MEMORANDUM

To: Distribution

From: [redacted]

Subject: The Software Product Business (ADAPSO Seminar)

Date: October 21, 1981

On September 30th I attended an ADAPSO Seminar on the Software Products Business in Chicago. Larry Welke, the President of International Computer Programs (ICP), conducted the seminar which was attended by about 25 executives from different software companies across the country. The program discussed how to enter the software product business and be effective in marketing. I will summarize some of the notes I took.

ICP

Welke first gave a pitch for his company. ICP publishes several directories and six different magazines on software products. The directories list software products from about 2400 companies around the world. The magazines are each directed to a different industry segment.

Strategy in the Software Products Business

Software companies sell products which include varying amounts of the following components:

- hardware
- software
- people
- communications
- data bases

The objective should be to maximize the value-added by varying the amount of each component contained in the product (value-added is the difference between the price and the cost of the purchased components). For instance, a company that purchased DEC hardware and then made its own cables and sold a computer with cables would be a very low value-added business. On the other hand, a firm purchasing a microcomputer but adding $50,000 worth of software and having a group of people provide installation and training would be an example of a high value-added business.

Very few firms specialize in just one of these components. More typically, a company picks an industry ("vertical segment") and then provides a variety of software products, time-sharing services, consulting, and other products to that industry.

As in any business, a software products company strives to erect an economic barrier to make it difficult for competitors to enter its business. This can be done by creating a bundled product which is sold within a narrow market segment. An example of this would be a turnkey computer system sold to automotive parts distributors. In such situations, the sale is usually tougher (substantial experience in the vertical market is needed), the support costs are high, but the cost to duplicate the software is substantial (a barrier). Of course, such markets are generally smaller, but if the objective is to obtain a high return- on-investment, erecting a barrier within a narrowly-defined market segment is often a successful strategy.

Welke suggests that companies start with a commodity product which appeals to a large market, in order to build up a sales and support staff. Once this has been done, he then suggests the company move to a more complex, focused product, to obtain higher profitability.

Why Enter The Software Products Business?

The "entry fee" for this business is relatively low. Welke cited several examples of people starting software companies with only $5,000 or $10,000 in capitalization. These companies often grew to over $5,000,000 in sales. However, Welke said the entry fee is going up because it is tougher to "bootstrap", and the level of competition is increasing steadily.

Because the industry is so new, there are many opportunities to build companies to create the marketing infrastructure. For instance, there is a good opportunity to build a company to market other peoples' products. There is a lot of software available, but few channels of distribution. It appears to me that companies that can obtain or create distribution channels will be the most successful companies in this business over the next five years (until other distribution channels mature).

A third attraction is that competition is not yet a major problem in tne industry. Welke said there were only 500 significant software product and services companies, and that since there were thousands of market segments needing software products, the competition within a segment was often minimal or non-existant. This suggests that the profitability of software companies should be high, which is shown by a study that I did.

Software Products Pricing

The price of software products tends to mirror the price of the hardware it runs on. A minicomputer software product may take as long to develop as a mainframe product, but the former will almost always be priced substantially less than the latter. This has ramifications in the choice of distribution channels: it is probably not possible to use a direct sale force to sell software when the price of the product is below a certain threshhold.

Selling costs are one of the key strategic cost elements in the business. Companies that direct attention to increasing sales productivity and controlling selling costs are looking in the right areas. One way to reduce selling costs per unit is to sell a lot of software to the same customer (another argument for serving vertical markets).

It is very important to keep accurate records on what it costs to develop software, so that accurate pricing can be done. Welke said that software companies often think that software is a unique product operating under some new set of pricing theories. This isn't true; normal pricing theory works just fine with software products. But it is important to understand cost as well as the value of the product before one begins to think about pricing.

In general, either value pricing or the cost it would take a company to develop the product in-house sets the ceiling for pricing. The cost obviously sets the floor. Since it is better to be rich than poor, one begins to price near the high end of the scale.

