Some Guidance on Software Pricing

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

by Paul Betten

Argonne National Laboratory
Industrial Technology Development Center
9700 South Cass Avenue
Argonne, Illinois 60439

Submitted to the
22nd Annual Technology Transfer Society
Meeting to be Held in
Denver, Colorado
July 20-23, 1997

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED.
DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.
SOME GUIDANCE ON SOFTWARE PRICING

by Paul Betten

Vitae:

Dr. Betten is responsible for the marketing and licensing of software, as well as Cooperative Research and Development Agreements (CRADAs) at Argonne National Laboratory. He has a Ph. D. in mechanical engineering from Carnegie-Mellon University.
ABSTRACT

For many organizations, software licensing represents an increasing share of their intellectual property portfolio. Because most software is new and the market uncertain, it would be useful to have a quick, "back-of-the-envelope" method of estimating the selling price. The replacement method of valuing software on the basis of total R&D expenditures tends to over-predict the value of software. This occurs because most research is not initially directed at software development per se; the research is usually more general, with a useful product emerging only later in the project. However, if the author can estimate the time and costs necessary to rewrite the software, using present knowledge, this modified replacement cost estimate is a better indicator of market value. It is proposed that 5% of this cost is a reasonable first estimate of the software "shrink wrap" selling price. The purpose of this paper is to present the replacement costs for several software licenses, discuss them, and compare them with other competing products on the market. In addition, a pricing survey has been completed for multiple users in order to estimate how one can rationally set multiple-user prices. Two distinct multiple-user pricing regimes were found: One regime, defined as "market share software," is slightly discounted from list price; the other regime, defined as "highly discounted software," is highly discounted from list price.

INTRODUCTION

It is said that in negotiations there are three prices for the license, (1) the licensee's price, (2) the licensor's price, and (3) the final negotiated price. The problem both the licensee and licensor have is determining royalties for the license. Several methods exist for determining royalties for a product. Replacement value represents a "backward-looking" estimate; that is, it looks at the capital already spent on the project and takes that value as the amount to be recovered. Because finding licensees is difficult and most license royalties are modest, some universities consider research and development (R&D) as "sunk" costs and are content if they can recover just the patent costs, which then represent a de facto replacement cost. Several other methods, such as industry royalty standards, the 25% profit rule, and net present value, are "forward-looking" estimates; with these approaches, it is assumed that a product is very close to market so that profit margins, market penetration, sales revenue, etc. can be estimated. Often, for new and unique products, these methods are merely guesses or assumptions that are "documented" only by being written on paper.

The problem of determining royalties is further compounded if the product is not well-defined because (1) the technology is not mature enough to define a product; (2) the funding spent on the research project is not representative of the product's value; (3) the market(s) for the product is not known or well defined; (4) the private capital and time to deliver a product are unknown; and (5) companies that would want to take a risk on this undefined product are not identified.

These problems occur with research programs developing both software and inventions. However, software differs from inventions in several respects. Software is generally developed for a particular use; as a product, its function and purpose are clear, and it may be ready for use. Unfortunately, there are many programming languages and operating hardware systems, and often the researcher may be using exotic languages or expensive/state-of-the-art hardware which can slow the transfer to commercial standards. Making the transfer to commercial standards can be time-consuming, and in a research environment the funding to do the transfer is often lacking; this presents another commercialization hurdle. There is also the question of whether a well-written technical manual and documentation are available. In other cases, if there is an existing commercial product, sometimes the competing product is more user-friendly and impressive, although the research product may be significantly better technically. Again, it would take time and money to modify the research software to make it visibly better than the existing commercial product. Finally, the author may become frustrated with the legal and marketing aspects involved in commercializing a software package. There seems to be an urgency with authors because they perceive that (1) the software is ready now; (2) the life span of the software is short, only three or four years; and (3) other engineers can write around or create their own similar software without violating any copyright laws.

