fails to fulfill contractual responsibilities, reasonable penalties should be levied. Penalties should serve both to motivate each party to perform their contractual obligations and to compensate one party for losses and inconvenience caused by the other.

6.3 Compensation

The contract should specify the cost of the package and support services, and the agreed terms of payment. It should include:

Itemized price listing. The price of the software package, modifications, and services should be itemized separately. If they are sold "bundled", the contract should specify whether they can be unbundled in the future, and the price penalties for doing so.

Terms of payment. The schedule of payments for the package, installation, and each support service should be listed. The package and installation cost should not be paid in full until the acceptance period is over; it is common to pay half before installation and half after package acceptance. Because of this, payment terms should be expressed as time elapsed since milestones, such as acceptance, instead of specific calendar dates.

Most contracts will provide for penalties for late payment of money legitimately owed the package and service providers. How late a payment can be before it is considered a voidable breach of contract should also be specified.

6.4 Other Issues

There are a number of other issues a software package licensing agreement and support services contract should cover.

6.4.1 Cancellation

If the agreement with the package or support service provider is cancelled, precautions should be included in the contract that mitigate the harmful impacts.

Both parties should fulfill all contractual obligations until the time that a cancellation is effective. If a software package turns out to be troublesome, it might be tempting for the maintenance service provider(s) to cut their losses and cancel the support contract. Before a cancellation of the contract is effective, all problems identified to that point should be resolved.

There should be a notification requirement of thirty days or more before either party can cancel any agreement. This allows enough time for any other necessary arrangements to be made and allows the purchaser to evaluate whether all service needs have been met for the period that the contract was in force.

In case of cancellation, a refund schedule, or termination support schedule, for unreceived services should be determined beforehand. Pro-rata refund is the most usual arrangement, but if this isn't equitable, another arrangement can usually be worked out. A cancellation penalty is often specified to discourage frivolous cancellation and to compensate the other party for any expenses or inconvenience incurred.

The contract can prohibit unilateral cancellation entirely, or it can describe a set of circumstances in which it is allowed. If both the package and support services are provided by the same vendor, the option of returning the package if support is withdrawn should be retained. If cancellation is due to nonfulfillment by one party of its contractual obligations, voidable breaches should be described (e.g., how late a payment can be, or how long a serious package problem can go uncorrected).

If support service providers are unable to perform their obligations adequately, they should bear the costs of having them performed correctly by some other party. The conditions under which this may be done should be specified.

In the event that either party cancels the contract, the service providers should turn over copies of all records and documentation relating to the contract. Service records and documentation of any modifications and data base reconfigurations are vital to continued maintenance of the package.

6.4.2 Condition waivers

If any contract terms will not apply under a particular set of circumstances, those circumstances should be detailed. The contract should specify that all waivers of contract terms must be in writing.

6.4.3 Package ownership

Software packages are usually licensed, not purchased. The terms of the package license should be precisely specified. The following issues should be addressed in the contract:

- o What is the term of the license?
- o Can the license be revoked?

- o Is license renewal guaranteed after the end of the initial term?
- o Is the license transferrable?

Copying rights for the package should also be delineated in the licensing contract. Can back-up copies of the package be made for internal use? If not, will the vendor or developer provide back-up copies, and at what price? If a copy of the package will be provided only if the damaged original is returned, what will the copy cost and how long will it take to deliver it?

If internal modifications have been made to the package, it is important that the right to make back-ups is retained. It should also be specified whether the package can be used at a second site for a reduced licensing fee, no additional charge, or the full package price.

The contract should also include an assertion by the provider that it owns the package or has the right to license it, and assumes liability if its ownership is later cast into question. If modifications have been made to the package, ownership of the modifications should be specified. If contracted-for programs or routines are ever marketed, the organization that paid for the package development should receive a reasonable royalty from them.

6.4.4 Nondisclosure of proprietary information

If, in developing modifications or in servicing or installing the package, confidential information about organizational operations will be revealed to package or support services providers, the contract should include a clause that prohibits them from revealing what they have incidentally learned.

6.4.5 Future performance/contract renewal

If use of the software package is anticipated for a number of years, and support service contracts run only annually, the contract(s) should include some provisions regarding future contract renewals. It should be agreed (and documented) that contract renewals will have the same terms as the contract replaced unless changes are submitted to the other party at least thirty days before expiration. If possible, limitations should be set on the amount that contract renewals may increase in price.

6.4.6 Contingencies

Contingencies that might affect contract performance should be included in the contract. If the package or service provider goes out of business, all documentation and records should be turned back to the customer. If the service provider is providing complex maintenance or modification services, source code may be required for another party to assume maintenance; if this is the case, the source code should also be turned over.

STEP SEVEN : PACKAGE INSTALLATION

1. Formation of the installation team
2. Identification of installation issues
3. Package customization
4. Employee training
5. Transition to new package/procedures
6. Implementation of back-up procedures
7. Post-installation performance evaluation

The period between the purchase of a software package and its reliable use involves considerable accommodations between the package and organizational procedures, files, employees, and physical environments. Organizational adaptation should take place in a number of different spheres. Careful planning helps to insure that the transition to the new software package will be trouble-free.

7.0 PACKAGE INSTALLATION

When a new software package is incorporated into an organization, adjustments are inevitable. Interfaces between the package and its organizational environment have to be identified and smoothed out. A number of different tasks comprise package installation:

- o Preparation of test data.
- o Package installation.
- o Employee training.
- o File conversion.
- o Package testing.
- o Implementation of procedural changes.
- o Institution of back-up procedures.

Package testing, although part of installation, is discussed separately in Chapter 8.

7.1 Formation Of The Installation Team

Once the software package has been selected and the contract negotiated, an installation team should be assembled. Its members should include a team leader, data processing expert, and representative of user personnel. The team will analyze and develop strategies for handling package installation needs and oversee the scheduling and implementation of the identified tasks.