There are many stories in the industry which demonstrate the freedom to price aggressively. For instance, Software AG prices their data base management system four times higher than anything else in the market. But because they provided many services along with the sale of the basic product, they have had no trouble getting their price.

Welke had several summary observations on software pricing.

1. The top price you can charge is either the economic value of the software to the customer, or about 40% of what it costs the customer to do it in-house (the 40% adjusts for the unrealistical way that most companies look at their in-house software development cost). The bottom price is either your cost or a credibility level. If you price your software too low, customers won't think your credible.

2. Price decreases will not increase sales. The threat of a price increase will increase sales. He says many companies fill up the prospect pipe line by August, and then announce a new higher price beginning October 1. The initial price on a software product is extremely important, because you live with it a long time.

3. Pricing decisions based on incremental costs are disastrous.

4. The cost of the software product is small relative to the total cost incurred by the company.

5. Ill-defined software systems can be higher priced than well-defined systems. Be nebulous!

6. * Customers do not make most software product decisions based on price.

7. The general pricing policy should be:

   * Bundle all services with a new product. Unbundle those services as the product matures.

8. It is easy to cloud prices in the contract terms. For instance, the length of the contract can be extended or contracted. Available services can be included in the base price or priced separately. You have to understand what your customer can pay and then write the contract accordingly.

Software As A Product

Welke suggested that companies resist changing the product to fit it to a customer's exact needs, although all software products probably have to be tailored a bit. Gross margins rapidly erode as customization efforts increase. His advice was to "know what your product is, and manage what it isn't." Be ready to answer complaints that your product doesn't do this or that, or that the competitor's products does this and yours doesn't. "You want to stay in the product business, and not get into the service business." Take time to design the product well, so that you can sell what you have without changes.

Welke pointed out that any software product will always have errors and design faults (design faults are the result of improper or incomplete systems analysis). He said that the goal should be to take "design faults" and turn them into priced options.

He said that it is very important to help the customer understand that the software he's purchasing will have errors in it. He said, "software is a living product", and that a customer who expects error-free software will probably cause you a lot of trouble in the long run.

Software enhancements should be offered on your timetable and at your pricing. He suggested that software companies emulate the hardware manufacturers. Even though IBM has a new technological innovation today, it may choose not to introduce the enhancement until some future date. He said every software company should have a "request for price quotation" system like IBM's.

He suggested that cost records be kept separately for each of the three key phases: creating software, maintaining software, and enhancing software.

Software maintenance should be charged to the customer on an annual basis. He said the typical industry charge was between 10% and 15% of initial software purchase price, with more companies moving to 15% today. He said this is one of the easiest sells to make in the system, and that maintenance benefits both supplier and consumer. This repetitive income stream is very important to profitability.

Successive versions of a software product require careful pricing. Can a customer move from version 1 to version 2 simply by paying the price difference? Or should the customer be required to purchase the new version at the new price?

He said that most companies today support the current version of a software system and two versions back.

Software enhancements should be viewed differently from normal software maintenance. One of the people attending the conference suggested that maintenance and enhancement be priced separately. He suggested that maintenance cost 15% of original purchase price, and enhancement 10% (on an annual basis).

The Packaging of Software

Many companies overlook how they package and promote their software product. Software is an intangible product to many people, so it is important that the documentation reflect the proper image; it should be done well and look nice.

Welke said that few companies really think through the preparation of the documentation. He suggested that someone spend a lot of time thinking about who has to use this software, how the software will be used, and what could be done to make it easier to use. He gave two analogies. The first was the consultant who suggested to John Deere that they manufacture and sell a tractor that had air conditioning and a high quality stereo system. They had thought about what it was like for a farmer to drive a tractor all day during hot summer months. The Deere tractor rapidly became one of the most popular tractors sold on the market. The second analogy was elective surgery kits. These are kits for simple operations, such as a tonsillectomy. Such kits contain all of the tools needed in the operation. A doctor opens up the package and unrolls the surgical instruments. He uses the first instrument on the left, and proceeds to the right until he has used the last instrument, when the patient should be sewed up and the operation completed. Software documentation should get similar attention. Think through all the steps the customer must go through to effectively use your product.

