OPPORTUNITIES AND STRATEGIES FOR SUCCESS IN THE SOFTWARE PACKAGES INDUSTRY

by

DORIAN PUNJ

S.B., Massachusetts Institute of Technology (1973)

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE at the MASSACHUSETTS INSTITUTE OF TECHNOLOGY September, 1974

Signature of Author .................................................................

Alfred P. Sloan School of Management, June 24, 1974

Certified by ................................................................. Thesis Supervisor

Accepted by ................................................................. Chairman, Departmental Committee on Graduate Students
OPPORTUNITIES AND STRATEGIES FOR SUCCESS
IN THE SOFTWARE PACKAGES INDUSTRY

by

Dorian Punj

Submitted to the Alfred P. Sloan School of Management on June 20, 1974, in partial fulfillment of the requirements for the degree of Master of Science.

ABSTRACT

The purpose of this thesis is to present some techniques for minimizing the risks and possibilities of failure involved in participating in the future growth of the software packages industry.

This thesis presents a thorough overview of the software packages industry, evaluates the present and future markets for proprietary software, and identifies good business opportunities in the industry for entrepreneurs and software vendors. Guidelines are provided for the successful development and marketing of software packages.

Software was divided into four basic groups: "Total" software packages, systems software, standard software, and applications packages. The market for all packages was found to be $175 million in 1973, and estimated to increase to $450 million by 1980.

A questionnaire was sent out to 81 software vendors, to define some of the major problems in the industry, and to define some guidelines for a successful company. Inter-
views were held with consultants, business executives, and magazine editors associated with the field of packaged software.

It was determined that although flexibility of design, documentation, customer support and company reputation are all very important factors, marketing is by far the most crucial, as well as the most expensive factor in determining a software package's success. Successful companies must also be wary of IBM's software policies, and must know how to make their packages financially sound for their customers as well as for themselves.

To help make packages more financially sound, a breakdown of the general costs, as well as an analysis of pricing strategies is presented. Guidelines are provided for marketing in-house software, and software brokers are discussed as an emerging factor in the software industry.

A brief overview of the packaged services industry is also provided, together with a market analysis and growth projections, and some strategies for success.

Thesis Supervisor: John F. Rockart
Title: Associate Professor of Management
ACKNOWLEDGEMENTS

I should like to express my gratitude to my thesis advisor, Prof. John F. Rockart, for his advice and guidance, and especially for urging me to structure and limit my several areas of interest into a manageable task. I must also thank Prof. Stuart E. Madnick and Prof. John J. Donovan for teaching me everything useful that I know about computers.

Special thanks, however, must go to Prof. Donovan, who, by inspiring me to start considering methods for transcending the rat race, sparked my interest in the topics of this thesis.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Abstract</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>4</td>
</tr>
<tr>
<td>Table of Contents</td>
<td>5</td>
</tr>
<tr>
<td><strong>Chapter 1. Introductory Remarks</strong></td>
<td>8</td>
</tr>
<tr>
<td>Purpose of Study</td>
<td>8</td>
</tr>
<tr>
<td>The Problem and its Setting</td>
<td>10</td>
</tr>
<tr>
<td>Methodology</td>
<td>13</td>
</tr>
<tr>
<td><strong>Chapter 2. The Present Market for Software Packages</strong></td>
<td>16</td>
</tr>
<tr>
<td>What is a Software Package?</td>
<td>17</td>
</tr>
<tr>
<td>The Types of Software Packages</td>
<td>18</td>
</tr>
<tr>
<td>&quot;Total&quot; Software Packages</td>
<td>20</td>
</tr>
<tr>
<td>Systems Software</td>
<td>22</td>
</tr>
<tr>
<td>Standard Software</td>
<td>23</td>
</tr>
<tr>
<td>Applications Packages</td>
<td>25</td>
</tr>
<tr>
<td>The Total Market</td>
<td>28</td>
</tr>
<tr>
<td>Breakdown by Users</td>
<td>28</td>
</tr>
<tr>
<td>Custom Work</td>
<td>29</td>
</tr>
<tr>
<td><strong>Chapter 3. The Industry Structure and Competitive Climate</strong></td>
<td>31</td>
</tr>
<tr>
<td>The early Software Industry</td>
<td>31</td>
</tr>
<tr>
<td>Unbundling</td>
<td>33</td>
</tr>
<tr>
<td>The Boom in Software Companies</td>
<td>34</td>
</tr>
<tr>
<td>The Present Industry Structure</td>
<td>35</td>
</tr>
<tr>
<td>Overseas Suppliers</td>
<td>37</td>
</tr>
<tr>
<td>The Future Industry Structure</td>
<td>38</td>
</tr>
<tr>
<td>Chapter</td>
<td>Title</td>
</tr>
<tr>
<td>---------</td>
<td>-------------------------------------------</td>
</tr>
<tr>
<td>4</td>
<td>Survey Results</td>
</tr>
<tr>
<td>5</td>
<td>Market Trends and Growth Projections</td>
</tr>
<tr>
<td></td>
<td>Factors Influencing Market Growth</td>
</tr>
<tr>
<td></td>
<td>A Preview of the Growth Potential</td>
</tr>
<tr>
<td></td>
<td>Growth Projections</td>
</tr>
<tr>
<td></td>
<td>Total Information Systems</td>
</tr>
<tr>
<td></td>
<td>Systems Software</td>
</tr>
<tr>
<td></td>
<td>Standard Software</td>
</tr>
<tr>
<td></td>
<td>Applications Packages</td>
</tr>
<tr>
<td>6</td>
<td>Strategies for Success</td>
</tr>
<tr>
<td></td>
<td>Reasons for Failure</td>
</tr>
<tr>
<td></td>
<td>Requirements for a Successful Company</td>
</tr>
<tr>
<td></td>
<td>Crucial Factors</td>
</tr>
<tr>
<td></td>
<td>IBM</td>
</tr>
<tr>
<td></td>
<td>Sound Economic Sense</td>
</tr>
<tr>
<td>7</td>
<td>Development, Marketing, Maintenance, and Protection</td>
</tr>
<tr>
<td></td>
<td>Development</td>
</tr>
<tr>
<td></td>
<td>Testing</td>
</tr>
<tr>
<td></td>
<td>Documentation</td>
</tr>
<tr>
<td></td>
<td>Marketing</td>
</tr>
<tr>
<td></td>
<td>Installation</td>
</tr>
<tr>
<td></td>
<td>Maintenance and Enhancements</td>
</tr>
<tr>
<td></td>
<td>Protection of Software Packages</td>
</tr>
<tr>
<td></td>
<td>Patents</td>
</tr>
<tr>
<td></td>
<td>Trade Secrets</td>
</tr>
</tbody>
</table>
Chapter 8. The Economics of Software Packaging

Pricing Strategies
Pricing the Package
Breakdown of Costs
A Note on Taxation
Cash Flows

Chapter 9. In-house Software and Software Brokers

Marketing In-house Software
Software Brokers

Chapter 10. A Note on the Packaged Services

Industry
What is a Service
Industry Characteristics
Reasons for Growth
The Market Growth
Facilities Management
Strategies for Success

Chapter 11. Conclusion

Bibliography
Appendix
Chapter 1

Introductory Remarks

Purpose of Study

This thesis has stemmed from what originally started as a market study that the author had undertaken for his own use. The author was bewildered by the confusing nature of the literature, which often contained contradictory statements in different articles. For example, the size of the market for software packages in 1980 has been estimated at figures ranging from as low as $400 million to as high as $3 billion!

Although the literature is replete with guidelines and suggestions for the user on how to purchase proprietary software or deal with suppliers, there is very little that has been offered as help to the supplier in solving some of the severe problems that he faces. For example, although the author has counted 17 different "checklists" for buying software packages, and 8 different articles in the last 3 years on the "make or buy" decision, there is not a single article that advises a software supplier on how to reduce his exorbitant marketing costs!

The purpose of this thesis then, is to present a broad and thorough overview of the software packages industry as
it is today; to evaluate the market and its future growth potential, and to identify lucrative opportunities in the market for software vendors and entrepreneurs to capitalize on; to present some strategies for success in the industry, and investigate the factors that contribute to a package’s success; to supply guidelines for the successful development, marketing, maintenance, and protection of a package, and to assist the vendor in anticipating his costs, and in pricing his package. In sum, the aim of this thesis is to present a sound method for minimizing the risks involved for those who would like to venture into the proprietary software business. Hence the information contained in this report should be of value to:

- Entrepreneurs who want to enter the software packages industry.

- Software vendors who do not want to see their enterprises end in disaster, like those firms that collapsed during recession of 1970.

- Software vendors who are interested in gaining additional revenues through their large marketing forces, by selling software for other firms.

- EDP managers who would like to recover some of their development costs by marketing their in-house software.

- Students who will someday be EDP managers, and need to
be aware of the increasingly important part that software packages will be playing in the world of data processing.

The Problem and its Setting

One-third of all the money spent on data processing is spent on the creation of software. This means that in 1973, American industry spent over $8 billion on this activity, and unfortunately, much of it was duplicated and wasteful.

The data processing profession is finding itself in a situation not unlike that of most industries in the 18th and 19th centuries, that had not yet discovered the virtues of mass production and standardized design. Programmers and analysts are still spending most of their time recreating the ordinary, mundane, and basic programs that industry demands of them, instead of spending their time approaching the truly creative tasks that automation lends itself to.

Moreover, whereas the cost of computer hardware is steadily going down, the cost of software is becoming more and more prohibitive. The high cost of software is being aggravated by the indirect costs that usually arise from slippages in delivery schedules and long lead times for systems, because software is generally on the critical path in overall system development (Boehm, 1973).

One of the responses from the EDP industry to offset this hardware/software cost imbalance is the software package,
the concept of which was officially endorsed by the hardware manufacturers' unbundling action of 1969. The software package industry is now 5 years old, and is growing in importance and size every year. Data processing managers have become more and more conscious of the costs of developing software, and management has become more demanding in justifying the cost of the computer center, and is looking at software in terms of its costs and benefits. Together with searching the market for the best buys for replacing disks, tapes and core, DP managers have become increasingly aware of software packages as a cheaper, more effective, more reliable, and faster means of getting programs up and running, and also as a means of reducing the vast amount of redundancy and duplication of effort that exists in the creation of software.

The market place today has about 800 software vendors, supplying over 3000 software packages. About one-third of these are successful (but not necessarily large) firms, with quality packages and good reputations. Another third of these firms are incidental sellers, and the rest of the firms are either very new, or possibly on their way to becoming successful (Welke, 1973).

The strong software sellers are characterized by the fact that they have a good idea of what it takes to survive in the software business. They are determined and persistent, over-document and over support their software
beyond any customer's needs, and market their product so aggressively, that sometimes 80% of their personnel will be involved in sales. Their software is dynamic, constantly evolving, and constantly improving. Aware of their huge investments in development and marketing, they are typically hard-nosed about their prices. They protect their software jealously, but in the end the user is satisfied. Successful software suppliers like these have produced at least 65 packages that have more than a million dollars in sales.

The incidental sellers will always be incidental sellers. They usually start selling software almost by accident. They include consultants that have developed software for a customer and find that they can sell it to other clients; the data processing centers of many large firms, such as banks or insurance companies; software houses that do "custom work" and sell the software that they have developed; research organizations and universities that receive inquiries from industry about their projects; users that have developed special applications packages; and computer service agencies that have developed special packages for their users. Often it is these incidental sellers that do not care to support or maintain their packages, as they are charging low prices in relation to the value of the software. Some of them are responsible for having given the packages industry a bad name, and consequently have made the job tougher for the other
more responsible suppliers.

Then there are the small, ambitious, striving companies. Often these firms consist of programmers that have broken away from EDP departments after working on a system, analysts that have finished working on a government funded project, or consultants that have decided to try their hand at selling packages. In many cases, the president of the firm has written the program, and the company has been formed to sell it. The founders of these companies are attracted into the business by the lure of instant riches: they know they can easily sell 2 packages at one-half of development costs, and then anything that they sell is pure profit. What they do not know is that because they have underestimated a number of costly factors, many of these firms will end in disaster, like the hundreds of software firms that have already folded. This report is for them. The risks in the software business can be minimized; the market is predictable, the causes of failure can be identified, and all the phases from development to marketing and installation are controllable. Once an opportunity has been spotted, a well thought out strategy can almost guarantee success.

Methodology

A number of methods have been used to compile, analyze, and present in a readable fashion, all available, relevant, and valuable information on the proprietary software industry.
They included an extensive review of the literature, annual reports of software companies, and news articles. A questionnaire was mailed out to 81 successful software companies, the purpose of which was to elicit from either their presidents or chief marketing officers the nature of the crucial problems faced by suppliers, the factors they consider to be most important to a software vendor's success, and their estimates of the future size of the market. The author obtained access to data compiled by the market research firms of Frost and Sullivan, and International Data Corp. Information was gathered from the U.S. Department of Commerce and other government agencies. Interviews were held with the software editors of Computerworld and Modern Data, and visits were made to two software "brokerage" firms in the Boston area. Interviews and talks were held with four software company executives and consultants in the Boston area to provide the author with the "feel" of the industry that he lacked prior to this research.

Chapter 2 discusses the present market for software packages, and presents a system for classifying the various types of packages. Chapter 3 discusses the industry structure and competitive climate, and analyzes the major factors that influence this structure.

Chapter 4 is a compilation of the results of the data that was gathered from the mailed questionnaires. Chapter 5
presents and justifies some market trends and growth projections, and investigates the factors that will influence this growth.

Chapter 6 discusses some general strategies for success in the industry, while chapter 7 analyzes in greater detail the specific problems of development, marketing, maintenance, and protection of software packages. Methods of determining pricing strategies and costs are provided in chapter 8, together with a formula that can be used to price a package. Guidelines are offered to the EDP manager who is interested in marketing his software, together with some insights into the brokerage function, in chapter 9.

Lastly, because much of the data was found together with the data on software packages, an overview of the packaged services industry is presented, complete with a breakdown of the present market, growth projections, and strategies for success. In the conclusion, some suggestions for the survival and growth of the industry as a whole are made, together with some possibilities for future research.
Chapter 2

The Present Market for Software Packages

When the purpose is to add more features to systems packages, or simply to consider the use of applications packages, one should consider the use of applications packages to get a job up and running quickly and cleanly, because the data processing manager has hundreds of software packages from which to choose. He is regularly inundated with brochures sent in the mail, or advertisements like Computerworld, placed by a wide variety of firms.