The installation team should collectively have a thorough understanding of both the system and organizational operations. Since no single person on the team will have a comprehensive understanding of all technical, procedural, and policy issues of installation, it is important that team members communicate freely. The team should meet regularly as a whole, and each member should be aware of the activities of the others.

Table 8 lists the duties typically required of installation team members.

Table 8 - Installation Team Duties

TEAM MANAGER

- o Coordinate installation
- o Assign installation tasks
- o Approve scheduling
- o Develop strategy for package introduction and acceptance
- o Serve as liaison to other managers

USER REPRESENTATIVE

- o Evaluate effectiveness of training proposals
- o Coordinate package with organizational procedures
- o Serve as liaison from team to employees

DATA PROCESSING EXPERT

- o Evaluate technical merit of installation proposals
 - o Estimate and coordinate file conversion and testing efforts
 - o Evaluate vendor performance
 - o Serve as technical liaison to vendor
-

7.1.1 The team leader

The functional manager of the package application should head the installation team. If more than one manager is involved, a team leader should be designated from among them. The team leader is responsible for selecting the other members of the installation team, making duty assignments, coordinating the installation effort, approving scheduling of installation tasks, and establishing installation priorities. The team leader is generally the most appropriate liaison to other departments or managers.

7.1.2 User representatives

The installation team should include employees who will actually be using the new package. They will be needed to evaluate and make recommendations on integrating the package into the working environment, on the adequacy of planned training, and on the effects of the new package on day-to-day organizational operations. If possible, there should be at least one user representative from each class of employee

that will use the package (e.g., if management and clerical staffs will both be using the package in a different way, each group should have a representative on the installation team).

7.1.3 The data processing expert

The data processing expert may be a member of the in-house data processing staff or an outside consultant. If installation options require technical consultation, the data processing expert can apprise the installation team of the consequences associated with different options. The data processing expert is also the most appropriate member of the installation team to communicate organizational concerns to the vendor and to evaluate the vendor's technical performance.

If in-house data processing personnel are not available, or if the organization has no data processing staff, a package installation consultant may be retained. If possible, a consultant should be retained who has experience with packages in the application area. It is better still if one can be found who has experience installing the same software package.

7.2 Installation Considerations

The central task in any package installation will, of course, be the installation of the package onto the computer. Although installation usually involves reading the package from magnetic disk or tape into the computer system, substantial preparations for installation often have to be made. Any new hardware the system will use should be ordered, installed, and tested. If reconfiguration of the workplace is required, that should be done as well. If installation requires dedicated use of the computer system, other users should be notified and installation scheduled for an appropriate time.

Another integral part of installation is package testing. Custom modifications to the package should be completed and tested. Test data should be prepared which invokes major package functions. Testing should be scheduled immediately after installation, while installation personnel are still present. Chapter 8 discusses package testing in detail.

A number of factors that should be addressed in the installation plan are discussed below. Table 9 summarizes installation considerations.

Table 9 - Components of Installation

SYSTEM CUSTOMIZATION

- o Arrange for package modifications

SYSTEM INSTALLATION

- o Prepare hardware environment
- o Design any special forms required
- o Order new supplies
- o Reconfigure workplace for new package

FILE CONVERSION

- o Determine file conversion needs
- o Arrange for file conversion
- o Have conversion program prepared and tested
- o Check new files for format and accuracy

PROCEDURAL CHANGES

- o Develop any new procedures required by package
- o Have new procedures approved, if required
- o Implement and propagate new procedures

BACK-UP PROCEDURES

- o Develop back-up procedures for files and processes

TRAINING

- o Develop training program
- o Define assistance procedures
- o Plan user forums until employees are comfortable with package

HUMAN CONSIDERATIONS

- o Develop plan for package introduction
 - o Implement formal mechanism for employee comments
-

7.2.1 File conversion

File conversion is a time-consuming and resource-intensive part of installation. Relevant organizational files and records must be converted into the form required by the new package.

If the new software package represents an initial computerization, data entry, although it is a significant undertaking, can help familiarize employees with the new package, and may therefore supplement training. If the

number of files is too large to enter entirely in house, it may still be wise to have in-house personnel enter at least part of them.

If the new software package requires a large amount of manual data entry before it can become fully operative, incremental file conversion may be worth considering. This involves entering data into the new package in relevant and coherent subgroups, so that the package can process some information before all organizational files are on-line. This can enable some functions, work locations, or departments to be brought on-line before others.

If previous records were computerized, a special computer program can usually be written to convert old files into the new formats. However, it is sometimes better to enter them, one by one, through the package as a series of transactions instead of simply reformatting present files. File information will then be passed through the new package's error-checking mechanisms, so mistakes can be caught and corrected. Also, when file conversion is processed as a series of transactions, an audit trail can be produced so that system or file conversion errors can be traced.

7.2.2 Procedural changes

Installation of a new software package is often accompanied by concurrent changes in organizational procedures. Changes in procedures and policies should be studied, detailed, and implemented prior to installation of the new package. If changes in procedure require approval, requests for consideration should be submitted early enough that the rest of installation is not delayed. Changes in job assignments should be identified, and employees should be notified of changes as early as possible. Organizational procedure manuals should also be revised to reflect policies or procedures that will be used after the package has been installed.

7.2.3 Back-up procedures

Procedures should be developed for the contingencies of computer failure and file loss. At the most basic level, this usually entails periodically copying computer records onto disk or tape so that current file information can be reconstructed if necessary. How often this should be done depends largely upon organizational needs and circumstances, and should be decided internally. If audit trails need to be maintained, provisions should be made for keeping transaction

records as well. A number of approaches to system back-up have been standardized and are documented in detail in other literature.

7.2.4 Training

Training should be scheduled so that employees are still able to do the work they are required to do. If employees will be doing manual file conversion, they should know the basics of package operation, and understand file formatting and data entry procedures before they start. Training is generally best scheduled for the period immediately preceding installation and initial testing. Noncritical practice time on a fully functional package is also helpful.