This paper addresses pricing for new software and suggests an approach to assigning both a cost and an estimated initial shrink-wrap selling price. This paper also addresses the software licensing question of pricing discounts for multiple users, such as for local area network (LAN) usage, and how multiple-user software is priced.
Software Evaluation: The 5% Solution

In discussing and evaluating a new piece of software with an author, questions involving the modified replacement method always enter into the conversation. The intent is to provide a "back-of-the-envelope" calculation establishing a cost and a tentative, nonexclusive selling price of the software. This "backward-looking" approach has validity at this point in time because only past R&D costs are known and can be quantified. If the software is sufficiently advanced and near commercial viability, other "forward-looking" methods can be applied. The questions asked the author include (1) the resources required to create the same software today, knowing what is now known (i.e., modified replacement cost); (2) the existence of competing or developing software; (3) the status of other related inventions or copyrights; and (4) the present status of technology advancements in this field. As a note, questions regarding the total replacement cost are no longer required because this cost is no longer relevant to pricing. It is further noted that the intent is to evaluate "stand alone" or shrink-wrap software. Software that must be used in conjunction or sold with certain equipment or hardware is not considered. If both hardware and software are integrated, there is a combined price and component separation is usually not possible.

For simplicity, "replacement cost" is defined as the total R&D cost of the project. Because the R&D occurred over a long time, the researcher's efforts probably were diverted to pursue many tangents and dead ends; these costs are also included in the total R&D costs. The patent and copyright costs could also be included as part of this replacement cost, although such costs may be small compared with the original R&D effort. However, there may be instances when this software is part of a bigger intellectual property portfolio, and patenting costs may be significant. Some of the intellectual property may also not be pertinent to the licensed technology.

The "modified replacement cost" is defined as the R&D cost required to create the same software today, knowing what one now knows. This cost is totally application-driven, and represents the cost to recreate the product directly. This method is realistic, because it is quite possible for a staff member to be hired by a private company to rewrite the software. And a rewrite may not infringe on the existing copyright. Because the technology is changing very quickly, it is also realistic that the software could be rewritten today in a more efficient manner, using recently updated commercial software or hardware. (In those instances where the division had a high overhead cost due to the maintenance and operation of laboratory equipment, the staff cost in calculating the modified replacement cost was reduced to that of an average staff in an office setting.) To date, no staff has had to recreate their software and their estimates are taken at face value. Most project managers, however, know that programs tend to take longer than expected and add a contingency factor.

After about two years of experimentation, a number of iterations, and price comparisons to market products, it was empirically found that 5% of the modified replacement cost resulted in a reasonable first estimate for a nonexclusive selling price. For example, if the modified replacement cost is estimated to be $100,000, the estimated selling price of the software would be 5% of this cost, or $5,000. This represents the price a university, national laboratory, author, or software distributor might set if they were selling it directly to the public; it is a one-time payment for a nonexclusive site or company license with no right to sublicense. Although the 5% approach is empirical, this approach appears to predict a market competitive price. Because the software is economically priced and market competitive, fully operational, and with full documentation, most companies would probably purchase it rather than try to invent around it. Further, only a modest 20 sales are required to recover the modified replacement cost. In considering the possibility of an exclusive license, one would propose summing up all the nonexclusive licenses that would be sold at the market price. The 5% of modified replacement cost could be a first estimate of that market value.

CASE STUDIES

The modified replacement approach has been employed for a number of software packages. Four case studies are shown in Table 1. The replacement costs, modified replacement costs, and selling prices are shown, along with price estimates for competing products.

In the table, the software case number is noted in the first column. The column entitled "Total R&D Cost" is the total cost required to develop the package and included the cost for all of the researchers working on the project at full salary, including benefits and overhead costs. The column...
for "Modified Replacement Cost" includes the same costs as the preceding column, but it also incorporates the staff's expertise and lessons learned if the software were to be rewritten today for a direct application. The estimated selling price, based on 5% of the modified replacement cost, is also noted. The "Revised Selling Price" represents changes made to the 5% price based upon competing products, "forward-looking" pricing methods, and other considerations. The last column, "Competing Market Products," is a reality check on pricing and represents the cost of similar products on the market, if any.