Software packages range in price from less than $100 to one million dollars, the variance reflecting the utility of available software, ranging from simple subroutines to integrated management information systems. However, the selling price may also reflect not only packages of installation and support available with the package, but also the disparity in prices that are sometimes charged for similar packages by different vendors.

Large firms are offered by the EDP departments of companies, firms at bargain basement prices, because the company ultimately gains additional revenues from a similar investment it has already made. On the other hand, a package can be offered at a price up to twenty percent simply by exaggerating the abilities of the product. (Datamation, April 1973, p. 119).
With all the confusing shapes and forms that packages come in, buyers often have a difficult time determining which packages best meet their requirements - depending on whether they have successfully identified their needs or not! It often happens that it is not till after meeting with a number of different software salesmen that a user develops a clear picture of what his precise needs are. It is the purpose of this chapter to familiarize the reader with the various types of software packages currently available, their uses, their present markets, and any problems currently associated with them.

What is a Software Package?

A software package can be described as a well-defined computer program, or a group of programs, that are designed to satisfy a general data processing need that many firms may have simultaneously (Snyders, 1973a). The package may be completely usable as is, or may have to be modified to meet the needs of a particular user.

But a complete software package is also much more than a few programs put together. Support, documentation, and the type of contract involved are as much an integral part of the package as is the actual functional program. Support includes installation of the package, training of personnel to use it effectively, as well as updating and maintaining the programs and concepts involved. The documentation provided ideally includes a user's manual, inst-
uctions on modifying the programs, and descriptions of file layouts and special features (Cowles, 1971). The type of contract can range from an outright purchase to leasing only the object code from the supplier, or perhaps only having access to the package through a service bureau.

The Types of Software Packages

There have been many attempts in the literature to categorize the various kinds of software packages, but many of these classifications have proved inadequate. The ICP Quarterly, which provides a listing of all available packages, has already changed its classification scheme several times. The most common classification for software has been into two categories - utility programs and applications packages. According to this classification, utility programs provide the tools that are necessary for the data processing function, such as compilers, data base systems, and aids to programming and flowcharting. On the other hand, applications packages deal with the functional operation of an enterprise, such as payroll, inventory handling, and scientific applications.

However, these two categories are too broad for the hundreds of packages in existence today. For the purposes of the evaluations to be made in this report, it is convenient to classify all software into one of four major functional categories:

1. **Total Information Systems** consist of complete
complete software systems. These include complete packages for a specialized industrial application, such as a medical information system for hospitals, or an entire information processing system for small to medium sized businesses.

2. **Systems Software** consist of packages whose purpose is to improve on capabilities that are normally supplied by the computer vendor. These packages include operating systems, extensions and supplements to operating systems, generalized data base management systems and plug-to-plug replacements for computer manufacturer's software.

3. **Standard Software** includes the whole spectrum of I/O modules, project management and educational tools, programming aids, and systems programming elements such as compilers and assemblers.

4. **Applications Packages** can be described as programs that bridge the gap between a user's needs and the capabilities of his equipment. Their functions cover a broad range, from general business to scientific and engineering applications. **Functional applications packages**, such as payroll programs, can be used across industries, whereas **industry-specific packages** are usually limited in scope, and the majority of them were probably developed as contract work and only marketed as an afterthought.
"Total" Software Packages

This is perhaps the fastest growing area of proprietary software, especially with the increased use of minicomputers for applications such as process control and communications. Prior to 1968 the market for total software packages was almost zero, but in 1973 net sales of this kind of software approached $20 million. Many of these systems are intended for industry-specific functions, and the majority of them are implemented on minicomputers. There are two basic methods by which package vendors supply total software packages:

In the first method, the package is supplied via a complete turnkey system, where the software supplier also provides all the necessary hardware. The hardware is usually dedicated to perform the particular function. Examples of dedicated hardware are fairly widespread in communications applications, such as front ends, message switching devices, and intelligent terminals.

In the area of process control, minicomputers are taking over an increasing number of functions as manufacturing plants become more and more automated. In many cases, a package originally intended for a specific industrial application can be readily converted for similar applications in other industries. Innovative applications of turnkey systems, such as in computerized typesetting, is causing the market to expand quite rapidly.
Turnkey systems are also being used for the traditional business applications. For example, the BASIC/4 system is marketed as a complete package with all the hardware included. These systems are becoming increasingly popular because little or no in-house professional programming is required to use them effectively.

The second method of supplying total software systems occurs when manufacturer's hardware is used, but the software vendor supplies all the programs on the machine, so that again no programming need be done by the user. The popularity of these systems stems from the fact that whereas many industries can benefit from automation, few of them can afford to (or want to, for that matter) properly staff and equip an EDP department. Hence some of these packages are supplied in a manner that borders on either a service or facilities management.

The more common examples of these systems are in small business accounting, banking and credit systems, and medical systems in hospitals or doctor clinics for automating the rather substantial amount of clerical functions.

The EDP departments of major firms have also contributed to the supply of total software systems. They have realized that they can recoup some of the tremendous costs of developing their highly specialized applications, provided there are no undue effects from selling software to competitors. For example, although it was advantageous for Eastern Airlines to
market its sophisticated Airlines Reservations Package, it seems unlikely that AVIS would sell its "WIZARD" system to a competitor.

**Systems Software**

There are only a few independent suppliers that sell systems software because of the high costs of installation and post sales support that is usually required for this kind of software. However, this market has been very profitable for the few suppliers in this field because of the high prices that systems software commands, and because users are understandably very hesitant to develop their own systems software.

In 1973, the total sales of systems software was about $45 million. It must be remembered that this figure also includes the software that users typically received for free before the unbundling announcements of 1969. A large part of this figure is also accounted for by the increase in the interest for data base management systems in recent years.

The operating systems that are offered by independents are plug-to-plug enhancements or even replacements for manufacturer's operating systems, that perform as an improvement on capabilities for a cheaper price. Most of these are implemented on minicomputers. In some cases, alternate operating systems are offered that provide a new set of features that are required for a particular type of application. Operating system emulators are also popular, which provide the
user with the capability of running programs from an older machine on a machine that the user has upgraded to - saving the user the time, money and headaches of converting an entire library of old programs.

Data base management systems (DBMS), are basically enhancements to the computer's operating system, and are designed to handle the creation and usage of large central data bases. The DBMS's primary concern is the means of storing and accessing of data. Thus a DBMS is an access method coupled with a specific, highly structured and chained file organization (Falor, 1971). DBMS packages are usually developed to be compatible with most programs written in either assembly or higher level languages.

The majority of DBMS packages are sold by computer mainframers, such as IBM's IMS and Honeywell's IDS, but several independents have also broken into this field. One of the lures is the high prices that users are willing to pay for a DBMS. For example, Compress Inc.'s DART-1, which is a data management system similar to IMS (but requiring less equipment) sells for about $60,000.

Standard Software

The standard software packages industry is the most highly developed of the four types of package industries. In 1973 the total sales of standard software was about $35 million, broken down as follows:
Sub-systems Software $12.0 million
Communications & Teleprocessing 7.5
Programming Tools 9.0
Simulators 4.0
Performance Measurement 2.5
Total $35.0 million

Sub-system software encompasses a wide assortment of spooling routines, access methods, file management systems and report generators, I/O handlers, utility programs, compilers and assemblers, tape library systems, information retrieval systems, and a host of programming aids.

File management systems must not be confused with database management systems. File management systems and report generators basically select data from and update files, and then format it into reports. They provide lower level programmers and even non programmers with higher level language capabilities with substantial reductions in programming complexity (Falor, 1970b).

Utility programs are usually smaller sized programs that perform simple functions like sorts, file dumps, disk to tape transfers, and other utility functions.

Programming aids include conversion, translation and simulation programs; debugging and testing programs; documentation and flowchart producing aids; job control producing programs; preprocessors for decision tables and shorthand language interpreters (Falor 1970a). They are used to help programmers create programs better, faster, and at less cost.
Performance measurement programs analyze the utilization of hardware and software resources in either operational or simulated environments. Communications and teleprocessing monitors are systems that control and support data communications and remote job entry functions.

Standard software has been very popular with software vendors as it is fairly easy to produce, and the need for it is general and not industry-specific. Many of the one-man companies and other amateur organizations are providers of standard software, and many EDP departments have been tempted to market the tools they have developed. Hence more care must be exercised in purchasing standard software.

Applications Packages

By far the largest sector of software products consist of applications packages. Although applications packages were not very popular immediately after the unbundling announcement because of the large amount of inferior software being marketed, user resistance to these types of packages is rapidly diminishing as they become aware of the many advantages of using them. This is evidenced by the fact that sales of applications packages jumped to a record $75 million in 1973 from a level of $55 million in 1972.

Considering their variety of uses, applications packages are surprisingly similar in their basic functional structure. The vast majority are essentially record keeping devices to handle large quantities of data. The processing centers around
manipulating and updating user-defined files that are created according to package specifications. This usually involves computations of varying complexity, ranging from simple additions to complex simulations and statistical calculations (Cowles, 1971).

Packages that perform general functions have proved to be profitable for software vendors as the market is not limited to any specific industry. However, there is also a lot of competition in functional packages which tends to keep prices fairly low. These functional packages include

- General business packages such as payroll, accounting, inventory control, billing, budgeting, costing, etc.
- Sales analysis and market research programs, which are becoming a "must" for marketing executives.
- Project control packages such as PERT and other scheduling programs.
- Educational programs for instruction in data processing techniques.
- Design and engineering applications. These programs are used very heavily on time-sharing facilities for purposes ranging from process simulations to stress analysis calculations.
- Scientific and mathematical routines which are often bought to enhance existing library programs.
Mailing applications such as magazine subscriptions lists, labelling, sorting and duplicating.

Industry-specific packages usually require a large amount of custom work. The total number of packages sold are usually much fewer than for standard software or functional applications packages. Except for the banking industry, the competition is not as strong and hence individual markups on packages is higher. The following industries are the major users of industry-specific packages:

- The banking industry is by far the largest user of applications packages, because of the large amount of clerical work that is involved. Banking packages include demand deposit accounting, loans and mortgage packages, general ledger systems, bond and portfolio analysis and credit card systems. Vendors often provide "custom" systems which simply involves putting together a number of different packages.

- Brokerage industry: Stock transfers, shareholder accounting packages.

- Manufacturing: Inventory control, bill of materials processors, production planning.


- Transportation: Route management and Vehicle management.

- Real Estate: Property management, financial analysis.
Other major users of industry-specific packages include the construction, distribution, petroleum, and chemical industries, the medical and legal professions, and insurance companies, hotels, and libraries.

The Total Market

In sum, the following tables show the market sizes of the software package industry in 1972 and 1973, that has grown from almost zero in the 1960's.

<table>
<thead>
<tr>
<th></th>
<th>1972 ($ million)</th>
<th>1973 ($ million)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Software Packages</td>
<td>12</td>
<td>20</td>
</tr>
<tr>
<td>Systems Software</td>
<td>38</td>
<td>45</td>
</tr>
<tr>
<td>Standard Software</td>
<td>31</td>
<td>35</td>
</tr>
<tr>
<td>Applications Packages</td>
<td>54</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>135</td>
<td>175</td>
</tr>
</tbody>
</table>

It can be seen that the market grew by a phenomenal 30% in 1973! A recent survey of data processing budgets showed that DP managers had budgeted for a total of $250 million for purchasing software packages in 1974 (McLaughlin, 1974) - an increase of 43% over 1973!! Software payments to hardware manufacturers were increasing by 50%, while payments to independents were increasing by over 100%. A complete analysis of the future growth of the market for software packages is presented in chapter 6.

Breakdown by Users

In a survey reported in Computerworld, (Leavitt, 1973a), 80% of all users of EDP equipment had leased or purchased some form of packaged software. Of these users, 61% had
acquired applications software, 51% had bought systems software, 51% had acquired programming aids, but only 12% had obtained plug to plug replacements for their manufacturer's software. The average expenditures represented about 2.75% of a typical user's total EDP budget.

Custom Work

In the same survey, about 63% of those that bought packages said that they had changed or customized their packages. The remainder bought their packages "as is".

Many of the packages that are only provided "as is" are modifications to operating systems, or complex applications packages, that are difficult to change. Of those that can be customized, modifications may range from slight output format changes to major redesigns in program logic.

Sometimes minor modifications can be made directly by the user via a series of control cards. On the other extreme, some packages are only skeletons of mainline logic, which are filled in with the particulars of logic to meet the user's specific needs. Other highly customized jobs are simply precoded modules to provide the custom effect. The vendors of these heavily customized systems typically have a background of consulting work in the areas covered by their products. These vendors are usually not
generalists, and their follow up support usually reflects their concern for clients (Leavitt, 1973b).
Chapter 3

The Industry Structure and Competitive Climate

Before the mid-1960's, all software was either developed in-house by users, or produced by computer manufacturers. Although the quality of manufacturer's hardware was constantly being improved, few real innovations in programming techniques were being made, resulting in poor and unsatisfactory programs that did little to exploit the capabilities of the third generation hardware that was being introduced about this time. Users were becoming increasingly dissatisfied with the results of their expensive programming efforts, and always had the troubling thoughts that they were duplicating what some other EDP department, somewhere, had already accomplished.

The early Software Industry

From 1965 through 1968, a few firms with experienced personncl and the necessary resources became involved in developing software packages for the express purpose of selling them to other firms. The first packages sold were very basic, with general applications such as billing and payroll. More complex, (usually industry-specific) packages evolved from consulting or contract work, and were then marketed as an afterthought. Some packages were developed by businesses who then sold the marketing rights to software firms. Several of the first proprietary software products were enthusiastically accepted by the marketplace,
which was an encouraging sign for software developers.

During the late sixties, more firms entered the market, many of them being 5 or 10 man enterprises producing one or two packages. Packages became more general, with less dependency on any specific mainframe manufacturer's machines. More software was developed with the specific intent of being marketed, providing for more consistency in documentation, standardization, and better testing than before. The software industry was starting to show some promise with a good potential for growth. Development costs were low, and production costs almost zero (Goetz, 1972a).

However, there were some hitches. IBM and other mainframers supplied all their software "free" with their hardware. Few users were willing to purchase software from independent vendors when they could get it free from IBM, even when the independent's software showed a considerable improvement over the manufacturer's. This "bundling" of hardware and software was very favorable to the mainframers, as it assured hardware sales for mainframers that had good software support (eg. IBM), and allowed them to charge higher prices for their hardware because of the free software.