If the functional manager or system manager will be given special training, that should be arranged. Training sessions should be arranged for the employee groups that have been singled out for vendor or developer training. (When scheduling training, be mindful of the system functions each group will have to perform during file conversion and parallel operations.)

The employee groups and the information not covered by vendor training should be identified. Arrangements should be made so that the training needs of each group are satisfied. Separate policy training may be provided in house if procedural changes are extensive or significant.

Training for employees whose contact with the package will be superficial may be provided by another organizational employee who has received more extensive training (usually a supervisor).

The training to be provided in-house should be identified and addressed in the installation plan. Trainers should be designated, and their training given appropriate emphasis. If in-house training documentation is to be written, its scope should be planned, writers should be assigned, and a reasonable completion date determined.

7.2.5 Human considerations

Some of the most persistent installation problems result from employees' reactions to introduction of a new package and the procedural changes that accompany it. Introduction of a package into the workplace should be very carefully handled. The installation team should be aware of human consequences of the new package and plan ways to mitigate any problems that may arise.

Employee resistance to a new package is not uncommon. It is often initially perceived as a hindrance rather than an aid to performing duties; and the changes in procedures, duties, and job titles that accompany a new package are sometimes resented. To minimize these problems, prepare employees for changes before they occur.

Consultation with employees should be sought throughout the package selection and installation process. User personnel can provide valuable input on day-to-day procedures, and this input can provide insurance that a particular package will be appropriate to an organization's application needs. If employees feel that their counsel is regarded seriously, they are less likely to feel that a package has been arbitrarily imposed on them.

Employee forums should be held to discuss how procedures might be streamlined through computerization. A suggestion box can be used to solicit comments and ideas that employees might be reluctant to offer publicly. The user representatives in the package selection and installation teams should serve as employee liaisons.

User forums should be scheduled weekly or bi-weekly for an indefinite period, and leaders should be selected. User forums can be discontinued at any point at which employees are comfortable and competent on the package. They should be scheduled, though, at least through the end of parallel operations.

If employees will be dismissed or transferred due to efficiency gains associated with the new package, this should be planned for early. To the greatest degree practical, dismissals or transfers should be avoided close to package introduction.

Installation analysis should identify any potential negative effects of the new software package and propose strategies for mitigating them. This analysis should try to assume the perspective of different groups of employees that will be affected by the package.

Any negative effects of installation should be distributed. If, for example, large amounts of tedious data entry are required, a single employee or group of employees should not do it all. It can be alienating to work on a computer terminal all day; if an employee does little else, both performance and job satisfaction may decline. Work should be structured so that it is interspersed with other tasks and contact with other people.

Be frank with employees about disadvantages of the new package. Acknowledge that their jobs might change in ways that might not always be welcomed. In most cases there is no

problem once employees have become accustomed to a new package, but the period immediately after package installation can be critical in shaping their attitudes.

Some employees may be "computerphobic," and may be so intimidated by the computer that they are not able to work effectively on it. In these cases it is sometimes helpful to run purely practice training data on the package for a while before actual organizational files are processed. This allows personnel to relate to the computer under less threatening circumstances and become familiar with the keyboard, terminal, and package commands.

Employees often fear that they may make mistakes on the new package that they will not be able to correct. Let the employees know that mistakes are not critical, particularly during the first few weeks of installation. Most packages have provisions so that important data cannot be destroyed without invoking complex commands; most mistakes are correctable if they are identified.

7.3 Installation Planning

Package installation involves many different functions, and formal planning is the only way to insure that it will progress smoothly. The installation plan should address all aspects of installation, testing, and training. Component tasks should be scheduled, and task leaders designated.

The day(s) that a software package is installed will be hard to schedule, since installation is in part concerned with correcting unforeseen difficulties. But installation should be systematized to the greatest degree possible.

Package demonstrations should be conducted for all employees who will use the system. If employees received some training before installation, they should have an opportunity to use the package. Questions or problems about package use should be addressed.

Employees should be familiar with the package before they do any actual work, including manual file conversion. At minimum, they should know the command and correction procedures for every operation they are required to perform. Formal training should be scheduled for a time very close to package demonstration, preferably immediately after.

7.3.1 Preparing the environment

Careful preparation and planning can help insure that the installation period will not be overly troublesome. The post-installation period is usually hectic. The more carefully installation is planned, the sooner the package can be relied upon for day-to-day use.

User, operator, and training documentation should be received before the package is installed. Training documentation, in particular, should be inspected to make sure that it is adequate and appropriate to organizational needs. If it is not, in-house training manuals or supplements to vendor manuals will have to be drafted.

Development of package modifications often requires dedicated use of the operating system during design and testing. If this is the case, arrangements should be made so that the system development staff have periodic access to the computer system as necessary. This sometimes requires that the computer not be used for anything else, and other work should be scheduled accordingly.

If required, a workspace should be cleared for use by installation personnel. If installation requires all computer resources, this should also be arranged. All necessary assistance (whether in the form of dedicated employee time or other resources) should be provided to the installation staff.

If automatic file conversion will be done during installation, current computerized files should be made available to installation personnel. Personnel should be assigned to spot-check converted files. If manual file conversion is to be done, employees should be trained to do the required data entry. Employees should not be responsible for checking the same files they have entered. If files from more than one source will be integrated by the new package, they should be assembled to facilitate data entry. If incremental file conversion will be employed, determine which files will be needed first.

7.3.2 Operational planning

If procedural changes will accompany installation of the software package, they should be worked out in detail. Although procedural changes were considered at the needs analysis stage of package selection, they should be reconsidered and refined during installation planning with respect to specific features and characteristics of the software package purchased. The package may have different features or data structures than detailed in the original

requirements document; procedures should be revised as appropriate.

The frequency that file back-ups are made, the storage location of back-up records, and the personnel responsible for performing back-ups should be detailed. Contingency plans should be defined. If operations must be continuous, manual procedures should be prepared for use in the event of system failure. Procedures should provide for recording of all transactions so they can be entered on-line when the system is operational again.