There should be several different types of documentation: one for the person who is going to make the purchase decision, another version for the user, and a third version for the technical people who have to support it. He also suggested that documentation be provided on different kinds of media: video cassettes, floppy disks, on-line storage, etc. The packaging activity should be well financed.

He said documentation should always be typeset. Companies should use expensive notebooks, since they it will often appear in an executive's office in an open area. He also suggested that a training session be devoted to learning how to use the documentation.

How a company installs a product also impacts the image of the package and helps determine the price that can be charged. Lots of software companies send tapes and a set of documentation through the mail. Some few companies, where the price of the software can support it, will take total responsibility for installing the product by sending installers to the site and remaining at the site until the system is up and operating. Then another group of people will come in to do some training, and a final group of techreps will appear to hold hands through the first day or so of use by the customer. Companies seem extremely willing to pay lots for this sort of service. The software product itself is only one element in the marketing mix: it seems that there is an opportunity for specific software companies to differentiate themselves by being extremely good in other elements of the marketing mix (service, support, etc.)

A given software product can often be sold in three or four different market segments simply by changing the documentation and targeting it to those segments. This document-tailoring can substantially increase the size of the available market.

Software Company Cost Structures

ICP surveyed 30 software companies to understand the cost structure in this business. They separated the firms into companies which produce system software, and companies which produced applications software.

COST AS PERCENT OF SALES:

Cost Category	Applications Company	Systems Company
Sales and Marketing	27%	37%
General and Administration	20%	20%
Product Cost (enhancement, maintentance, documentation, etc.)	35%	31%
Profit before tax	18%	12%
	100%	100%

SOURCE OF REVENUES

Revenue Category	Applications Company	Systems Company
Software Product	73%	82%
Modifications	13%	3%
Maintenance	10%	11%
Miscellaneous	4%	4%
	100%	100%

The product expense category (35% of sales) often breaks up as follows:

Development:	13%
Support:	13%
Enhancement:	9%
	35%

Sale of the Software Product

There are several ways to sell a software product.

Sale of Product:
Hardly any one sells a software product outright.

License of Product:
Most people license their software products today. This is not the same as leasing a software product. There is an important legal difference between licensing and leasing that must be understood, but that will not be discussed here.

In a typical licensing arrangement, there will be a single payment at the time the system is delivered. This often is 80% of the agreed upon price, with the 20% coming once the software has been accepted by the customer. Maintenance is generally billed separately.

Rental of Product:
This provides an income stream for a long period of time. The monthly rental payment always includes maintenance. Generally, monthly rental is approximately 5% of the purchase price. Rental is often done in order to sell a software product where a department has budget problems or a spending limit.

Time-sharing Service Royalty:

Many software products companies also have a time-sharing service for customers who want to try the product out, or for customers who cannot afford to buy the product. It's amazing how profitable this service can be, and for companies that offer it, it is often a major part of their total revenue (and even a greater part of their profits).

Multiple Installation Fees:

Many customers want to purchase multiple copies of the software to run on different machines. If a company is selling system software which enhances the performance of the hardware, there should be no discount given for multiple installations. Application software can be discounted, but probably never more than 50%. A careful analysis should be done of the actual economies achieved in selling multiple copies to the same customer. Splitting such economies with the customer is an equitable sharing arrangement.

Maintenance:

Companies should try to get 15% of the purchase price annually to cover maintenance. Maintenance should never be discounted in a multiple installation sale. Warranties, which used to be one year, are now more typically six months. Most companies submit one new release of the software system per year under the maintenance contract.