There are many examples like the case of the west coast firm, which in 1964 built a better compiler for an IBM 7040. However, users could get the inferior compiler from IBM, and no users could be found that were willing to pay cash for real cost savings in improved performance, causing the firm to fold (Oyer, 1970).
unbundling

On June 30th 1969, because of increasing pressure from several sources, including the Justice Department because of the antitrust issues involved, IBM announced that it was "unbundling" its software. Theoretically, at least, software would no longer be free but would be priced and sold separately from IBM hardware. As a consequence, hardware prices dropped - but only by about 3% (Oyer, 1970). For most users, IBM's historic announcement meant higher prices for their computer and its support services, because IBM now charged extra fees for education and training, systems engineering services, customer software and consulting support, basic software (such as language compilers), and applications programming products. For some, the price increases ran as high as 50%.

There are other reasons why unbundling actually favored the hardware manufacturers. When users evaluate computer hardware, the availability of high-quality software is a key consideration. Unbundling provides independents with an incentive to produce quality software for the unbundled mainframer's machines, since they are usually reluctant to invest in development of software packages that may face competition from a mainframer's free software. Thus, a bundled company may actually lose hardware sales because of the poor quality of its bundled software (Goetz, 1972).
The Boom in Software Companies

Users began to look for alternatives to IBM's expensive software. Entrepreneurs reacted to the unbundling announcement with amazing speed. The lure of instant wealth, with seemingly infinite revenues and negligible costs of production, coupled with no fear of free competitive products from IBM, was so strong that by the end of 1969 well over 100 companies were founded specifically to exploit the infant software packages market! Some of these companies were large, well financed firms, such as the subsidiaries of the large aerospace companies. But many of these companies were one-or-two man businesses operating out of their homes, causing a proliferation of poor quality software packages on the market. Many firms were underfinanced, and often underestimated the need for a strong marketing force and the total cost for effectively marketing a software package. The recession of 1970 brought about financial ruin for many of these companies that were financed by private or public stock issues, forcing them either to fold, or to merge with more successful firms.

The ease of entry into the software package industry can be attributed to the apparent simplicity of producing software. An individual can write a program in his spare time, rent computer time to debug it, and put it on the market. If it sells, he has all the makings of a successful enterprise. Moreover, data processing personell can easily break away from their employers and become vendors by selling a program that they developed for in-house processing.
The firms that survived the trial by fire in 1970 produced a variety of packages. Not only were they developing more sophisticated packages, but they were making software available for areas that had never been automated before, such as the investment function for banks. A large number of packages were also being produced for use within data processing itself, such as project control systems and job accounting packages. As user resistance to packages began breaking down, the boom in applications packages was started. Independents also became convinced that there was a future in total software packages, because of the successful experiences of large EDP departments that had sold their systems.

The Present Industry Structure

Today there are over 800 independent companies with some portions of their revenues coming from the sale of software packages. Only a small percentage of these firms rely on software packages for all of their revenues, since they consider it to still be a risky business.

The largest vendors of software are still the mainframe manufacturers, with IBM being by far the largest supplier of proprietary software. Giant IBM controls over 40% of the market, although the approximately $70 million it received in revenues from packaged software in 1973 are not a very significant part of IBM's total revenues. NCR's fastest growing divisions are its software package and service operations, while Burroughs specializes in banking and financial packages. Digital Equipment Corp. has recently started to
greatly increase the amount of software that it is providing with its machine, because as hardware prices continue to drop, marketing software is becoming much more profitable.

The next largest vendors of software are the very large independent software firms such as Cordura, Automatic Data Processing, Electronic Data Systems, and the large EDP subsidiaries of the large aerospace firms such as Boeing Computer Services and McDonnell Douglas Automation. However, the ten largest independent software firms receive most of their revenues from services such as time sharing and facilities management. Only an average of 12% of their total billing results from packaged software sales (Seligman, 1972). For example, Ross Perot's Electronic Data Systems receives most of its revenues from facilities management for the brokerage industry. It should be noted that size does not necessarily guarantee that a firm will survive or that it produces high-quality software. Of the twenty largest software firms in late 1969, only twelve were still in the software business in 1972 (Seligman, 1972).

The World-Wide Directory of Computer Companies includes a cross reference of all companies that sell software, from which the following data on the distribution of the various sizes of the remaining companies in the software industry was derived:

<table>
<thead>
<tr>
<th>SALES</th>
<th># of Companies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Under $100,000</td>
<td>20%</td>
</tr>
<tr>
<td>$100,000 - $500,000</td>
<td>27%</td>
</tr>
<tr>
<td>$500,000 - $1 million</td>
<td>12%</td>
</tr>
<tr>
<td>$1 million - $5 million</td>
<td>28%</td>
</tr>
<tr>
<td>$5 million +</td>
<td>-36-</td>
</tr>
<tr>
<td></td>
<td>13%</td>
</tr>
</tbody>
</table>
Almost 35% of these firms employ 10 or fewer persons. Although 30% of them employ from 26 to 50 people each, few of them break over the 50 employee mark. It seems that these companies find their hands very full at this point, where the new packages they are developing are just enough to offset the old packages that are no longer selling successfully.

It is many of the small firms, with less than 10 employees that often depend on software package sales for all of their revenues. These are the firms that find it difficult to survive a depression in the economy, because there are many risks associated with being dependant on only software packages, as is explained in chapter 8.

A number of major users are also suppliers of software. This is hardly surprising when one considers that some users spend over a million dollars a year on EDP. The software is often marketed in a casual manner, (in some cases without even top management's knowledge), at prices that are usually cheaper than that charged by independents. A common practice is for users to sell the distribution rights of their software to independent companies. This method seems to benefit all the parties concerned: The company gets a return on its package development costs without going into the software business; the distributing party receives a commission on all sales; and the buyers can purchase software cheaper than they can produce it themselves.

Overseas Suppliers

A number of overseas companies have attempted to break into
the U.S. software packages market, because they know that a
tlarge part of the cost of developing software is paid out in
salaries, and the cost of paying professionals in Asia and
some parts of Europe is considerably lower than in the U.S.
For example, IBM developed a large portion of its 360 and 370
software overseas for this reason.

However, there is little or no threat of foreign firms
gaining a larger share of the American software packages mar-
et. As is explained in chapter 8, the costs of development
are small when compared to the costs of marketing software.
The success of any software package depends very much on the
ability to market it, and hence the only area that overseas
firms are likely to be a factor is in contract programming
and custom work. In fact, because of the greater experience
that U.S. firms have in developing and marketing software,
European users will most likely be swamped with U.S. made
software packages in the not too distant future.

The Future Industry Structure

As it becomes more mature, the software industry will
have to make some adjustments. There will be a great reduc-
tion in the number of vendors. Small vendors will not be able
to afford the large marketing efforts that are required to
sell software. The large firms will be able to support
national marketing organizations because they have several
dozen packages to sell. Smaller companies will probably be
better off selling the distribution rights of their software
to these larger firms.

-38-
As prices become more standardized and commensurate with the product, software firms will not be able to enjoy the excesses of earlier days. The firms with high prices will be squeezed out, while the firms with low prices will not be able to market their software effectively. In general, participation in the software industry will require much larger investments than at present, hurting the small firms, while favoring the large software companies and the hardware manufacturers (Frank, 1970).

In the end, a significant factor will be the attitude of the large computer manufacturers toward the software industry. The fewer vendors will probably provide more coordination in their product lines (Cowles, 1971), reducing the amount of redundant and overlapping software. The result will probably be that informal standards will be set, with more uniformity in quality, documentation and support.
Chapter 4

Survey Results

Of the 81 questionnaires that were sent out, 23 were received in time to be analyzed. This is a return of over 28% and is considered to be a good return for this type of questionnaire. One of the reasons for this encouraging return was probably because the companies were offered a summary of the results in return for filling in the questionnaire.

The questionnaires that did come back were, for the most part, very well filled out. The executives involved seemed quite willing to discuss the subject, and some of them mentioned that research like this was needed for the software industry, especially to help bring about more standardization.

The following is the analysis of the replies to the individual questions. A copy of the entire questionnaire can be found in the appendix. Instead of providing actual numbers, percentage figures are given in as many cases as possible, because many of the respondents avoided answering one or more of the questions.

1. **How many years have you been selling software?**

This was an introductory question. The age of the various
companies ranged from 3 to 13 years. However, 49% of the respondents were either 4 or 5 years old. This is significant because the unbundling announcement was made almost exactly 5 years ago -- which sparked the formation of these companies. Surprisingly enough, about 35% of the firms were formed more than 6 years ago. Perhaps this might mean that many of the successful companies have been in the industry for a long time.

2. **How many packages does your firm sell?** The answers ranged from 3 to 25, with the median number of packages being 6. There were two very clear groups in this regard. Almost half of the companies produced 3, 4 or 5 packages, while the other half produced over 10 packages.

3. **What types of packages does your firm sell?** The vast majority of the companies sold applications packages. 78% of the firms sold applications packages, 37% sold standard software, 27% sold systems software, and 18% sold "total" software systems. Surprisingly, none of the firms claimed to sell plug-to-plug replacements for manufacturer's software.

There seemed to be no correlation between the types of packages a firm sold, and either the age of the company, or the number of packages that the firm sold.

4. **What method(s) do you use to protect your software?**

This question yielded some very interesting results. All
but one of the respondents used more than one method of protecting their packages. All the respondents included "legal contracts with buyer" as one of the methods they used. Exactly 50% of the respondents demanded "written non-disclosure commitments" from the buyers, 43% classified their software as trade secrets, 43% used copyrighting for added protection, and only 37% used licensing as a means of protecting their software.

5. What is the breakdown of the costs of the package?

A wide range of responses were received for this question. However, after eliminating the extreme cases (e.g. one firm's replication costs are 80% of its price), a definite pattern of costs could be identified. The average breakdown of costs was as follows:

- Development 24%
- Marketing 28%
- Administrative Overhead 10%
- Maintenance & Installation 10%
- Replication 7%
- Profit 21%

Although the marketing costs seem to be lower than expected, there was a definite correlation between the number of packages a firm sells and the portion of the cost allocated to marketing. Those firms that carried between 3 and 5 packages allotted between 40% and 55% of their costs to marketing. For those firms carrying ten or more packages, the marketing costs fell between 10% and 20%.
6. **What criteria have you found that customers place high priority on when evaluating software?** The most popular criteria were checked by the following percentage of the respondents:

<table>
<thead>
<tr>
<th>Criteria</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td>75%</td>
</tr>
<tr>
<td>Reputation of Vendor</td>
<td>68%</td>
</tr>
<tr>
<td>Package Costs Vs. In-house Development Costs</td>
<td>68%</td>
</tr>
<tr>
<td>Amount of Support Provided</td>
<td>43%</td>
</tr>
<tr>
<td>Cost Vs. Value of Results</td>
<td>43%</td>
</tr>
<tr>
<td>Flexibility to Modify Package</td>
<td>37%</td>
</tr>
</tbody>
</table>

The other criteria, such as "vendor's delivery schedules", and "financial status of supplier" were checked very few times. The responses are indicative of how necessary users feel documentation is, and how much programmers usually hate doing their own documentation.

7. The firms were asked whether or not customers were usually adequately informed on:

   a) **What packages are available on the market?**

One of the respondents summed up the feelings of the firms by saying, "either they are well informed or not at all". Of the remainder, exactly one-half of the firms felt that customers were adequately informed about the market, because most of them received the ICP Quarterly or the Auerbach Software Reports. The other half felt that customers were not usually adequately informed.
b) What criteria should be used to evaluate software? The overwhelming majority of the suppliers felt that customers were usually not very qualified to buy software. They felt that the job of selection was often given to persons lacking any expertise on the subject, and that when faced with "apples and oranges", customers often based their decisions on price instead of price versus value.

8. The firms were asked to rank the top 3 issues that customers complained about. The following was the number of times each issue was checked:

<table>
<thead>
<tr>
<th>Issue</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td>73%</td>
</tr>
<tr>
<td>Maintenance and Updating</td>
<td>69%</td>
</tr>
<tr>
<td>Package does not Satisfy</td>
<td>49%</td>
</tr>
<tr>
<td>Changing User Needs</td>
<td></td>
</tr>
<tr>
<td>Installation</td>
<td>24%</td>
</tr>
</tbody>
</table>

The remaining issues, such as general "performance of package" "efficiency of the software" and "programmers can't get at the code", were checked very few times. An interesting fact is that in all but one of the times that documentation was ranked, it was ranked #1.

It would be interesting to compare these statistics with how users actually feel about software. In a 1973 survey of users, (Leavitt, 1973a), they were asked which aspect of software disappointed them most. 40% of the users felt that documentation was usually inadequate, 25% showed
a concern for better efficiency, and 20% called for "better performance". Maintenance was criticized by another 20% of the users.

The fact that "efficiency" and "performance" were noted very rarely by the suppliers, is probably due to the fact that few of them would be ready to admit, (even to themselves), that perhaps one of their packages performs poorly!

9. For how many years can you expect to sell a package? 70% of the respondents that provided figures said the average life of their packages was 5 years. One response was as low as 2 years, and another as high as 10. Several firms said that their packages were constantly evolving with user requirements, and hence had an indefinite life.

10. The firms were asked to rank their principal selling methods. 67% of the firms used salesmen as one of their chief selling methods, and of these, salesmen almost always ranked #1. 67% of the firms also listed "mailing package information to potential users", but this method was consistently ranked #2.

Surprisingly enough, while 60% of the users ranked "advertising in journals" within their top 3 methods, 55% of the users also listed "word-of-mouth" as a chief selling
tool. From some of the general comments that were added on, this was taken to include situations like consultants advising their clients, satisfied users writing articles in journals, seminars and courses that are offered, trade shows, and publicity in journals by software editors.

As far as "other" methods were concerned, many firms believe that listing a package in the ICP Quarterly is a very effective selling tool.

11. What are the principal difficulties that you encounter in marketing software? This was an "open" question, but not surprisingly, the same difficulties in marketing are being experienced by a large number of the firms.

The difficulty that was cited most often was overcoming the bad name that has been associated with the software industry. Companies had problems convincing customers that theirs was not just another "cheap" package, especially those that had already been hurt by a "bad" software firm.