If changes in policies or procedures require approval from higher management, they should be submitted early enough that installation will not be delayed. In large organizations, it is typical that procedures are formally documented in policy manuals. Policy manuals should be revised in accordance with organizational requirements to reflect installation-related changes in workflow, job assignments or procedures.

The installation plan should also detail how new policies will be propagated. Memos should be circulated summarizing changes in policies immediately before or after package installation. If changes are profound, new policies and procedures should be integrated into package training.

7.3.3 Package customization

If the software package will be customized, the installation plan should provide for regular monitoring of the progress of package modification. If the development staff requires the computer for testing the package modifications, arrangements should be made so they may do so.

Documentation of package modifications should be reviewed by the data processing member of the installation team; good documentation is absolutely vital for later package enhancement and modification. After the package has been in use for several years, documentation is often the only comprehensive guide to package modifications.

7.3.4 The transition period

The transition period is the time from package installation until the end of parallel operations. During this time, files and procedures should be changed to suit the new package, and package performance is constantly checked and monitored.

Demands on personnel are likely to be high during this period. Not only are they learning to use the new package, but current operations must be maintained. During this period, package results should also be closely checked against manual procedures. For these reasons, it is best that installation take place during a slow period, if possible.

If temporary personnel need to be hired, this should be arranged. Regular employees should work on the new package to the greatest extent possible; temporary personnel should conduct former procedures and spot check files. They should be supervised closely enough to insure that current procedures are conducted correctly.

The new package should not be depended upon for any crucial report or deadline until parallel operations have been completed. In fact, manual records should be kept, if possible, of every file in which a transaction has occurred during the transition period. This way, if the package has processed data incorrectly, the files with faulty information can be identified and corrected.

7.4 Performance Evaluation

After the package has been installed and in use for some time, an assessment should be made of how well it performs with respect to organizational needs. This evaluation should address whether anticipated efficiency gains have been realized, and should identify any problem areas that might be corrected.

Performance evaluation should take place six to twelve months after package installation, after major package and procedural obstacles have been resolved. It is helpful to do a similar evaluation annually to insure that the package continues to be effective in changing organizational circumstances. Policies, workloads, legal requirements, and organizational duties can change over time, and re-evaluation of package performance can help insure that the package remains appropriate.

7.4.1 Evaluation criteria

The most basic performance evaluation is simply an assessment of whether the new package meets organizational requirements. The package specifications in the contract can be used as a checklist, and any nonconformance should be noted. If the package exceeds requirements in any important respects, that should be noted as well.

A more detailed assessment can include the specific way in which the package has been integrated into the organizational environment, with particular focus on the interfaces between employees and the package. Areas which deserve particular attention are suggested in Table 10.

The most complete kind of performance evaluation involves calculation of the organizational costs saved or incurred by using a new software package. Depending on how carefully it is done, this evaluation can be quite precise (and time consuming). The opportunity cost guidelines in Section 5.6.1 can be used to evaluate package cost effectiveness. Simply recalculate each cost identified as experience dictates. Subtract from this all package costs including the cost for the package and its modifications, installation, maintenance, and training over the package's life. The result will be a net cost or savings associated with the new software package.

7.4.2 Performance monitoring

A variety of strategies can help insure that the package's performance is optimal. They involve early identification of substandard package performance and procedures for regular re-evaluation of this performance.

7.4.2.1 Performance milestones

At the time the package is installed, a series of performance criteria should be established that reflect organizational needs and priorities. Table 10 may be used as a guide, but any peculiar organizational factors should be reflected. These may be drafted into checklist form so that major users can rate package appropriateness and performance on a numerical scale. Such evaluations should be made at least annually, and may be required more frequently in a quickly changing environment.

7.4.2.2 Employee feedback

Some formal mechanism should be established to receive and consider employee input regarding the software package. Complaints, suggestions, and observations should be solicited and saved; those that cannot be immediately resolved should be considered in the packages's annual performance evaluation.

Table 10 - Performance Evaluation Criteria

PROCESSING TIME

- o Interactive processing
- o File searching
- o Report generation
- o System closeout

ACCURACY/RELIABILITY

- o Frequency/severity of downtime
- o Accuracy of processed information
- o Adequate error detecting mechanisms
- o Predictable response to all input

HUMAN INTERFACE

- o Operators comfortable and conversant with package
- o No significant operator-induced errors
- o Training is adequate for operator needs
- o Operating procedures not unduly time consuming

PACKAGE SPECIFICATIONS

- o Appropriate file space limitations
- o All required information maintained by package
- o Adequate file search categories
- o Peripherally stored data readily retrievable

ORGANIZATIONAL INTEGRATION

- o Organizational and package procedures integrated
 - o Required information easily locatable
 - o Work flow and document flow rationalized
 - o Package accessible to all who need it
-

4.2.3 Maintenance contract evaluation

The package's maintenance contract or agreement should be reviewed annually to insure full compliance with organizational requirements. It is best to do this prior to renewal so appropriate additional provisions can be added to the next contract. Any problems with maintenance services should be noted and resolved before contract renewal.

7.4.3 Insuring conformance to needs

If any outstanding problems or limitations of the software package are discovered, they should be resolved immediately. Less important problems and limitations should be documented and saved for inclusion in the next performance evaluation.

At each evaluation, all problems and suggested improvements to the package should be listed. Particular attention should be paid to the interfaces between employees, policies, organizational goals, and the software package. Any inefficiencies that can be identified (not just in the package, but in any part of the application) should be noted.

Secondly, a list should be made of all proposals to increase system efficiency and improve system interfaces. Employee suggestions might provide some valuable ideas. For those suggestions that require changes to the software package, a data processing expert may be consulted to make sure that they are feasible and cost effective. Those suggestions that are practical and would result in significantly increased efficiency, clarity, or employee satisfaction should be implemented.

Figure 15 is a timeline of the major tasks involved in an installation. It is offered only to give an idea of the general timeframe of a package installation; it should not be considered a guideline, but simply an example.