Installation:

Installation costs should probably be included in the software purchase price. Two days usually covers the installation effort. Time and material charges should also be specified in the contract for any additional work that has to be done. Training can be included in a purchase price or specified separately. Companies must be careful to specify that they do not agree to go back to train new employees for free.

Demonstration Charges:

Some software companies actually charge for demonstrations. Software AG charges $2500 to demonstrate its data management software. In the early days, these demonstration fees kept the company alive. And people paid!

Software Protection

There are several ways to try to protect software: patents, copyrights, trade secrets, trade marks, or business contracts.

Patents:

Patenting software is still very much up in the air. The Supreme Court seems inclined not to permit patents. Patents aren't generally a useful device anyway, so this protection mechanism should not be relied upon.

Copyright:

Materials which are published can get copyright protection. Copyrights are good for 50 years plus the life of the author. Copyrights only protect against the act of copying, not the essence of an idea. Copyrights should be used on all printed materials but not on the program listing. Copyrights can be used on video cassettes as well.

Trade Secrets:

To get a trade secret, you must not publish. Trade secrets are not very helpful in this business.

Trademarks:

You can trademark the name of the product. Trademarks are very inexpensive (about $2.00) and are very important. A lawyer at the meeting said that it was not necessary to establish a sales history before a trademark could be given.

Business Contracts:

This is the best protective device of all. Contract law is well established, and most people abide by it. Thus, contract terms should be specified well. A contract should have the following parts:

Statement of the type and term of sale (lease, rental, license, etc.).

Non-disclosure clause.

Limited liability (liability limited to the price of the software).

What's included in the price. Be specific! Cite deliverables, number of copies of documentation, the form and content of delivery, whether training is included, etc.

Maintenance Agreement. Again, be specific. The maintenance agreement can be a separate contract if it is priced separately.

Payment terms. Specify what constitutes delivery, and when payment is expected.

Tax Liability (should always be assumed by the customer).

ADAPSO publishes model software contracts which are widely used. I have ordered a set of these and will distribute it to anyone who requests a copy.

Marketing Alternatives

If you do not have a product, there are three ways you can get one:

1. Develop it yourself. This way, you can control the structure of the program, the language used, make it easy to enhance, etc. However, it is expensive to do this and it takes a long time. It usually takes about two years before such a product is ready to be marketed.

2. Get it from a user. This shortens the development significantly, but you often end up with a product which is inadequately documented and debugged. However, these products are often already installed somewhere so you begin with a reference account. Since references are critical to software sales, this approach is often used.

   University Computer Company and Cullinane generally try to buy software rather than building it themselves. It usually costs them between $100,000 and $150,000 to get a product ready for marketing.

3. Get it from another software firm. This is a good approach if you get total control of the product. If you don't get total control (for instance, if enhancements are done by the original supplier) you have constrained your marketing activities. If you do get all rights from the software firm, you can expect to pay them between 10% and 15% royalties.

If you do have a product, you have two alternatives:

1. Place it with a software house. You should look for a software house that has a significant market presence and has already developed its marketing and sales staff. You can generally expect to be paid for five or six years. For the first year after the sale, you will be heavily involved in helping the marketing company develop the product. You should be paid money up front as enticement to help transfer the product to them. You should try to have the product installed at five customer locations before you try to sell it, to convince the buyer that your product has merit.

   Welke said that the software business is rapidly shaking into two groups: product developers and product marketers. It is important for each firm to understand which identity it will develop.

2. The second approach is to sell it yourself.

How To Sell A Software Product Yourself

1. Select an audience. It is very important to focus as much as possible. You may feel you are limiting your market, but you are also increasing your probability for success and your profit margins. Serving a narrow market segment well is a key factor in profitability.