The next biggest problem was cited as being the "not invented here" syndrome. In fact, those same three words were used by every firm that mentioned this problem of users wanting software developed by their own programmers, being wary of packages and preferring to put idle programmers to work instead of investing in outside software.
Another problem that was cited often was that many buyers simply "shop around" and try to steal ideas for their own applications. Several firms complained about the long time periods that DP departments take to make their decisions, the high cost of traveling to make presentations, and the fact that many DP departments are incapable of making or understanding the "make or buy" decision.

12. Do you foresee software marketing costs in general declining in the future because of third party salesmen or some other methods? Over 70% of the users felt that quite the opposite was true, that the costs of marketing their software was going to increase. The most common reason given was that applications software was becoming more and more complex, and hence more difficult to sell. Other reasons given were the increasing costs of travel and keeping salesmen, relative to the declining costs of computer time and hardware needs.

However, the 30% that felt that the portion of software costs devoted to marketing would decrease in the future made a strong case for their point of view, but were generally looking at the long range picture, when the industry "matures", sometime in the 1980's. The following quote from one of the firms sums up their feelings quite well:

-47-
"It's inevitable that marketing costs will decline. An industrial product must mature to a lower quotient of sales costs.

The product maturity, the buyer knowledgeability, the merchandiser capability, all will enhance distribution efficiency, as they have for all new major cost products from autos to plastics".

13. What do you feel are your company's major strengths that made it succeed in this competitive industry? Although this was also an "open" question, fully 50% of the firms replied that their major strength was their "superior" technical expertise and computer systems knowledge. Another 50% of the firms added that they had a good, or sound product that fitted their user's needs very well. Another factor that was often interspersed between other strengths, can be described in the respondents' own words such as "determination and "perseverance": marketing know-how was another factor, but mentioned much less frequently.

14. What factors or prerequisites do you feel are essential for success in this industry? There was a marked contrast between the answers to this question and the answers to the previous one. Skillful marketing was considered to be by far the greatest requirement for success. The general consensus was that although having a sound product and technical expertise helped, marketing was a key issue. Adequate financial backing was also considered to be very
important, as was the need to build a good reputation. Words such as "honesty" and "integrity" cropped up fairly often, as did words like "long hard hours", "perseverance" and "stick-to-it" attitude.

From the answers, it seems that the suppliers feel that to succeed, one initially needs a good, technically sound product, with adequate financial backing. Thereafter, one must concentrate on marketing, and building a reputation of being an honest, sound company.

15. How much do you feel the market for the following types of packages will increase by 1980? The respondents were given 5 types of packages: total software, applications packages, systems software, plug-to-plug replacements, as well as standard software. They were also given 6 choices for the increase, as shown below, including a "don't know" category to discourage wild guessing. The following shows the distribution of the votes, by percentages.

<table>
<thead>
<tr>
<th>Packages</th>
<th>Decrease</th>
<th>Don't Increase</th>
<th>0-100%</th>
<th>100-200%</th>
<th>300%</th>
<th>&gt;300%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Packages</td>
<td>12</td>
<td>5</td>
<td>34</td>
<td>19</td>
<td>5</td>
<td>25</td>
</tr>
<tr>
<td>Appl. Pkgs.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems Software</td>
<td>5</td>
<td>10</td>
<td>56</td>
<td>19</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Plug-to-Plug</td>
<td>22</td>
<td>63</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Standard Software</td>
<td>28</td>
<td>31</td>
<td>31</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

From this data we can draw some rough conclusions: Most firms believe that there is a good future for both Total
Software Systems and Applications Packages, with the growth till 1980, being somewhere between 100 and 300%, but probably between 100 and 250%. Systems Software will also show some good growth, and more than half the firms agree that it will grow by about 100%.

The firms do not seem to be interested in plug-to-plug software at all, with 63% not knowing, and a majority of the remainder feeling the market will decrease. The firms also felt that Standard Software has some good growth potential, probably between 50 and 150%.

16. If possible, please rank those industries that will show the largest increases in demand for packages between now and 1980. Only 12 of the 23 respondents answered this question, probably because the packages they produced were generally industry independent.

There was a wide range of opinion in the answers to this question. Although there was some consensus on which industries will show the greatest increase in demand, these conclusions should not be considered to be very accurate.

Manufacturing was the clear leader in this case. It was ranked either #1 or #2 over 70% of the time, and within the top 4 in 100% of the cases.

Surprisingly, government applications came in second,
This included education, with retailing being a close contender. Health care applications, transportation, and the traditional areas of banking and insurance, also consistently scored within the top 5.
Chapter 5

Market Trends and Growth Projections

One of the keys to business success in any field is undertaking a well thought-out and accurate market growth analysis. Knowing future market conditions is important, because it is generally easier to capture a share of the increase in the market than to compete for part of an existing market. Moreover, attempting to predict the size of a market provides one with valuable insights into the causes of the growth. Understanding the basic patterns and factors that influence growth, can prove to be a useful asset to software suppliers and entrepreneurs in minimizing the risk of not "picking a winner." It will also assist suppliers in developing software that will not be obsolete in the near future. A market analysis is of special value to the entrepreneur who wants to identify the best opportunities in the software industry in the next several years. This chapter will discuss the various factors that will influence the growth in the software packages industry, and identify the more lucrative opportunities for the independent supplier by presenting and justifying some future growth projections.

Factors Influencing Market Growth

The major factors that will probably be most responsible
for any growth in the market for software packages can be identified as follows:

IBM's software policies. The demand for a given type of package from independents depends very much on whether IBM (or, to a much lesser degree, other hardware vendors), intend to produce that package. IBM still gives some of its hardware away for free, and competition is almost automatically eliminated from these areas. Moreover, it is difficult to try to underprice IBM for software that it does charge a price for, because IBM sometimes prices its packages unpredictably. The hardware manufacturers also have much stronger and better organized marketing operations than independent software vendors.

One area that will always be safe for vendors to capture is when IBM announces that it will no longer support an old system. For example, as IBM gradually withdraws support from its 360 models, it opens up whole new fields for software manufacturers, since a large number of 360's will be in use for quite some time. Software companies are still thriving on the fact that IBM stopped supporting its model 1130.
Increased acceptance of packages by the user.

Users used to be very resistant to buying software from outside for a number of reasons. Because they felt it threatened their jobs, programmers and systems analysts would often suppress information about software packages, or conjure up a variety of reasons why an in-house job would be the only feasible solution to a problem. EDP managers resisted packages because of the "not invented here" syndrome that was often mentioned in the survey.

But all this is changing, and the trend in the mid-seventies will definitely be to a much greater acceptance of software packages. There will again be a shortage of programmers, making packages more attractive. Standard and system software packages, that can improve programmer productivity, will be very much in demand. By buying commercial packages, the DP manager will be able to free his valuable programming personnel to those data processing problems unique to his particular installation.

Furthermore, managements are forcing EDP departments to become much more cost-effective. However much a DP manager may want to develop a program in-house, the time and cost of documentation and maintenance will force him to consider packages more favorably.

-54-
Moreover, DP personnel often do not have the necessary technical expertise to develop some of the sophisticated packages that are needed today.

Packages are also becoming better and better as far as quality is concerned. The "horror stories" of the earlier days are becoming less frequent, and potential buyers have a greater number of satisfied users to confer with before buying any package. Thus, one can say with some confidence, that this increased interest by the user, coupled with aggressive marketing efforts by 80Q software companies, is bound to increase the demand for software packages quite substantially.

The increase in the EDP market itself. The EDP budgets of U.S. companies have been increasing steadily every year, rising to a total of $28 billion in 1974. Not only are the total budgets increasing, but the number of EDP installations is also increasing rapidly. Canadian and European EDP budgets are also increasing at a rapid pace. Although Europe has traditionally been a few years behind the U.S. in computer applications, it is narrowing the gap, and as software suppliers are waking up to the fact that IBM sells a lot of its software abroad, they will start to capitalize on the market opportunities.
in foreign countries.

**Increasing standardization of software packages** will definitely help the packages market. Programmers will find that with more standardization they can string together packages in more combinations for a variety of applications. An immediate consequence of this will be that as applications programming becomes easier, more and more automation will be introduced into industry, further increasing the demand for packages. Programmers and analysts prefer to spend their time designing software instead of coding, documenting and maintaining programs.

**Communications.** The seventies are heralding a much broader use of communications-oriented data processing systems (Frank, 1970). The great increase in the number of terminals in use, and applications requiring automated data entry, will create a new demand for a variety of software.

A more realistic attitude to selling software will also help the software package industry. The industry is now five years old, and as it matures, users are also becoming better informed about the availability of packages. Better distribution systems will emerge, as third parties and software "brokers" provide some economics of scale to marketing. Vendors are also
pricing their products more consistently, and are becoming more adept at maintenance and customer relations.

In addition to these factors, software is becoming increasingly complex to write, and DP managers are becoming aware of the vast duplication of effort that exists from installation to installation.

A Preview of the Growth Potential

In a 1973 survey of software users reported in Computerworld, (Leavitt, 1973a), an impressive 45% of all users said that they expected to increase their annual use of software packages by more than 10%. Another 48% planned to increase their use of packages by up to 10%.

A more recent survey of 1974 DP budgets, (McLaughlin, 1974), also revealed some impressive statistics. Software funds to be paid to mainframe manufacturers were up by 50% from 1973, but the increase in funds budgeted for independents was up well over 100%.

Smaller sized DP installations were increasing their budgets for software purchases from mainframe manufacturers by 25%, but they had budgeted for a 400% increase in money for independents.

Medium sized installations were intending to spend an
average of $15,000 on outside software in 1974, as compared with $6000 on software from IBM and other mainframe manufacturers.

The large DP departments were spending 85% more for acquiring software from their computer vendors, but were increasing their budgets for outside software packages by 230%. Although 100% of the users pay mainframers for software, and only 66% pay independent vendors, 75% more real dollars are going to the independents.

This trend of dramatic growth is expected to continue by almost everyone connected with the industry, from users and consultants, to software vendors and IBM salesmen. As DP managers get pressured to produce more results for the same money, the outlook for software packages is one of rapid growth from industry to industry, until the markets for particular packages is close to saturation. The market for some software, such as "total" software systems and applications packages, will probably increase indefinitely.

Growth Projections

The author has attempted to derive a set of figures for the size of the market for software packages in the year 1980, by analyzing all available data on the subject. The general method used was as follows:

The survey results in Chapter 4 can give us some idea
of how fast the market is growing. All the respondents have a stake in the future of the market, and hence their estimates are probably quite realistic.

The market research firms of Frost & Sullivan and International Data Corporation have derived some statistics on the future of the market. Although the author was permitted access to this data, he is not permitted to quote these statistics in this thesis. However, the figures have been used as inputs into the market analysis. Frost & Sullivan have predicted the market for the year 1982 to the nearest million dollars! The author is a little skeptical of some of these figures, since it is impractical, for example, to predict the market for systems software in 1982, without having any knowledge of, say, IBM's "Future System". However, in many cases, there was close correlation between its figures and those available from IDC.

IBM software salesmen, after some prodding, were willing to discuss the future of the market. As IBM controls 40% of the market, their opinions were given some weight.

The software packages market has been compared in the literature with other markets, such as the computer mainframes market in the mid-sixties. By analyzing the patterns of growth in these industries, an understanding can be gained of the probable patterns in the software industry.
We have statistics on the present distributions of users for the various types of software packages. The market for 1975 can be predicted fairly accurately, as most of these packages are either in existence now, or in the development phase. The trend from 1969 to 1975 has been analyzed, and projections have been made till 1980.

Lastly, the literature contains many predictions of the future of the market, ranging from actual figures (e.g. Frank, 1970), to mere guesses and wishful thinking. This data has also been considered, but more importantly, the reasons for this growth that have been provided in the literature have been considered in the analysis.

**Total Information Systems**

The most phenomenal growth will occur in the area of complete software packages. Although in 1968 there were almost none of these packages, the market demand has quickly climbed to $20 million in 1973. As more industries such as manufacturing installations, communications, medicine, and retailing, require automation, the number of total information packages sold in 1980 will be about $100 million -- an increase of more than 500%.

There is a large need for integrated management packages, because, although the individual elements such as
payroll, inventory, production control, accounting, and common data accessing programs do exist, the capability to combine them is relatively new (Cowles, 1971). The bulk of the increase will come from the stupendous increase in the demand for total packages for minicomputer applications. In 1973, about $3 billion was spent on professional staff, such as programmers and analysts, for minicomputer applications. Packages will be used to reduce this cost substantially, because many manufacturing applications can be customized with modular packages.

In the areas of process control and communications, (front ends, intelligent terminals, etc.), although the variety of packages will not be very great, the quantities of these packages sold will be quite substantial.

Innovation in the use of minicomputers in various fields will also influence the demand for total packages. Innovation in the automation of medical centers, libraries, and a host of other areas, will cause a number of turnkey systems (sometimes bordering on facilities management or packaged services) to be supplied.

Systems Software

The growth in this market, from about $45 million in 1973, to $90-$100 million in 1980, will be helped both by the high unit value of the individual systems packages, and also because users rarely want to commit themselves to
developing their own systems software.

Data Base Management Systems will be the leader in this field, because of the increasingly large amounts of data that need to be stored and accessed. Plug-to-plug replacements for manufacturers software will also increase as users finally wake up to the announcement of unbundling, and as mainframe manufacturers stop supporting their older models.

Many sources predict, however, that this market will be saturated by 1980, as vendors will be supplying customers with most of their systems software needs, and the rate of technology changes in hardware will be slowing down.

**Standard Software**

The market will probably increase from the 1973 level of $34 million to only about $50 million in 1980. Many of the areas that are included under Standard Software have already been saturated by the host of suppliers in this field.

The major part of the increase will be due to sub-systems software, with increasing needs in areas such as file management systems and report generators.

Communications and teleprocessing packages should double in sales because of the national data processing networks and communications needs described in Chapter 2.
Very little increase is expected in packages such as simulators. The need for performance measurement packages will also not increase, because although there is increased interest in measuring performance, it is only an occasional measurement, and users are generally finding it easier to bring in specialists from outside to do the job.

Applications Packages

Although the picture portrayed for standard software is not too bright, it is more than made up for by the spectacular increase that will occur in the demand for applications packages. Overall, the market is expected to increase from the 1973 level of $75 million, to about $200 million in 1980. Excellent growth is expected, not only in functional applications packages, but also in industry-specific packages.