ACCOUNTING PACKAGE INSTALLATION

Software package selected	-0-	Feb 1
Vendor selected		
Package modifications scheduled		
Manual file conversion estimated	-0-	Mar 10
File conversion program scheduled		
Organizational procedures analyzed		
New hardware ordered	-0-	May 1
New forms designed		
Schedule training		
Back-up procedures defined	-0-	Jul 1
Training program designed		
Review vendor progress		
Have new procedures approved	-0-	Aug 15
Schedule manual file conversion		
Order new supplies	-0-	Oct 1
Arrange for temporary transition personnel		
Prepare package test data		
Arrange to have workload reduced during installation period		
Promulgate new policies	-0-	Nov 15
Check on vendor progress		
Test file conversion program		
Arrange for computer down time during installation		
New hardware arrives	-0-	Dec 20
New supplies arrive		
Documentation of customizations inspected and approved		
Clear workspaces for installation	-0-	Jan 28
Make installation personnel available to vendor		
Have test data ready for vendor		

(continued)

Figure 15 (page 1 of 2) -

Sample Installation Milestone Timeline

	(continued)	
Installation	-0-	Feb 1
Package specification testing		
Sample data testing		
Package demonstration		
Initial training begins	-0-	Feb 15
Automated file conversion done		
Manual file conversion begun	-0-	Mar 15
Daily output checking		
In-house supplemental training		
File conversion complete	-0-	May 1
Parallel operations begun		
User forums begun	-0-	Jun 15
File formats double-checked		
First cycle of parallel operations	-0-	Jun 24
closed out, package output checked		
Conference with vendor		
Second cycle of parallel operations	-0-	Jul 5
Evaluate in-house training		
Evaluate package performance		
Package acceptance		
Third cycle of parallel operations	-0-	Jul 19
--decide whether to suspend		
Prepare to store or discard old		
hard files		
Conduct performance evaluation	-0-	Sep 15
Spot check files		
Conduct user forum		
Following year...		
Evaluate vendor performance	-0-	May 1
Evaluate package/organization		
interface		
Renew maintenance contract	-0-	Jun 1

Figure 15 (page 2 of 2) -

Sample Installation Milestone Timeline

STEP EIGHT : PACKAGE TESTING

1. Identification of test goals
2. Performance of package testing
3. Package acceptance

Careful package testing before and during installation can minimize the expense of prolonged parallel operations, and help insure that package operation will be accurate and trouble-free. Application of a number of progressively designed testing procedures helps insure that any problems that may exist are detected at an early stage, and at minimum cost.

8.0 PACKAGE TESTING

8.1 Testing Issues And Goals

Testing is an essential part of the installation of a software package. Thorough testing should provide assurance of the package's reliability.

One significant advantage of off-the-shelf software packages is that their reliability may often be greater than that of custom-developed systems. If an off-the-shelf package has had many previous users, program errors have often been found and corrected.

A primary concern during software package installation is identifying and eliminating input errors. A number of factors can affect the accuracy and suitability of input data. Incorrect or inadequate training can result in employees entering incorrectly formatted data or inappropriate commands. If file conversion is faulty, data can be lost or incorrectly stored. Poorly planned interfaces between the new package and organizational procedures can result in incorrect data entry or lost paperwork. Thorough testing identifies and enables correction of these circumstances.

Testing should be done in several stages so that problems are detected as early as possible in the installation process. The earlier that problems can be identified and corrected, the sooner a package can be fully installed. The software package should not be counted on for any critical function or report until testing is completed and parallel operations have resulted in at least one trouble-free cycle.

Testing is treated separately from performance evaluation, which is discussed in Chapter 7, although their purposes overlap. The goal of acceptance testing is to insure that the package performs as expected and that it is sufficiently reliable and accurate to support its application. Performance evaluation is more concerned with optimization of package performance with respect to organizational goals and circumstances.

The following is a guide to package testing for the functional manager. In most cases, testing is overseen by data processing personnel, but the manager should be aware of the procedures and issues involved. If the package is small, or testing cannot be overseen by data processing personnel, the following may be used as a guide to conducting package testing. A more detailed discussion of all aspects of the testing process can be found in NBS Special Publication 500-136, "An Overview of Computer Software Acceptance Testing" and in the Federal Information Processing Standards

A detailed test plan should be written that lists every test to be performed and the data fields that should be checked after each test. Acceptance criteria should be specified in precise terms to insure that the package meets all requirements. Testing should be scheduled, and personnel assigned for file-checking. A vendor representative should be present during specifications matching and sample data testing to help resolve any immediate problems.

Parallel operations offer the greatest degree of protection against damage. Other measures are less adequate but still valuable. If testing is done in the order presented, testing costs are minimized, and the most potentially damaging problems can usually be circumvented in early stages of package evaluation. Table 11 presents a summary of the tests discussed.

Table 11 - Software Package Testing

PRE-INSTALLATION TESTING

- o System walk-throughs
- o Benchmark testing
- o Trial Use
- o File conversion program validation

INSTALLATION TESTING

- o Specifications matching
- o Sample data testing

POST-INSTALLATION TESTING

- o Parallel operations
-

8.2 Pre-installation Testing

A number of tests should be conducted before the installation plan is drafted to help insure package functionality and reliability. They include system walk-throughs, benchmark testing, and file conversion program validation. These tests are helpful in identifying issues the installation plan will have to treat. Testing of the

file conversion program, if there is one, can help insure that data processed by the package is correct. These preliminary tests help insure that the period of parallel operations will not have to be extended. Since parallel operations are expensive to conduct, installation costs can be significantly reduced if they are required for only two or three cycles.

8.2.1 System walk-throughs

A system walk-through consists of tracking each type of data through a software package, using package documentation as a guide. It is a preliminary form of package testing that can help identify potential problems of data storage and processing.

Special attention should be given to the points of interface between the software package and organizational components, specifically where data originates, which people handle it, where it is stored, and which organizational and system functions require its use. Any unclear or unaccounted-for steps should be resolved in the installation plan. The same process should be followed for package outputs.