2. Identify the prospects. A prospect is a name of a person within an organization, not just the name of the organization. There are several good sources of information on purchasers of computer software. IDC, in Framingham, has a hugn data base describing hardware installations around the country. Computer Intelligence, in La Jolla, has a similar file but is more complete on installed software. Focus Research, in East Hartford, is also a good source of data. For mini and microcomputers, the Mini-Micro survey is the best resource. Mini-Micro is a Haydn publication.

3. Prioritise the prospects. It is often good to focus geographically as well as by market segment, at least initially. You might first go after prospects in the city you're in, and then roll out geographically across the country and internationally.

4. Select a marketing approach. It is extremely important to understand the channels of distribution in the software business, and the importance of selling cost as a key strategic variable. The price of the product probably dictates whether you use direct or indirect sales channels. If the price won't support a direct sales force, you must then look for the various indirect forms of marketing: direct phone, direct mail, retail dealers, wholesale distributors, etc. Significant efforts should be given to decreasing the sales cost within each organization if that can be done without decreasing the value of the marketing mix sold. Seminars are excellent ways to get a message across, but these are still quite expensive. One way to reduce selling costs is to sell a lot of products to the same customer. Some companies will try to get the customer to come to them. They often permit the customer to deduct the travel cost if the system is eventually purchased.

5. Understand the sales cycle. It is important to think through the entire sales procedure on a step by step basis. How was the initial contact made? What was accomplished then? What follow-up calls are necessary? Who gets a presentation? A software sale is fairly complex, and it often takes five months to complete a sale.

Profitability Cycle

Most software companies go through the same profitability cycle. It generally takes two years to develop a software product, and another three years before the company is profitable. Over those three years, a company should expect to spend twice as much on marketing as it did developing the product during the first two years. The first two or three years of the selling cycle can be extremely frustrating. It is important not to abandon the effort unless the effort is strategically flawed. A company can go for a whole year and get only four or five reference accounts. However, once there is a base of reference accounts, sales start to pick up substantially and by the end of the third year the company is usually profitable.

An executive from Cincom said that in the first selling year, you need to get a minimum of 10 customers or there is something wrong with your strategy. At Cincom, any product which doesn't get 10 customers goes through a total re-evaluation. They completely redesign their marketing effort.

The Selling Effort

Sales productivity has to be about $500,000 per salesman today. Most salesmen can sell up to four products, but it is very difficult to try to sell more. UCC has 17 software products and Informatics has 19 products. The person from Cincom said that they had too many products, and if they were doing it over would probably not try to sell more than four. Thus, the ideal company might be one which sells four related software products via one sales force in a narrowly defined market segment to the same sets of customers.

During the second marketing year, sales should generally be three times what they were the first year. During the third marketing year, sales should more than triple again.

Welke said there were very few good salesmen in the software products business today, because they have to be good salesmen as well as be technically oriented and experts in the market segment being served. Because of this problem, Welke said that companies should expect to spend a lot of money to try to pull their product rather than just pushing it.

Starting salesmen today often make between $30,000 and $35,000. Experienced salesmen have no cap on their earnings, which permits them to earn as much as they want. In 1980, MSA (a major software products company) had 65 salesmen. Eight of them were earning over $150,000 each. Mac & Dodge, another software products company, had one saleswoman earning over $200,000 selling just one product to a municipal government (they were buying over $1.5 million of product from Mac & Dodge per year).

The best system is a quota plus a commission. Generally, the quota is about $500,000. Quotas for application software are generally higher than for system software. At MSA, the quota is $650,000. At UCC, the quota is $475,000. At Pansophic, the quota is $400,000. Most commission plans are extremely aggressive. For instance, if the salesman makes his quota at MSA, he'll barely get by ($30,000 to $35,000) but if he sells 50% above quota, he'll double or triple his income.

Few companies give stock options to salesmen, although people in sales management often receive options.

Welke ran out of time, and was not able to discuss advertising and communication strategy. He is sending me a copy of his notes and I will make those available to anyone who is interested.

/cf