As far as functional application packages are concerned, competition is likely to cause prices to stabilize considerably, because expensive packages will not be able to compete effectively, and the unsupported, inexpensive packages will lose their customers.

The best increases in functional packages will probably be seen by general business packages, including the traditional payroll, accounting, and inventory packages. There will also be good growth in marketing related packages.
such as sales analysis and test marketing packages, and also in scientific and engineering programs. However, the market for project control packages and mailing applications will only grow as fast as the EDP industry itself.

Industry-specific packages will also show good growth on an industry-by-industry basis, with the best areas being in health care, banking and manufacturing applications.

Although banks have large amounts of clerical and processing work that needs to be automated, a large number of banks have already either installed their own computers, or have their processing performed by a service company. But most suppliers feel that banks will still contribute to the increase in demand for application packages by automating more functions within the banking industry itself. The total demand for banking packages is expected to increase from the 1973 level of $22 million to almost $40 million by 1980.

Health care affords perhaps the best prospects for industry-specific packages. The increase will not come from sophisticated packages that automate complicated functions such as patient monitoring or medical diagnosis, but will stem from automating simple clerical procedures. Hospitals and medical centers have large amounts of clerical work, such as insurance claims processing, accounting, patient and bed scheduling, and processing bills and accounts
receivables. The pattern of sales to hospitals is expected to be similar to that of the banking sector in the late sixties. In general, because hospitals are becoming more cost conscious and desire more effective administrative procedures, the total sales of health care related packages should climb to almost $30 million from the $4 million level of 1973.

In manufacturing, a number of major, well supported packages will help stimulate demand. Few DP managers like to write their own inventory management or bill-of-materials-processing programs, and hence, there will be some excellent growth prospects in this area.

The demand for applications packages is also expected to grow in areas such as transportation, petroleum and chemicals, utilities, and insurance. Generally, it will be a question of supply, and not demand; whenever a good industry-specific package is developed, it will probably sell well to the market that it was intended for, if the rules for development and marketing described in Chapter 6 and 7 are followed.

In sum, the growth in the market for all software packages is as follows: ($million)

<table>
<thead>
<tr>
<th></th>
<th>1973</th>
<th>1980</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Software Packages</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>Systems Software</td>
<td>45</td>
<td>100</td>
</tr>
<tr>
<td>Standard Software</td>
<td>35</td>
<td>50</td>
</tr>
<tr>
<td>Applications Packages</td>
<td>75</td>
<td>200</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>175</strong></td>
<td><strong>450</strong></td>
</tr>
<tr>
<td><strong>-65-</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Chapter 6

Strategies for Success

From the growth projections that have been presented in Chapter 5, it becomes apparent that there are some prime opportunities for entrepreneurs in the software packages market, especially in the areas of "total" and applications packages. On the other hand, it is a fact that many software houses are now either defunct or have been forced to merge to survive. What, then, one might ask, are the dangers and risks involved? What strategies for success can be isolated to minimize the risks involved for the potential entrant? The purpose of this chapter is to provide a framework within which an attempt can be made to answer questions like these.

Although the successful development of packaged software is a risky business, fraught with dangers and uncertainties, it is still a highly technical area in its infancy -- which the author contends is a situation that is not only predictable, but controllable.

Reasons for Failure

Let us first examine why many software companies have failed. Often, there was only a limited market for the
package, and after the market was saturated the company had not had the foresight to develop other packages. These were usually the one or two man shows operating with little financial strength, who also could not afford the costs of effectively marketing software. But more often than not, the unsuccessful companies failed on one or more of the following points:

1. They grossly underestimated the costs of effectively marketing software. To sell software one needs a stronger, and more experienced marketing force than has usually been anticipated.

2. The actual development costs were considerably greater than estimated. This is also true of many in-house software development projects, where the final costs often run more than twice what was expected. But this can mean disaster for a new company that is operating on a tight budget.

3. The software products were not sufficiently generalized to capture as large a market as anticipated. Although user attitudes are changing, there is still some resistance to packages, and consequently users that do need a package will hesitate to buy one if they feel they can develop one themselves that is more suited to their specific needs.
4. Many companies were poorly financed and failed to develop some steady sources of revenue to sustain a financial base from which to develop new operations.

5. Although they did have a successful product, one of the large users or mainframe manufacturers "dumped" a package on the market at low prices so that competitors were squeezed out. The price of the package was then raised. This tactic has been used several times by a certain mainframe manufacturer from Poughkeepsie, New York, but a good example of user "dumping" took place in early 1973:

The Cuppertino, California, firm of Boole and Babbage was marketing two performance evaluation packages for about $8800 apiece. The University of Georgia's Cosmic Computer Center, under contract to NASA, had developed two very similar packages for about $300 apiece, because any revenues were simply "frosting on the cake" for the University of Georgia. It was not worth the $100,000 to Boole and Babbage that it would have cost to take NASA to court for patent infringement actions, and hence its two performance packages flopped.

Requirements for a Successful Company

From the responses on the questionnaires, the litera-
ture on the subject, and especially from the causes of failure in the past, a set of general guidelines can be derived that should be followed by anyone intending to be a contender for the substantial profits to be made in the software package industry.

1. An evaluation of the market must be made. It is important to define as accurately as possible the size of the potential market for all packages that are being considered for development. A reasonable amount of information can be derived from lists of thousands of computer users in the U.S., defining variables such as operating systems used, core size, and type of usage. These lists can be obtained from firms like Standard & Poor, Quantum Sciences and IDC, among others. Lists of European users may also prove to be helpful.

2. A viable program product must be selected. What needs of users does the program satisfy? Is it general or industry-specific? How much custom work will have to be done? The best way to determine what characteristics must be included in the package is by conducting discussions with a sample of those that have already been defined in Step 1 as being good prospects for the package. According to Hummel (Hummel, 1973), this means conducting interviews at a relatively deep technical level.
to determine file structures, performance characteristics, and to get a feel for the reactions of users who will have to decide if they want to buy and install the package.

3. An analysis should be made of competitive products. It is not very wise to try and introduce a package that will not withstand competition. The actual analysis is fairly simple, as complete listings of competitive products are available from sources such as the ICP Quarterly Computer Digest, and the Auerbach Software Products, which provide statistics such as prices, number of users, geographic concentrations, and functional features.

4. A commitment must be made for the long-term investments of time and money which are required to provide the many years of maintenance and support that users expect from a good package.

5. The package should be flexible enough so that custom work can easily be done on it. One cannot usually expect to sell a package "as is" to a large number of users. Modifications should be able to be made to meet the changing needs and expectations of users.

6. The package must be priced at considerably less than what it would cost the user to develop the package himself.
7. The package must have a useful life (i.e. the length for which it can still be sold) of at least three, and preferably, five years, for reasons to be explained in Chapter 8.

8. The exorbitant costs of marketing, installation and maintenance must be anticipated and included in the price of the package. For new companies it is sometimes preferable for it to make arrangements with a larger company or software broker to market the software.

9. A financial base must be built up, with monthly revenues coming in from one or more leased packages.

10. The vendor must protect his software adequately.

Crucial Factors

Some of the above rules may appear obvious, but almost every software firm that has failed has overlooked at least one of these rules. But there are two more major factors that are involved in planning a strategy to succeed in the software business that deserve to be discussed at greater length:

11. The vendor should carefully note and evaluate the activities of IBM in the software area.

12. The package should make sound economic sense for the user as well as the supplier.
IBM

With 40% of the market, IBM continues to completely dominate and influence the software packages industry. This was plainly evidenced when, in 1969, one announcement by IBM would breathe life into over 100 companies. IBM holds onto its power because it has publicly stated that some of its software will always be free.

For example, IBM provides for free, all system control programs, which it defines as those programs that are "fundamental to the operation and maintenance of a system" (Goetz, 1972c). Among other implications, this means that software companies cannot safely develop software to compete with packages such as IBM's TSO, OS, DOS, or CRJE.

IBM also provides for free, programs that were available prior to its unbundling announcement. Thus it is foolhardy for an independent to try and market a new Fortran or Cobol compiler, or an ISAM package. Programs that have been enhanced since then, such as ANS Cobol, are still made available by IBM at absurdly low prices.

Moreover, IBM provides its "Type 3" packages for free. These are programs that have been "donated" either by an IBM division, or by a user. Independents must also note that IBM does not separately and fairly price all its existing software. Many software houses have complained that IBM's pricing policies are illogical and unpredictable.
IBM's most frustrating tactic is to keep the prices of some of its packages very low, force competitors out of business, and then raise the price of the package to make users pay and pay (Goetz, 1972c).

However, IBM prices all its unbundled "program products" fairly and predictably. Thus the best hedge against IBM's policies for any independent software supplier is to concentrate on developing those "special applications" packages that IBM includes under its program product pricing plan. This is not as limiting as it may sound. Most "total" software packages and industry-specific packages fall into this category.

A careful assessment must also be made of the probable actions of competitive users. The case of the University of Georgia that was mentioned earlier is not an isolated incident but occurs when users find that by "dumping" their programs on the market at very low prices they can recover some of their development costs. If this "dumping" discourages competitors, and these dumped programs become popular, they will not hesitate to raise their prices. Many EDP departments are tempted to go this route, since there is little risk involved, and any revenues received are pure "frosting on the cake."

**Sound Economic Sense**

The other important factor that can assure success is
that a package should not be developed unless it is economically sound for both the user and the supplier. For the user this means that he must be convinced he will experience significant cost savings, and his investment has low risks associated with it -- i.e., if the deal turns sour he can back out of it with not too much trouble. The user must be guaranteed of good installation, maintenance and updating of his package.

For the supplier it means he must be aware of the typical breakdown of costs that are analyzed in Chapter 8. He must anticipate all the indirect costs of program modification, personnel training, installation of the package, and travel (Shively, 1971). He must estimate as accurately as possible, the number of buyers he expects to find, as 90% of his large investment must be made before he can receive a single dollar in revenues.

To raise the probability of success even higher, there are a number of rules that can be set down for the supplier to follow concerning factors that he has 100% control over. Almost all these principles have been learned by software vendors the hard way. These guidelines are concerned with developing the package so that it is a viable, high-quality product; maintenance and installation which the user typically finds extremely important; marketing the package which requires a strong and experienced marketing force; and
protecting his package, which with recent patent and trade secret rulings, is becoming increasingly complicated. These four topics are considered to be important enough to be discussed in the whole of the next chapter.
Development

It is very important that there be a sufficient amount of planning before developing a package. It is generally true that the time and money spent before development in making sure that the package will be able to fit a variety of users' requirements is well worth spending, especially when compared to the costs of making changes in later stages of the development phase.

The key to a successful development effort often lies in defining the fine line between developing a package that is so general that it is impractical to build, or one which is so restricted that it is virtually unmarketable. The best method of distinguishing which functions can easily be generalized and which will have to be tailored, is by actually discussing them with potential users. Potential users must also be asked to define the quality standards that they would like to see in the program itself, its documentation, maintenance schedules and functional efficiency. Simply guessing at what users probably want always proves to be more costly in the long run.

As a minimal guideline, the following points should be
kept in mind while planning the development of the package:

1. Programs should be made as modular as possible. Not only will this ensure that they can easily be tailored to users' needs, and be modified easily, but by using modular programming techniques, such as structured programming, it will become much easier to code, debug, and test the program during all phases of the development.

2. The input and output options should be flexible. These are the most common forms of customizing that customers want.

3. Editing features should be provided.

4. The program's operation should be demonstrably efficient, so that the customer is at least assured that some degree of technical expertise went into the package design.

5. Data validation or edit procedures should be provided where required for data control purposes (Baker, 1972).

6. The package should be dynamic so that data items can be added to records, file sizes can change, and programming logic can be changed to add on programs that perform new functions.
Testing

It cannot be assumed that product development is complete when the program has been installed in the supplier's environment. Tests must be performed on a variety of systems environments because of the many different kinds of operating systems in use today (Hummel, 1973). It is very helpful if potential users are allowed to test the system thoroughly as part of the field test, because this will bring up any installation difficulties that may exist, and it will also provide the supplier with some valuable references for future use. The field tests will also help in defining any enhancements that the program needs to satisfy a majority of the market.

Documentation

The fact that adequate documentation is vital to a package's success cannot be overemphasized in this report. Inadequate documentation is by far the biggest reason for customer dissatisfaction with software packages today. Many users expect higher standards of documentation from software suppliers than from themselves, because documentation is often the only link between supplier and user, especially if the supplier's firm folds after one or two years!

Although providing adequate documentation is expensive, the returns in the form of increased user satisfaction and enhanced company reputation, make it well worth the invest-
ment. Even comparatively weak software sells well if it is carefully and thoroughly documented, while a high performance package will not sell with weak documentation (Hummel, 1973).

Adequate documentation should include:

1. **An applications manual**, which is an overview of the scope of the system, its functions and features, and capabilities that can be achieved through future enhancements.

2. **A user's manual** that is simple and easy to read.

3. **A systems manual** that defines programming internals, with individual program descriptions and logic diagrams.

4. **An operations manual** which gives source listings, and diagnostic and error handling procedures, etc.

5. **An installation guide**, for those packages that the user must install himself, or use in some combination with other software.

It also pays to package and format all the documentation in such a way that it is easy to insert program enhancements and updates that are distributed to customers. Neatly packaged documentation, using quality paper stock, and attractive binding, is an effective sales tool, especially when one considers that it is one of the few tangible parts of the entire software package!
Marketing

A software product cannot sell itself. The software firm must be prepared to make the substantial investments that are required to develop and support a thorough and effective marketing campaign. Martin Goetz (Goetz, 1972b), an expert in the proprietary software field, feels that a reliable measure of the potential success of any package is the amount of resources allocated for effective marketing. Unfortunately, it is probably not until after the development phase, that many new firms realize that now they will be spending an average of one-half of their entire resources bringing their software to the market.

Although the process is very difficult for those that have never done it before, a definite marketing plan must be created, which defines strategies, techniques, and resources that will be used to interest all prospective customers in the cost-saving potentials of the package to be sold. It is generally recommended that the main emphasis of all sales brochures should be on how the package's functional characteristics can satisfy the user's needs, together with the necessary cost justification criteria, as well as guarantees of maintenance and periodic checkups. A list of satisfied users can prove to be very valuable, as potential users usually like to check with them before committing themselves.
The following are the most popular techniques for marketing software, and it would not be unwise to consider a marketing strategy that uses all of these methods:

1. **Advertising** in computer journals and publications. This is expensive, and should be preferred by firms that market a number of packages, as it is an effective means of building up a company's name, which can be crucial in the software business. However it is not as cost-effective a method for generating inquiries as some of the other sales techniques discussed below.