8.2.2 Benchmark testing

Benchmark tests check a package's performance to insure that it conforms to package specifications. Generalized computer programs are used to enter data into the package and check the accuracy of basic calculations; the timing and accuracy of data storage and retrieval functions; and file and memory limitations.

Off-the-shelf software packages have usually undergone benchmark testing before they are marketed. Unless extensive modifications have been made to the program, or testing results are untrustworthy, additional benchmark testing is usually unnecessary. Generally, it is safe to forego benchmark testing of a software package if it is widely used (more than 100 users) and has a sound reputation. However, if an off-the-shelf package has been extensively customized, it may be a good idea to do benchmark testing to insure that customized features do not interfere with the package's processing and do not contribute errors of their own. When customizations are performed by the vendor, such benchmark testing is usually done before the package is delivered to the customer.

Benchmark testing may be of limited use because it does not accurately simulate the actual working environment in which the package will be used. Certain kinds of often-used calculations may not be invoked by benchmark testing, and actual processing may move and manipulate data in ways that a benchmark test cannot simulate. However, it can be helpful in identifying package data handling and storage limits and in testing basic calculations employed in package enhancements.

8.2.3 Trial use

Many vendors allow as much as 30 days' trial use of a package before purchase. If the new package does not require extensive modification or new hardware, trial use allows the user to see how well the package can be integrated into the environment in which it will operate and how well it meets organizational needs.

Every function likely to be used if the package were to be permanently installed should be employed. If possible, user personnel should operate the package; their comments may be useful in deciding whether to purchase the package and whether modifications may be needed. If trial use of a package is planned, some basic level of employee training should be arranged so that employees are able to operate the package.

Trial use also allows the user the opportunity to see how the package reacts to predictable kinds of misuse. Impossible values should be entered so that error-checking mechanisms can be tested. If more than one key is pressed simultaneously, how does the package react? If no information is entered for a vital field, does the package notify the user? If the system is unplugged in the middle of an operation, can it recover at least part of the information it had processed? Trial use allows these matters--so important for reliable day-to-day use of the package--to be resolved.

Trial use is of limited value in guaranteeing reliability, but it does allow hands-on use of the package. If organizational resources and time constraints allow, it is recommended. If trial use is employed, the following factors should be explored:

- o Is the package relatively easy for operators to use?
- o Is all output accurate and in conformance with organizational standards?

- o Are report and file formats acceptable?
- o Is the response time acceptable?

8.2.4 File conversion program validation

If file conversion will be computerized, it is vital that the conversion program be tested before files are entered into the package. A variety of sample data should be run through the file conversion program, and test files should be retrieved from the new system and checked to insure that they have been accurately transferred. If possible, this should be done before package installation, because installation could be delayed if the file conversion program needs to be altered.

8.3 Installation Testing

When the package has been installed, a series of tests should be performed to insure package reliability. These tests, which are described below, are specifications matching and sample data testing.

8.3.1 Specifications matching

After entering a small amount of prepared sample data, check the various outputs of the package and make sure they are correct and in acceptable formats. Files should be retrieved as specified in the package contract specification. Reports should match any special forms on which they are printed, and all required output should be generated correctly from the test data. Any deviations from specifications should be reported immediately to the vendor.

8.3.2 Sample data testing

This test calls for the new software package to process information so that package output may be checked. Again, a small amount of test data should be used, albeit more than for specifications matching. Sample data should be entered in every field that will be employed when the package is in full use; it is best to compile test data before the package has been installed so that this stage is not unduly time-consuming. Make sure that the output conforms to the output of current manual and/or automated procedures. If the

new package has error-checking features, "impossible" values should be entered to insure that the package catches them.

Generate samples of every report or other output for every function that will be used, and check the results against those obtained from current procedures. If there are problems, they should be listed and presented to the vendor or package support personnel.

8.4 Parallel Operations

Once the package has been installed, and the initial checks described above have been made, the package should be used to process current transactions. Until package reliability is absolutely assured, current procedures should continue concurrently. Package output should be compared to the results obtained from current procedures, and all discrepancies should be noted. It is best to check every file that has processed a transaction so that all mistakes are caught.

Later, mistakes can be traced to determine whether they were due to the new package or to operator or user error. All mistakes not attributed to human error should be referred to the vendor; they will help the vendor trace the problems that caused the errors.

It is customary to conduct parallel operations for several cycles, until at least one entirely satisfactory cycle has been completed. If organizational cycles are unusually short or long (shorter than two weeks or longer than a month) the package's processes should be closed out biweekly to create an artificial cycle. If the data being processed is absolutely critical, as with accounting or payroll programs, longer parallel operations may be called for.

All functions should be invoked at the end of every cycle, including those that are only used annually or quarterly (e.g., production of W-2 tax reporting forms), and output should be checked carefully.

Parallel operations, as can be seen, place unusual demands upon an organization's resources and thus require careful planning. If temporary personnel will be hired to assist in conducting parallel operations, arrange for this before the package is installed. To the greatest degree possible, permanent employees should be devoted to working on the new software package so that they can become familiar with its use. However, there should be enough monitoring of temporary personnel to insure that former procedures are being conducted correctly.

If problems are found, they may affect file data, so all system files should be checked for accuracy before current procedures are discontinued.

8.5 Package Acceptance

After testing has been completed, all problems identified should be satisfactorily resolved by the vendor. The contract should be reviewed to insure that the package is in total conformance with the specifications, which can be used as a checklist of minimum package performance.

When the package is formally "accepted," the vendor has contractually fulfilled all its installation obligations. Before accepting the package, it should be deemed adequate in all respects and information should be correctly processed in all fields. It might be helpful to draft a set of acceptance criteria before the package is installed to be sure that all important factors are considered.

If vendor performance has been less than satisfactory, the package should not be accepted. If the vendor proposes to deal with any problems at a later date, it should be understood that acceptance will be deferred accordingly. If the vendor absolutely will not, or cannot, bring the package within specifications, any financial penalties the contract allows should be invoked. If the vendor's performance is still unsatisfactory, it is often helpful to contact the firm that developed the package; they may have a technical understanding of the package that the vendors cannot match, or may be able to exert pressure on a vendor/distributor that will result in better vendor performance. Beyond this, sanctions and remedies should be sought through the legal system.

8.6 Conclusion

This report presented a framework for software package evaluation, selection, and installation. Component tasks were described, along with the activities to be performed at each step of the process and the decisions to be made. If the selection process is systematically planned and executed, and key personnel -- managers, technical people and users -- are actively involved at each step, the result should be the effective, satisfactory incorporation of the software package into the organization.

APPENDIX I

STEPS IN THE PACKAGE SELECTION PROCESS

STEP ONE : REQUIREMENTS ANALYSIS

1. Formation of the requirements analysis team
2. Definition of current procedures
3. Identification of constraints
4. Estimation of package life

STEP TWO : THE REQUIREMENTS DOCUMENT

1. Development of functional specifications
2. Documentation of requirements

STEP THREE : IDENTIFICATION OF CANDIDATE PACKAGES

1. Identification of candidate packages
2. Selection of packages for intensive analysis
3. The make-or-buy decision

STEP FOUR : ASSESSMENT OF SUPPORT NEEDS

1. Documentation review
2. Modification support review
3. Installation support review
4. Training support review
5. Maintenance support review

STEP FIVE : PACKAGE SELECTION

1. Solicitation of proposals
2. Proposal evaluation
3. Package quality evaluation
4. Vendor evaluation
5. Support evaluation
6. Cost-benefit analysis
7. Final package selection

STEP SIX : CONTRACT NEGOTIATION

1. Package contract negotiation
2. Support services contract negotiation
3. Specification of performance guarantees
4. Determination of compensation arrangements

STEP SEVEN : PACKAGE INSTALLATION

1. Formation of the installation team
2. Identification of installation issues
3. Package customization
4. Employee training
5. Transition to new package/procedures
6. Implementation of back-up procedures
7. Post-installation performance evaluation

STEP EIGHT : PACKAGE TESTING

1. Identification of test goals
2. Performance of package testing
3. Package acceptance

APPENDIX II

RELATED ICST DOCUMENTS

NBS Special Publication 500-75, "Validation, Verification, and Testing of Computer Software," W.R. Adrion, M.A. Branstad, and J.C. Cherniavsky, Feb. 1981. (1)

NBS Special Publication 500-87, "Management Guide for Software Documentation," A.J. Neumann, Jan. 1982. (2)

NBS Special Publication 500-93, "Software Validation, Verification, and Testing - Technique and Tool Reference Guide," P.B. Powell, ed., Sept. 1982. (2)

NBS Special Publication 500-98, "Planning for Software Validation, Verification, and Testing," P.B. Powell, ed., Nov. 1982. (2)

NBS Special Publication 500-114, "Introduction to Software Packages," S. Frankel, ed., April 1984. (2)

NBS Special Publication 500-131, "Guide for Selecting Microcomputer Data Management Software," C.L. Sheppard, Oct. 1985. (2)

FIPS PUB 38, "Guidelines for Documentation of Computer Programs and Automated Data Systems," Feb. 1976. (1)

FIPS PUB 101, "Guideline for Lifecycle Validation, Verification, and Testing of Computer Software," June 1983. (1)

FIPS PUB 105, "Guideline for Software Documentation Management," June 1984. (1)

(1) Available from

National Technical Information Service
5285 Port Royal Road
Springfield, VA 22161
(703) 487-4780

(2) Available from

Superintendent of Documents
U.S. Government Printing Office
Washington, DC 20402
(202) 783-3238

APPENDIX III

REFERENCES AND RELATED READING

"A Buyer's Guide to Software Acquisition," Datapro Reports on Software, Datapro Research Corp., Delran, NJ, April 1986.

I. Brownstein and N.B. Lerner, Guidelines for Evaluating and Selecting Software Packages, Elsevier Science Publishing Co. Inc., New York, NY, 1982.

G. Durbin and F.A. Romberg, "Should I Buy Applications Software or Develop It In-House?," Computerworld, March 10, 1986.

A.B. Ferrentino, "Making Software Development Estimates 'Good'," Datamation, Vol. 27, No. 10, Sept. 1981.

Management Evaluation of Software Packages, QED Information Sciences Inc., Wellesley, MA, 1985.

J. Martin, Application Development Without Programmers, Prentice-Hall Inc., NJ, 1982.

E.F. Miller and W.E. Howden, Tutorial: Software Testing and Validation Techniques, IEEE Computer Society, 1978.

G.J. Myers, The Art of Software Testing, Wiley, New York, 1979.

W.E. Perry, A Structured Approach to System Testing, QED Information Sciences Inc., Wellesley, MA, 1983.

W.E. Perry, How to Buy Software for Personal Computers, QED Information Sciences Inc., Wellesley, MA, 1985.

W.E. Perry, How to Test Software Packages: A Step-by-Step Guide to Assuring They Do What You Want, John Wiley & Sons Inc., New York, NY, 1986.

D.J. Reifer, Tutorial: Software Management, IEEE Computer Society, 1985.

L. Scharer, "Pinpointing Requirements," Datamation, Vol. 27, No. 4, April 1981.

C. Shamlin, A User's Guide for Defining Software Requirements, QED Information Sciences Inc., Wellesley, MA, 1985.

G.M. Weinberg and D.P. Freedman, "Reviews, Walkthroughs, and Inspections," IEEE Trans. on Software Engineering, Vol. SE-10, No. 1, Jan. 1984.