2. **Direct mail** contact of potential users is an inexpensive method that enables one to send a controlled amount of sales brochures and introductory letters to large numbers of sales prospects. Response rates to mailers will not generally exceed 3% to 5%. All inquiries should be followed up, at least with additional literature. Some companies are just collecting literature and manuals, so it is not always wise to spend the money to follow up with a salesman (Hummel, 1973).

3. Presentations at user meetings, seminars, trade displays, and business meetings are a proven method of generating a substantial amount of inquiries, (Baker, 1972), especially if they are directly followed up by salesmen.
4. "Success stories" in computer magazines, describing how and where the system has been successfully used, is probably the most effective, least costly, and the most underused method of selling software. Software editors are usually quite willing to accept articles of this type, especially if it is of value to other users.

5. Salesmen are being mentioned with some reluctance, though they are by far the most popular method of selling software, because one cannot advise anyone on how to succeed in the software business by emphasizing the use of salesmen. Salesmen are the most expensive means of selling software; they have to be trained, must have a large amount of technical knowledge, command high salaries and commissions, have high travel expenses, and can cover only a limited geographical area. Every other method should be used before relying on salesmen to create the entire link from seller to buyer. They should only be used to follow up an inquiry by a potential customer, or to close a sale, but almost never to introduce a package to a single user at a time.

6. Most customers will not look at a package twice if it is not listed in either the ICP Quarterly, or the Auerbach Software Reports or both. If the package has any merit at all it will generate inquiries simply from
being listed, especially if it receives a good evaluation from Auerbach. The advantage of a listing in the ICP Quarterly is that the supplier is allowed to write the one page description of the package himself.

7. The marketing rights to the package should be given to as many independent sales agents and software brokers as possible, especially if they only demand a commission. It must be noted that if the average percentage of marketing costs per package is about 45%, the supplier is still winning big if he has to pay a commission of 35%! However, there are some disadvantages to using brokers, which will be discussed in the next chapter.

Finally, it must be remembered that salesmen alone cannot close a sale. Particularly for large packages, that have numerous options or that need some custom work before installation, some sort of technical staff is required. In many cases this can be the same personell as the installation staff. But in any case, this technical staff is usually very necessary, as varying degrees of "customer handholding" is quite common, particularly if the package involves file, data, or program code conversion before or during installation, (Hummel, 1973).

Installation

As far as installation is concerned, it is wise to
develop a standard procedure as soon as possible, keeping in mind the various operating systems that the package will probably have to be run on. The blocks of machine time that are required should be scheduled in advance, and a series of tests should be devised that will be convincing to the user.

Adequate documentation, especially an installation manual, can greatly facilitate the installation process. At the Applied Data Research Users Conference in Princeton, N.J., December 1971, several hundred users from across the U.S. voiced their observations on the pitfalls of installing software packages (Fried, 1973). The general consensus was that the greatest difficulty that was encountered when installing software packages was the lack of adequate documentation. This statistic should speak for itself.

Customer training, to teach the user to utilize the package effectively, is an item that many new software vendors completely forget to budget for. Classes for systems programmers, operations personnel and users must be prepared in advance, and usually involves finding (headache!), and hiring one or more full or part time teachers that must become familiar with the use, maintenance, logic, and other details of the system. A large number of user personnel must usually be taught how to use the new system, ranging from keypunchers who must prepare input data, to management that must read the reports.
Maintenance and Enhancements

A large part of the success of a package depends on the technical support provided with it. The experience of several major, successful software companies verifies that extensive programming costs are likely to be incurred throughout the life of a package (Goetz, 1972), and a firm should budget for these substantial expenses. Customers are usually more impressed by concrete guarantees of product maintenance, personnel training, and installation assistance, than by exaggerated sales claims.

The best packages are those that require a minimum of maintenance, but are always improving and evolving through periodic enhancements. Users generally do not like their operations disturbed by maintenance men, but welcome enhancements. Maintenance usually involves changing or modifying the package, eliminating errors, or keeping the package compatible with the user's environment. The user may have changed to a different operating system, upgraded to a new system, or require a different set of input and output formats, or an accounting package may have to be modified to become compatible with new tax laws. However, the best packages are those that are flexible enough to allow the user to make these modifications himself, perhaps via a few control cards.
Enhancements usually provide the system with new features and capabilities. These changes should be anticipated and planned for, as they are the best means of extending the life of a package. Enhancements are also a convenient means of gaining additional revenues, as it is much easier to sell a user on an "enhancement" than on a whole new package.

In most cases, a large amount of money must be spent on hiring and training salesmen, installation staff, customer training personnel, and a maintenance crew. For small, simple packages, all these functions can sometimes be performed by the same person. But this is rarely the case.

Protection of Software Packages

Because proprietary software is very expensive to create, and so very inexpensive to reproduce, it is important that a software company adequately protect its packages. Although most users will generally adhere to the restrictions placed upon them by a marketing contract, it is important that the package also be protected from outsiders to the transaction, who are often tempted to ignore these restrictions.

It is often not necessary to have to resort to legal protective measures. Many software suppliers prefer to limit use by only releasing object code, keying program
tapes to particular CPU's, keeping source code in escrow with a third party, or placing uncharted linkages in the source code to prevent the source code from being explored. If, however, a supplier genuinely requires legal measures to supplement these technical methods, it is necessary for him to understand, despite some of the legal complexities involved, the advantages and disadvantages of the various legal methods for protecting software, so that he can use the correct combination of methods in the many different situations he will face.

**Patents**

The Justice Department has deliberately avoided making a definite decision on the patentability of software for a number of years. The situation to date can be condensed as follows:

In 1967, the Presidential Science Adviser rejected a presidential commission recommendation that software not be allowed to be patented, (Young, 1972). This was followed by a number of cases that were decided by the U.S. Court of Customs and Patent Appeals (CCPA). The CCPA has succeeded in deciding, to date, that "a general purpose computer, when supplied with a new and unobvious program, becomes a new and unobvious machine", (Young, 1972).

Software suppliers were, understandably, very wary of
Having their programs patented. Although the first software patent was granted in 1968, there was still considerable confusion over the patentability issue. However, in 1972, the case of Gottschalk Vs. Benson (read: Patent Office Vs. Bell Labs), caused a great deal of excitement.

Basically, the Patent Office had refused Gary Benson of Bell Labs a patent on a computer method for converting binary coded decimal numbers to binary numbers, but the CCPA ruled that the method was, indeed, patentable. Pressed by IBM and other mainframe manufacturers, the Patent Office appealed the case to the Supreme Court.

The Supreme Court, in its now historic decision, denied Benson a patent because it would otherwise be patenting "an idea". Although it expressly denied that its decision "precludes a patent for any program servicing a computer", the court added that if programs were to be patentable, only committees of Congress could cope with the broad problems involved.

Despite the ambiguities of the decision, the intent was clear (Milgrim, 1973): Software cannot be patented, unless the companies concerned are willing to bear the enormous costs in money and time that are involved in appealing to the legislative branch.
Trade Secrets

Before the Benson decision, firms were also hesitant to use trade secrets, because the supreme court had ruled (Lear Inc. Vs. Atkins, 1969), that if a product could be patented, it could not be protected by a trade secret, but there was still considerable ambiguity regarding whether programs could be patented or not! As a result of the Benson decision, trade secrets became popular overnight.

Trade secrets usually consist of valuable business information such as formulas, processes, or special techniques. As applied to software, it is not the program itself, but the "secret know-how" which is licensed out that is protected. Moreover, the information being protected need not be unique to one company. Although trade secrets prevent employees, or customers, or others standing in a "legal relationship" with the owner of the trade secret to wrongfully use or disclose the technical data protected by the trade secret, it does not prevent a third party from either using or stealing the product.

Trade secrets are much more convenient than patents could ever have been, because unlike patents, trade secrets are inexpensive to obtain, can be obtained immediately for international protection, the secrets never have to be disclosed to patent officials, and courts usually decide
in favor of the owner of the trade secret. However, the courts may invalidate a trade secret if the owner has not made a "reasonable effort" to protect it, and hence a software supplier that uses trade secrets should take at least the following precautions to insure that the trade secret will hold up in a court of law (Milgrim, 1973):

- **Limit access** to the program to only those who have a "need to know".
- **Label** documents and tapes with "confidential" and "proprietary information".
- **Employment agreements** should contain strict restrictions, especially on post employment actions taken by employees who have had access to the trade secrets, this being the most common form of disclosure of trade secrets.

**Leasing**

Leasing software has become a very popular method for protecting software, because it brings the user of the software into a legal relationship with the lessor, in which the lessor enjoys a number of advantages. The lessor can put a number of restrictions regarding the use of his software, such as restricting use to particular CPU's, as well as preventing the transfer and copying of the programs. This approach is similar to that used by the motion picture
industry in leasing its movies to theaters.

**Copyrighting**

There are two types of copyrighting that can be made use of for protecting proprietary software: statutory copyright and common law copyright.

*Statutory copyright* can be obtained by registering a printout of the software package with the copyright office. In most cases, simply affixing the notice on the form is sufficient; the actual registration is only necessary if legal action is brought. Statutory copyright is only applicable to "published" materials, which means that everyone can use the program, but cannot physically copy it. Thus the seller cannot control the buyer's use of it.

*Common law copyright*, on the other hand, protects programs that are not published, but are used for limited distribution. This can be an effective means of protection if it is coupled with express non-disclosure commitments (Freed, 1972).

**Written Non-Disclosure Statements**

These are what are usually called "legal contracts", and are used to eliminate the major flaw of trade secrets, in that written non-disclosure commitments provide protection against misuse by third parties. Thus, this method

-91-
should be used in conjunction with trade secrets, because it makes the buyer liable if anyone steals the program.

A program that has been copied can easily be traced to the source from which it is copied, if the developer makes some minor, but detectable, changes in the flow of logic or the code of his package for each of his customers. If a third party does come up with a copy, it can then be traced to the buyer from whom it was stolen, who can then be sued.

_Licensing_, in a way, is a specific form of a non-disclosure commitment. It brings the buyer into a legal relationship with the seller, which limits use of the program, as well as bars copying or disclosure of it. Licensing contracts are usually enforceable only if the work is leased, and not sold outright.

**IBM's Proposal**

In 1970, IBM proposed a new form of protecting software, that has not been implemented as of today, but could possibly be used by future software firms. The proposal is basically a trade secret registration scheme, whereby software would be submitted to an agency, which would protect only the program itself for a few years, while the general concepts embodied in the program would be available to the public. The program need not be original in concept or design, but if unauthorized use of the protected program
were discovered, an infringement suit could be brought against the violators.

**Stratégies**

The best legal protective measures seem to be some combination of the above methods. In many cases, protection should include common law copyright (because no publication is involved), leasing the package to limit use and restrict transfer, licensing to restrict making copies, and imposing non-disclosure statements to bar disclosure of the ideas to third parties. In other cases, trade secrets, fenced with appropriate non-disclosure commitments seem to be more appropriate (Freed, 1972).

In conclusion, the following quote from Roy Freed, one of the foremost experts on legal protection of software, is appropriate:

"In the many situations in which it is advisable, for valid marketing reasons, to depart from the strict legal requirements of the various protective measures, it is important that the marketing reasons truly be sound. All too often, marketing personnel seek to degrade those measures in order to make it easy for them to secure customers. Where compromises are firmly based, their existence should be acknowledged within the supplier's organization explicitly to avoid later recriminations when deficiencies are taken advantage of by customers or others successfully".
Chapter 8  
The Economics of Software Packaging

We have seen in chapter six that one of the most important factors for success lies in the need for the software package to be economically sound for both the supplier as well as the user. Although the most effective marketing strategies are usually built around the economic benefits that can be proved will accrue to the user, the unfortunate experiences of many software firms has emphasized that they have failed to make their ventures financially sound for themselves. This chapter will identify some of the major financial problems that are unique to the software industry, and offer some guidelines for solving them. The chapter discusses pricing strategies, pricing, cost analysis, cash flows that must be understood, and the financial risks associated with the software packages business.

Pricing Strategies

There are a wide variety of pricing strategies in use today, each having its own unique advantages and disadvantages. The pricing plan that is ultimately chosen for selling software is as important to the user as is the actual price of the package, and is an important factor for the success of any marketing plan.

A smart software vendor would use more than one pricing plan, so that he is flexible enough to meet the needs of
individual buyers. In general, leasing and renting, and other methods that involve monthly payments, should be preferred by the supplier, so that there is at least a minimum of continuous cash flows with which the company can build a financial base. Almost all pricing strategies in use today will fall into some combination of the following general categories:

1. A **free trial period** is offered, after which the user must decide whether he will buy the package or not. This strategy minimizes the risk for the user, and hence is an effective sales tool.

2. A **three month "minimum lease"** trial period for which the user pays a flat fee, usually covering installation costs. This has been IBM's favorite strategy (Goetz, 1972b).

3. Payments are made on a monthly basis, cancellable at any time. This protects the user, is risky for the supplier, and is only used when the supplier cannot get the buyer to settle on the terms in item 4 below.

4. **Long term leases**, with contracts ranging from one to three years. This arrangement is used to protect the supplier, but the user usually will not agree to it unless used in combination with either items 1 or 2 above.

5. **Permanent licenses**, under which a buyer can use a package indefinitely for a flat fee. This method carries
with it the risk that the vendor will abandon the user by not supporting or maintaining the package after installation. The user can hedge against this by insisting on a legal contract. One of the disadvantages of using this method is that the IRS may rule that the software was sold, resulting in the user losing some of the tax advantages of licensing.

6. *Metered usage* involves using an internal meter to charge the user according to the amount of actual usage of the program.

7. *Outright purchase* is usually not favored by either the buyer or the supplier. The vendor cannot adequately protect his package, and the buyer is not assured of adequate maintenance.

8. Payments for the package are made based on "savings to the user" (Goetz, 1972b). This system makes payments difficult to calculate and enforce, and hence have not gained much in popularity. It does have an advantage, in that it is a close approximation to "value pricing".

It is clear that there exist a wide range of pricing methods from which the supplier can choose. Especially in today's economy, costs and pricing plans are critical to a software buyer when there exist several competitive packages from which he must choose. An alert software manufacturer would be well advised to be constantly reevaluating his
pricing policy in the light of varying economic and market
trends.

Pricing the Package

After having invested a large amount of time and money
in developing a package, all software vendors have to face
the difficult problem of determining an optimum price for
their products.

This is no easy matter. The posted price must be high
enough to recover development costs and to generate a
satisfactory level of profit, but it must also be low enough
to be competitive. Pricing is often the key to the ultimate
success of the package.

The lack of knowledge on the part of the user makes
pricing an even more difficult problem. For example, one
of the respondents in the survey wrote that the market place
was so irrational, that during a test period it priced a
package in a range from a low of $3,000 to a high of
$30,000, with no variance in the level of sales! The
president of the firm that publishes the ICP Quarterly,
(Welke, 1972), feels that some buyers are so naive that
salesmen use the "flinch" method of pricing: If the user
does not 'flinch at the quoted price of, say, $10,000, the
salesman merely adds "thats for Phase 1"! However, in the
long run, it can harm a firm's reputation to charge some
customers a higher price than others for the same software.
Although the literature in microeconomic theory is replete with methods for price determination, and in our case we are dealing with 800 suppliers, (which is an economist's dream of perfect competition), we have a situation in which no two products are ever exactly alike! As any software salesmen will readily tell a prospective customer, one is really comparing apples and oranges when it comes to comparing software and their prices. Thus, until perhaps the early 1980's, when packages will conform to more standards, and there will be a smaller number of large suppliers, microeconomic theory can do little to help vendors price their products more adequately.

But over the past 5 years successful companies have priced hundreds of packages quite soundly. Pricing software need not be the hit-or-miss situation that many vendors make it out to be. A rational approach to pricing can be developed by considering the following factors that will invariably affect the pricing decision:

1. **Value to user.** Ideally, all software packages should be value priced, which is the price a user would be willing to pay for software as an alternative to in-house development. Value pricing can also be based on savings or benefits that will accrue to the user. If a vendor has a unique product, it should be value priced by all means: The lead time required to develop a package is usually long enough that a firm need not
worry about competition entering his field before the
"half-life" of the package has run out.

2. The degree of competition. The greater the degree of competition, the lesser is the probability of the optimum value pricing level ever being achieved. Competitive pricing will generally drive prices down, and new entrants often stand little chance of competing with older firms that have already recovered their costs.

3. The expected life of the package. All packages reach a stage where the marginal cost of selling one more package exceeds the revenues that will be received from that package. At this point the market is saturated. Development costs can be spread over a greater number of buyers with a package that sells well and has a long life. Moreover, there is a form of "learning experience" that comes from repeatedly maintaining, installing and marketing a given package (Shively, 1971).

4. The type of package. Industry-specific packages, and those packages that perform special functions can usually command higher prices. Moreover, the amount of custom work that is involved will also affect the price. If the package requires a large amount of custom work, the base price can be quite low, as substantial profits can usually be made on custom work, especially if the "custom work" has already been done for another user!
5. **The financial strength of the developer.** IBM can afford to post unpredictable and illogical prices because it has more than a billion dollars in cash reserves! Small companies selling their first package must usually attempt to post the highest prices that they can get away with.

6. **Costs.** The minimum price that can be charged can be determined by calculating costs, dividing by the number of potential users, and adding on a margin of profit. The first step requires developing a sales and revenue forecast. Only high-probability prospects should be included in the estimate, and revenue flows should be calculated at various prices.

   The next step involves defining all fixed and recurring costs. A thorough analysis must be made of fixed costs such as program development, testing, documentation, training of sales and maintenance personnel, and legal help for contract preparation. Recurring costs include the cost of sales, salesmen, and administrative overhead. The fact that this estimate will probably be too low cannot be overemphasized. For example, total documentation costs for the average package can range from $50,000 to $100,000. The cost of a 100 page users manual alone, containing diagrams and illustrations, can easily run as high as $10,000, assuming that the documentation from which to prepare...
the user's manual already exists! (Hummel, 1973).

**Breakdown of Costs**

In estimating costs, the breakdown of costs has often been grossly misjudged by new companies. Marketing and maintenance costs have typically been underestimated the most.

As far as marketing is concerned, it must be realized that except for listing a package with the ICP Quarterly, there is no established distribution channel for individual packages. This means that the entire link from seller to buyer must be created afresh by each vendor, an extremely costly process. Unless a package can be sold through the mails, a software company must expect to place salesmen on both coasts, and one in Europe if he expects to sell abroad. Salesmen commissions range from 5% to 35% of the price of the package. Some companies have as many as 9 out every 10 persons in the company on the marketing force! (Welke, 1973).

Regarding maintenance, a manufacturer should expect to invest at least an additional 200% of initial development costs during the program's life on maintenance and enhancements (Goetz 1972b). The reasons for these additional costs are usually that

- enhancements to the product may be required to keep the product competitive.
- Changes may be required to expand or modify the package, because of shortcomings that are found out after the first few sales.

- User requirements and/or expectations may have changed.

The costs of the individual components of the final product, that a vendor should expect to budget for, are as follows: (These statistics have been compiled from the survey results in chapter 5, as well as from sources in the literature that have discussed individual costs - e.g. Baker, 1972; and Frost and Sullivan, 1973.)

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amortization of Development Costs</td>
<td>10%</td>
</tr>
<tr>
<td>Marketing</td>
<td>45%</td>
</tr>
<tr>
<td>Maintenance &amp; Installation</td>
<td>15%</td>
</tr>
<tr>
<td>General &amp; Administrative</td>
<td>10%</td>
</tr>
<tr>
<td>Profit</td>
<td>20%</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>100%</strong></td>
</tr>
</tbody>
</table>

From these figures we can derive a formula that can help us in our pricing problem. If, for example, our package is expected to cost $100,000 to develop and document, and we have 20 high-probability users, then each buyer must absorb one-twentieth of the development costs. This figure, $5,000, becomes 10%-15% of our price (development cost), and hence our price range should be between $35,000 and $50,000. If, on the other hand, 100 buyers are expected to be found, the price can drop considerably, to $8,000 - $10,000.

If the number of buyers is expected to vary with the
price of the product, our problem becomes similar to that used in microeconomic theory, where the appropriate profit maximization price can be found by the intersection of the supply and demand curves.

Although the marketing costs per package may vary from firm to firm (large firms usually have lower marketing costs per package), the amortization of development costs percentage almost always ranges between 10 and 15%. This is explained by the fact that larger firms, with lower marketing costs, tend to spend more money on developing new packages, and must also compensate for any packages that do not "make it".

The price determination example given above illustrates the importance of estimating development costs and potential buyers as accurately as possible before posting a price. If accurate estimates cannot be made, most suppliers would agree that a not-so-absurd hedge is to multiply estimated development costs by two, and divide the expected number of users by two!

How many buyers can a supplier normally expect? A 1972 survey by the market research firm of Frost & Sullivan produced the following data on the sales of the best selling packages of software companies:
<table>
<thead>
<tr>
<th>$ of sales of best selling package.</th>
<th>Percentage of total # of firms.</th>
</tr>
</thead>
<tbody>
<tr>
<td>more than 100</td>
<td>12%</td>
</tr>
<tr>
<td>50 - 100</td>
<td>9%</td>
</tr>
<tr>
<td>25 - 50</td>
<td>13%</td>
</tr>
<tr>
<td>10 - 25</td>
<td>29%</td>
</tr>
<tr>
<td>less than 10</td>
<td>37%</td>
</tr>
</tbody>
</table>

Thus, 66% or two-thirds of the software companies sold less than 25 of their best selling package! This rather surprising statistic can explain why more packages have failed because of prices that were set too low, rather than because their prices were too high!

A Note on Taxation

It is usually advantageous for a supplier to lease its software, not only to protect himself from the user reselling it, but because of the tax advantages to the user. Moreover, for those firms that provide "total" software systems, where there is some hardware involved, it is advantageous to "bundle" the hardware and software into one package.

The reason for this is because under IRS procedures for software costs, the treatment of unbundled software as an intangible asset results in less of a tax break for the user than if the software is obtained under license, or is included in the price of the hardware. If obtained with the hardware, it is tangible, and as such is eligible for tax advantages as investment credit, and advantageous depreciation techniques. This can explain why, before the unbundling
announcement, buyers often resisted buying superior programs from independents when they received programs with their hardware. Although licensed programs do provide a tax break, care must be taken that the total amount paid over the period of license does not equal or exceed the purchase price, because the IRS may rule that a sale has been made (Nelson, 1970).

Cash Flows

A software company should be aware of the fact that steady cash flows in the form of monthly revenues are necessary for it to be able to continue operating, and also investing in the development of new packages. Many firms are now wisely insisting on a minimum 12 month contract once a sale has been made, usually in the form of a leasing arrangement.

It must also be remembered that a software package usually takes from two to three years to develop, and to learn how to market and maintain it successfully (Welke, 1973). It is only after this period that it can start breaking even, and thus the package must have a longer useful life to make any profit. A package with a short estimated life should rarely be used to start a business with, although it can be considered as an investment by an older, more established firm.

The firm's largest financial problems occur during the first two years of operation, while the package is being developed, and a marketing plan is being implemented, and no revenues are coming in! The fact that one cannot tell whether a package is going to be a success until after 90% of the
investment has been made, and the product is put on the market, is one of the major reasons why investors consider the software packages market as risky. The most common method that has traditionally been used to hedge against these high costs is to have someone else pay for the development work. This fact can explain why so many successful software suppliers developed their packages either as consultants involved with custom work, or as in-house programmers, or as part of a government funded project.

The strain of this cash-flow burden in the first two years has been responsible for some firms folding before they ever made their first sale. Moreover, many firms complain that it is difficult to pick a successful package, and hence the losses incurred by the losers must be covered by the successful packages, warranting the high (sometimes more than 50%) profit margins on most packages. Thus, there are some risks involved in picking a package that does not sell, or one that soon becomes obsolete, or one that unexpectedly receives competition from a major computer manufacturer. But as one of the respondents in the survey wrote, perserverance is the name of the game, and it must always be kept in mind that when a packages' revenues exceed its fixed costs, very substantial profits can be realized.
Chapter 9

In-House Software and Software Brokers

In-House Software

Many firms can add to their profitability by selling valuable computer programs that have been developed by their EDP departments. Although many firms have tried to recover their development costs using this method, they have often overlooked the fact that marketing in-house software requires considerable preparation, organization, and expense.

With the proper planning, in-house software does not have to be "dumped" on the market at low prices, but can be sold at regular prices with little difference in the number of sales. Although it could follow the general strategies that have already been outlined, it must be realized that an EDP department's main business is not selling software, and it will usually be only an incidental seller. However, the following guidelines should be of help to the EDP manager:

A company that desires to sell its in-house software cannot usually succeed in doing so if it decides to market it after it has been developed. A decision to attempt to
recover some of the development costs must be made before the initial design stage, because if the software design has been geared to the specific needs of one company, there will be great difficulty involved in making the program flexible enough to meet the requirements of another firm.

It is recommended that some form of market study be made before the extra dollars are spent on developing in-house software to be general and flexible for other organizations. The market study serves more than one purpose, because by identifying competitive software packages one may then realize that an in-house development effort is not required, after all, since a competitive package can supply most of one's needs. If no competitive package can be found, then there is always a good likelihood that some other firm has the same problem. These firms should be identified and contacted. (Now there is also a possibility that another firm has developed the same software, but has not thought of marketing it. An offer should be made for the software, as it can often be bought quite cheaply.) If no other firm has developed adequate software, other firms should be approached as potential customers, and the design requirements worked out with them. If this procedure is followed by an EDP department before all of its in-house development efforts, it will probably find that
90% of its software requirements can either be bought from outside, or developed in-house and marketed.

If a competitive package exists, but does not fit some particular requirements, then its functional characteristics should still be carefully studied, together with its penetration of the market, pricing strategy, customer training and installation support. The decision to market in-house software should now be followed by a well defined development and marketing strategy, similar to that which has been outlined for software firms. The only difference is that in many cases, a separate maintenance, installation, and customer training staff may not be required, as in-house personnel are usually quite capable of doing this job. However, an EDP department may not want to make the large commitment it must make to bring the software to the market and to the user. In this case, the use of software brokers should be seriously considered.

Software Brokers

A new industry of data processing brokers is emerging today, whose main business is the selling of computer packages that have been developed either by in-house EDP departments or software package firms. These brokers act as independent agents and their marketing areas usually cover the entire country, including all the regional and
and national trade shows. Firms such as Computeria, Comsec, and Delta Computer Systems are examples of these software brokers. In some cases, software vendors with large marketing organizations will also consider marketing software for other firms (eg. Cullinane Corp., Phi Computer Services).

Interviews with two of these software brokers in the Boston area has provided the author with some important insights into the present and future implications of the emerging brokerage profession. The following points should be of interest to both buyers and sellers of software packages:

Some precautions must be taken before turning over a package to a software broker to sell. The qualifications of the broker must be assessed, including the products that the broker sells, and where he sells them. This is important because brokers always receive a commission on sales, and will always push a package with a high commission much harder than others that he may carry. Thus a small, inexpensive package will be virtually ignored by the software broker. The broker will also be tempted to ignore a package with a long sales cycle, since time is usually a broker's most critical asset. Thus a broker will always favor packages that provide the highest returns in the shortest possible time.

If an in-house EDP department wants to use a broker,
it must first install the package itself at least once or twice before turning it over to a broker. Brokers do not have the time and staff to review all the packages that are offered to them for sale. Moreover, by making the initial sales, the developer demonstrates that he has a package that is transferable and saleable. In addition, brokers prefer to be able to refer potential customers to a satisfied user.

A common complaint that has been made about software brokers is that it is difficult for a broker to thoroughly get to know a package that has been developed elsewhere, and hence cannot get deeply enough involved with the package's details to discuss them with a buyer. Although this is generally true, the complaint is not a valid one. Brokers should only be used as salesmen; only to create the link from buyer to seller. All the program modifications, system changes, and installation problems must be discussed with the user by the developer. There are some brokers that help EDP departments sell their software by suggesting modifications to make it more marketable. But in any case, the broker should still be used only to sell; all the maintenance, installation, and enhancements should never be left up to the broker.