L. Wilson, "The DO's and DON'Ts of Documentation," Datamation, Vol. 27, No. 10, Sept. 1981.

M. Yonda, "Negotiating Computer Contracts," Computerworld Focus, February 19, 1986.

U.S. DEPT. OF COMM. BIBLIOGRAPHIC DATA SHEET <i>(See instructions)</i>	1. PUBLICATION OR REPORT NO. NBS/SP-500/144	2. Performing Organ. Report No.	3. Publication Date November 1986
4. TITLE AND SUBTITLE Computer Science and Technology: Guidance on Software Package Selection			
5. AUTHOR(S) Sheila Frankel, Editor			
6. PERFORMING ORGANIZATION (If joint or other than NBS, see instructions) NATIONAL BUREAU OF STANDARDS DEPARTMENT OF COMMERCE Gaithersburg, MD 20899		7. Contract/Grant No.	8. Type of Report & Period Covered Final
9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (Street, City, State, ZIP) Same as item 6.			
10. SUPPLEMENTARY NOTES Library of Congress Catalog Card Number: 86-600593 <input type="checkbox"/> Document describes a computer program; SF-185, FIPS Software Summary, is attached.			
11. ABSTRACT (A 200-word or less factual summary of most significant information. If document includes a significant bibliography or literature survey, mention it here) This report describes a systematic procedure for identifying and evaluating off-the-shelf software packages, and for incorporating the selected package into the organizational environment. Its purpose is to enable the layperson to choose and implement software packages with a minimum of dependence on technical personnel. The report provides guidance on each phase of the package selection and implementation process.			
12. KEY WORDS (Six to twelve entries; alphabetical order; capitalize only proper names; and separate key words by semicolons) applications software packages; off-the-shelf software; packages; software applications; software packages; software package evaluation; software package selection.			
13. AVAILABILITY <input checked="" type="checkbox"/> Unlimited <input type="checkbox"/> For Official Distribution. Do Not Release to NTIS <input checked="" type="checkbox"/> Order From Superintendent of Documents, U.S. Government Printing Office, Washington, D.C. 20402. <input type="checkbox"/> Order From National Technical Information Service (NTIS), Springfield, VA. 22161		14. NO. OF PRINTED PAGES 121 15. Price	



**ANNOUNCEMENT OF NEW PUBLICATIONS ON
COMPUTER SCIENCE & TECHNOLOGY**

Superintendent of Documents,
Government Printing Office,
Washington, DC 20402

Dear Sir:

Please add my name to the announcement list of new publications to be issued in the series: National Bureau of Standards Special Publication 500-.

Name _____

Company _____

Address _____

City _____ State _____ Zip Code _____

NBS *Technical Publications*

Periodical

Journal of Research—The Journal of Research of the National Bureau of Standards reports NBS research and development in those disciplines of the physical and engineering sciences in which the Bureau is active. These include physics, chemistry, engineering, mathematics, and computer sciences. Papers cover a broad range of subjects, with major emphasis on measurement methodology and the basic technology underlying standardization. Also included from time to time are survey articles on topics closely related to the Bureau's technical and scientific programs. Issued six times a year.

Nonperiodicals

Monographs—Major contributions to the technical literature on various subjects related to the Bureau's scientific and technical activities.

Handbooks—Recommended codes of engineering and industrial practice (including safety codes) developed in cooperation with interested industries, professional organizations, and regulatory bodies.

Special Publications—Include proceedings of conferences sponsored by NBS, NBS annual reports, and other special publications appropriate to this grouping such as wall charts, pocket cards, and bibliographies.

Applied Mathematics Series—Mathematical tables, manuals, and studies of special interest to physicists, engineers, chemists, biologists, mathematicians, computer programmers, and others engaged in scientific and technical work.

National Standard Reference Data Series—Provides quantitative data on the physical and chemical properties of materials, compiled from the world's literature and critically evaluated. Developed under a worldwide program coordinated by NBS under the authority of the National Standard Data Act (Public Law 90-396).

NOTE: The Journal of Physical and Chemical Reference Data (JPCRD) is published quarterly for NBS by the American Chemical Society (ACS) and the American Institute of Physics (AIP). Subscriptions, reprints, and supplements are available from ACS, 1155 Sixteenth St., NW, Washington, DC 20056.

Building Science Series—Disseminates technical information developed at the Bureau on building materials, components, systems, and whole structures. The series presents research results, test methods, and performance criteria related to the structural and environmental functions and the durability and safety characteristics of building elements and systems.

Technical Notes—Studies or reports which are complete in themselves but restrictive in their treatment of a subject. Analogous to monographs but not so comprehensive in scope or definitive in treatment of the subject area. Often serve as a vehicle for final reports of work performed at NBS under the sponsorship of other government agencies.

Voluntary Product Standards—Developed under procedures published by the Department of Commerce in Part 10, Title 15, of the Code of Federal Regulations. The standards establish nationally recognized requirements for products, and provide all concerned interests with a basis for common understanding of the characteristics of the products. NBS administers this program as a supplement to the activities of the private sector standardizing organizations.

Consumer Information Series—Practical information, based on NBS research and experience, covering areas of interest to the consumer. Easily understandable language and illustrations provide useful background knowledge for shopping in today's technological marketplace.

Order the above NBS publications from: Superintendent of Documents, Government Printing Office, Washington, DC 20402.

Order the following NBS publications—FIPS and NBSIR's—from the National Technical Information Service, Springfield, VA 22161.

Federal Information Processing Standards Publications (FIPS PUB)—Publications in this series collectively constitute the Federal Information Processing Standards Register. The Register serves as the official source of information in the Federal Government regarding standards issued by NBS pursuant to the Federal Property and Administrative Services Act of 1949 as amended, Public Law 89-306 (79 Stat. 1127), and as implemented by Executive Order 11717 (38 FR 12315, dated May 11, 1973) and Part 6 of Title 15 CFR (Code of Federal Regulations).

NBS Interagency Reports (NBSIR)—A special series of interim or final reports on work performed by NBS for outside sponsors (both government and non-government). In general, initial distribution is handled by the sponsor; public distribution is by the National Technical Information Service, Springfield, VA 22161, in paper copy or microfiche form.

U.S. Department of Commerce
National Bureau of Standards
Gaithersburg, MD 20899

Official Business
Penalty for Private Use \$300