An important insight into the software brokerage profession is that ideally all software package firms should either become software brokers themselves, or else use
software brokers to market all their software. This will
benefit both the software vendors and buyers, by resulting
in increased software distribution efficiency, reduced
marketing costs, and lower package prices. The reasoning
is as follows:

The typical small software vendor spends from 40 to
50% of the revenues from a package on marketing. A firm
that is large enough to have a national marketing force
usually budgets about 20% of the price of each package for
marketing expenses, because it has achieved sufficient
economies of scale by carrying several packages. At this
point the firm should become a software broker, because
it can sell the smaller firm's software for about 20%
cheaper. Thus by charging a 20% commission, not only is
the broker making a profit, but the software vendor is
receiving the benefits of a national marketing force at
lower costs.

There are two ways by which a broker may charge for
its services. First, the broker can charge the average 20 to
25% commission on sales, and sell the package at the
vendor's normal price. In due course of time the package's
price will drop down to a level which reflects the true
marketing costs. Second, the broker can buy the packages
at the standard price less marketing costs, (about 60% of
the standard price). Because of his economies of scale in
marketing several packages simultaneously, he can easily
underprice the software vendor in a manner not unlike
that of ITEL underpricing IBM in the hardware field (but for
different reasons). The important point here is that
in both cases the price of the package ultimately drops
because of more efficient distribution by using a broker
- benefiting both the users and suppliers.

On the other hand, a software vendor that prefers to
market its own software should make every effort to
become a software broker for other firms, so that he can
achieve greater economies of scale in his marketing strategy
by carrying more packages. From the experience of the
giant software firms like Cordura and ADP, it appears that
the portion of costs allocated to marketing can be as low
as 10 - 15% of the price of the package.

Thus, although the software broker has not yet emerged
as a dominant force in the market place, he must ultimately
do so before the industry's general marketing costs can be
reduced. There are still some problems with software brokers
that must be overcome first, such as the fact that he gives
preference to higher priced packages, and that he cannot get
deeply involved with the details of a package. But the
constantly traveling software salesman also leaves a lot
to be desired, both economically and in the view of the
buyer, and so it is inevitable that the software broker or
third party salesman will play a major role in helping the industry mature in its overall marketing concept in the not too distant future.
Chapter 10
A Note on the Packaged Services Industry

After reading the preceding sections, one is probably left with the impression that there are some excellent opportunities for potential entrants into the software packages market, with plenty of room for entrepreneurs during the boom in demand over the next several years. However, the field of data processing offers an even more lucrative field to explore, with less stringent requirements for success, as well as a unique opportunity for rapid growth and good profit potentials. This field is the packaged services industry.

What is a Service?

A packaged service does not consist of a product, but is a capability. It can consist of renting out time on computers, offering time-sharing capabilities, performing billing and accounting functions for small companies or groups of professionals, or offering the use of shared systems. Selling a packaged service often involves automating an operation that more often than not was previously performed manually. The initial service bureaus evolved from the first time-sharing facilities, when firms started charging additional fees for the use of software packages that had been implemented on the system.

Industry Characteristics

There are some important characteristics of the pack-
aged services industry that make it different from the packaged software industry. The most important difference is that IBM is not a factor that has to be contended with and watched carefully in the services market. Because of its Consent Decree with the Justice Department, IBM sold its entire services division to Control Data Corp. AT&T has also been excluded from the market when it also agreed to a Consent Decree.

Although the largest mainframe manufacturer and the largest common carrier have been eliminated from the field, the market is still quite competitive. The packaged service revenues of almost all of the largest suppliers of packaged software is usually much larger than the sales of their software products. Most of the middle sized packaged software firms obtain a major part of their revenues from packaged services. Mainframe manufacturers and common carriers (except for IBM and AT&T) have shown an increasing interest in the potentials of the services market. Service companies have generally been more successful than software houses, and hence Wall Street has recently been showing an increased interest in a number of time sharing and computer services related companies.

There is good reason for excitement in this area. The packaged services industry is one of the fastest growing industry in the U.S. today. In 1973 about $650 million was spent on packaged services in the U.S., and this figure has
been estimated to double to about $1.3 billion by 1980. (All market estimates in this chapter have been derived from the same sources as those in chapter 4. There was, however, no input from a mailed questionnaire as there was for the estimates of the software packages market). A major part of the increase in market size will come from innovative applications of computer services. Small firms and businesses that do have data processing needs, but cannot afford to have an in-house EDP department, will also be a major target for service organizations.

Reasons for Growth

The crucial reasons that explain why the packaged services industry is so lucrative are:

It is easier to value price a service than a software package. Businesses are more willing to pay for a service that improves their efficiency and lower costs - there are no in-house programmers or DP managers that will provide any resistance. Thus with several users sharing the same system, a service bureau can realize up to a 40% rate of return on its investment. Only in some markets, such as banking, does the competition depress prices far below the value pricing level.

It has generally been accepted by the business community that almost every business can benefit by the use of a com-
puter (Naylor, 1973). However, most small businesses cannot afford to spend the time or the money for a well-equipped EDP department. Alternatives to the in-house business computer will be provided by the on-line accounting, inventory analysis, billing, purchasing and bookkeeping systems that service companies can provide via one or more terminals. It is often possible to "customize" a system for a customer by putting together a combination of the packages that are available to the user.

Selling a service is much easier than selling a software package. Often the presence of a slick, new terminal will get a customer interested in the capabilities of a system, and then the salesman can display some of the very tangible benefits that can be realized by simply hooking up the terminal to a phone line.

The new communications networks that are coming up all over the country, together with the cheaper and more specialized remote terminals that are being produced, will provide an additional spurt to packaged services, especially to those that can provide nationwide capabilities.

Moreover, exercising the imagination opens up a host of possibilities for applications of computer services. Almost every industry is a potential customer for the service industry - from supermarkets and retailers to high schools and even travel agencies!
The Market Growth

Of the $650 million spent on computer services in 1973, the largest portion, about $300 million, was spent by banks and credit bureaus. Many of the suppliers of this service were the EDP departments of major banks that were spun off to become separate profit centers. Although the banking services market will soon be near saturation, because enough banks will soon be able to afford their own equipment, there are a number of other industries that will show a great demand for DP services:

The next largest sector of current DP services are the traditional business services that companies such as Keydata provide. These systems require only a small investment in equipment by the user, and offer the business community flexible services that reduce clerical personnel and inventory levels, and provide management with better reporting systems. In 1973, the total market for these types of systems was $280 million, and is expected to increase to almost $500 million by 1980.

There are also other major areas of growth, the most important of which is the health care industry. In 1973, health care related data processing in services and facilities management totaled $45 million, and this figure should rise to well over $150 million by 1980. There are a number of reasons for this phenomenal increase:
Hospitals and medical centers such as HMOs are increasingly becoming "for profit" organizations, with increased awareness of costs and inefficiencies involved with the large amount of billing, accounting, documentation and identification functions that must be performed. McDonnell Douglass Automation has already tied in over 40 hospitals into its data processing center. The automation that service bureaus provide has been found to be superior to that of in-house facilities. Service companies usually start by automating one part of the hospital at a time, proving first that each part works. In contrast, an attempt to develop a total in-house system in California was a complete failure. Thus, because of the increase in the size of the industry itself, aggressive nationwide marketing, and the good service that is now being provided, it seems that medical computer service bureaus have a bright future.

Another major area of growth are distribution systems which should almost double in sales from the 1973 level of $40 million. The transportation industries' increase in demand for DP services will mainly be provided by competitors within the industry itself. On the other hand, brokerage houses, which traditionally have had large amounts of clerical work to be automated, will increase their demand for services such as those Electronic Data Systems provides relatively slowly, because more and more of them are building their own in-house systems.

But the most spectacular increases should come from
new applications of services, in areas that have not as yet been automated. One firms' estimate of this increase is as high as $100 million. Although the research for this thesis has uncovered some of these potential areas, the reader is invited to speculate for himself on where these areas might be.

Facilities Management

Although facilities management has been becoming very popular with firms in recent years, it has not been considered to be a service in the statistics given above. Firms that offer facilities management services are concerned with providing large computer users with a more cost effective alternative to in-house operation. The demand for facilities management services at the present time is around $400 million.

Strategies for Success

The major advantage in entering this industry is that there is no competition from IBM involved. There are still some risks involved in entering the packaged services industry, but many of these can be minimized with the proper planning. Instead of failing for reasons such as poor product concept, inadequate pricing strategies, or high marketing costs as in the software packages industry, the main reason for failure seems to be insufficient financial backing to maintain the costs of keeping the computer during the early set-up and break even stages. However,
a substantial part of this cost can be avoided by doing all one's development work on rented computer time.

Most businesses that hire an EDP service company are only interested in getting a job done more quickly, more accurately, and for less money (Hammerton, 1969). Thus, success in the packaged services industry will ultimately rest on not bothering management with DP technicalities, and providing uninterruptable services with excellence in capabilities and special features, coupled with a well-honed advertising and marketing effort.
Chapter 11

Conclusion

We have seen that there are many lucrative opportunities in the software field in the next several years for software vendors, especially in the areas of specialized applications packages. For success, the firm must emphasize variety in applications, flexible designs, and high standards of documentation, maintenance, installation and customer support. But marketing is the key to success, and the firm that has a cost-effective, yet aggressive marketing plan, is bound to succeed.

For the industry as a whole to succeed, more quality controls and standardization should be introduced into the EDP industry itself. Mainframers should work somewhat closer together so that it becomes easier to standardize the production of software. For these controls and standards to be effective, hardware and software manufacturers must work together on some form of regulatory board to set and implement these controls. IBM is an obvious choice for taking the initiative in advancing this kind of system.

The software industry as well as users would benefit from more efficient distribution and marketing of proprietary program products. Although the ICP Quarterly was a major step in the right direction, perhaps more emphasis should be placed on the use of software brokers, whose activities
could also be supervised by the regulatory board. Such advances would contribute not only to increased use of packages, and the development of more high quality software, but also to the stability of the entire data processing industry itself.

As far as further research is concerned, a questionnaire that is sent to all 800 software manufacturers would be of great value to the industry, because few statistics of data are available in this field. A major study could also be performed on the kinds of controls and standards that would be best suited for both the software industry as well as the entire EDP industry.
Bibliography

1. Auerbach Software Notebook, Auerbach, Inc.

2. Auerbach Software Reports, Auerbach, Inc.


-125-
31. ICP Quarterly, International Computer Programs, Inc.

-126-
33. Leavitt, Don, "Choice Runs from "as is" to Near Custom", Computerworld, February 28, 1973b, p. 28.

34. Leavitt, Don, "Cost Consciousness Leads to Package Use", Computerworld, February 28, 1973c, p. 32.


-127-


Other References


APPENDIX

The following pages are a copy of the questionnaire that was sent to either the presidents or the chief marketing officers of 81 software package vendors:
1. For how many years has your company been selling software? ______

2. How many different packages do you sell? ____________________________

3. Please check the types of packages you sell.
   ______ "Total" Information Systems.
   ______ "Systems Software" such as operating systems, data base systems
   ______ Applications Packages.
   ______ Plug-to-plug replacements for manufacturers' software.
   ______ "Standard Software" such as I/O modules, compilers, assemblers and programming tools.

4. What method(s) do you use to protect your software? Why?
   ______ Copyrighting
   ______ Leasing Arrangements
   ______ Legal contract with buyers
   ______ Other (specify).

   _________________________________________________________________
   _________________________________________________________________
   _________________________________________________________________
   _________________________________________________________________

5. What percentage of the price of your software package is determined by costs of
   ______ Development
   ______ Marketing
   ______ General & Admin. overhead.
   ______ Replication of package
   ______ Maintenance & Installation
   ______ Profit before taxes

   -130-
6. What criteria have you found that customers place high priorities on when evaluating software?

- ____________ Documentation
- ____________ Flexibility to modify package
- ____________ Reputation of Vendor
- ____________ Financial Status of Vendor Firm.
- ____________ Amount of support and upgrading provided
- ____________ Costs vs. Value of results
- ____________ Package cost vs In-House development costs.
- ____________ Vendor's delivery schedules
- ____________ Other (specify below, as this list is by no means exhaustive)

7. Can you comment on whether or not customers are usually adequately informed on

a) What packages are available on the market. ____________

b) What criteria should be used to evaluate software. ____________

8. Please rank the top 3 issues that customers complain most about.

- ____________ Efficiency of the software
- ____________ General performance of package
- ____________ Installation
- ____________ Maintenance & Upgrading
- ____________ Documentation
- ____________ Package does not satisfy changing user needs
- ____________ Programmers can't get at the code.
- ____________ Other (specify) ____________

(Comments) ___________________________________________________________________________
9. For how many years can you expect to sell a package - i.e. what is the length of an average package's life?

__________________________________________________________________________

10. Please rank your principal selling method(s).

   ___ Advertisements in journals etc.       ___ Mailing package information to potential users.
   ___ Salesmen                              ___ Through third parties
   ___ Word-of-mouth                         ___ Other (specify) ______

__________________________________________________________________________

11. What are the principal difficulties that you encounter in marketing software?

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

12. Do you foresee software marketing costs in general declining in the future because of third party salesmen or some other methods? If so, why?

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________

__________________________________________________________________________
13. What do you feel are your company's major strengths that made it succeed in this competitive industry?


14. What factors or prerequisites do you feel are essential for success in this industry?


15. How much do you feel the market for the following types of packages will increase by 1980 as compared to 1973?

<table>
<thead>
<tr>
<th></th>
<th>Decrease</th>
<th>Don't Know</th>
<th>Increase 0 - 100%</th>
<th>Increase 100 to 200%</th>
<th>Increase 200 to 300%</th>
<th>Increase Over 300%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Info. Systems</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Applications Pkgs.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Systems Software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Plug to plug repl.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Standard Software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Comments:
16. If possible, can you rank those industries that will show the largest increases in demand for packages between now and 1980?

- Banking
- Insurance
- Manufacturing & Inv.
- Brokerage
- Printing
- Real Estate
- Chemical Industry
- Government
- Health
- Transportation
- Retailing (Billing)
- Travel
- Other

Comments

General Comments

Thank you very, very much for your time and patience, and I hope you will benefit from the report I shall send you. You are guaranteed complete anonymity throughout my thesis.

Please write down below the name and address to which you would like the report to be mailed. Moreover, any company information (such as a brochure on your products) will be of great use to me, and I would appreciate it if you could have it sent to me.