DISCUSSION

The total R&D costs (i.e., total replacement costs) calculated in Table 1 are significantly higher than the modified replacement costs, by a factor ranging from three to six. This is expected because the total R&D cost will include a number of R&D tangents and dead ends that an application-directed effort would not have. The initial selling price estimate, based on taking 5% of the modified replacement cost, is within plus or minus 15% of the actual selling price in three of the four cases noted. Considering that these pricing estimates were made about one year before the pricing concerns and/or software negotiations had begun, this agreement is good. The fourth case involved several commercialization options, and it was decided that wider distribution was possible if it was sold in conjunction with another well-known software package.

The last column shows the selling price of comparable products presently on the market. This comparison can be viewed in two ways. The first view is that no comparable products existed for two of the four software packages, and a wide price range existed for the other two. It makes sense that an R&D organization would create software that did not have a fully developed commercial market yet seemed to have partial application or potentially address an industrial problem. This is the proverbial technology push, a technology looking for a market (as opposed to a market-driven situation, where a market is looking for a technology). The second way to view this comparison is to say that two, and maybe three, of the software packages did have "vaguely similar" or "somewhat related" equivalents in the open market. The disparity in pricing of "Competing Market Products" comes from the fact that the low-end products are not very versatile, and the high-end products are indicative of a lot of tailoring to a specific end-user. Thus, the big pricing spread indicates that the technology has entered the market, but it is not broad enough or versatile enough to meet the general public's needs. Nevertheless, it is a good reality check to determine the pricing of market products, if they exist. It is not necessary to achieve an exact match, but only to get a sense of an acceptable "going rate." There is great variety in product function, user-friendliness, technical support and maintenance, upgrades, etc. and price comparisons alone may not necessarily be indicative of the value of a product.

It is a good procedure to verify R&D costs and pricing strategies by some other method. The "25% profit rule" and industry standard royalty rates were tried, but they had limited applicability.

In cases 1 and 2, the modified replacement cost exceeds $250,000, and the resulting 5% estimated price exceeds $12,500. Intuitively, this seems high for a shrink-wrap software package (but possible, as noted in these two cases), although it may not be high for a customized package, or software with excellent technical support and constant upgrades. However, our organization, like most nonprofits, is not in the service or consulting business of customizing software. Thus, the reader is suggested to use caution in applying the 5% approach for very high modified replacement costs, or perhaps a lower percentage number may be applicable.

SOFTWARE PRICING FOR MULTIPLE USERS

All of the software summarized in Table 1 had the potential to be licensed to multiple users, either as part of a local area network (LAN) or as high-volume sales. The questions then arose, "What are typical LAN or volume discounts?" and "Is this a reasonable pricing structure?" The intent was not to perform an in-depth or definitive study, but simply to understand the trends for multiple users and price. A software pricing study was made in late 1995, early 1996. All of the software had 10 or more users, with the exception of one package that had five users. There are 34 data points based on 12 multiple-user software packages. The software packages include SQL Windows, Notes Client, Adobe Photo Shop for Windows, Adobe Acrobat, Lotus for Windows, CC-mail Desktop, Informix 46L, Saber Menu Pack, Macromedia Director V4, Solares PC Administrative Server, WordPerfect, and Deals.
The multiple-user pricing data were plotted in two formats (1) normalized to the single-user selling price, and (2) normalized to the total cost. The data are plotted in Figures 1 and 2, respectively. In Figure 1, the software price clearly becomes discounted with more users. Surprisingly (see Figure 1), it was found that there are two distinct regions in each graph. The first region is defined as the "market share" software and is represented by the lower-priced, high-volume software that is discounted modestly with more users, such as WordPerfect and Lotus. These packages are discounted modestly with volume, and after about 10 sales they tend toward an asymptotic price at between 60 and 80% of the single-user selling price. The market share software is very competitively priced and tends to fall in the $100 to $400 range. The second region is defined as "highly discounted" software and represents higher-priced, low-volume packages where the price drops dramatically with the number of users, such as Solares Server and Saber. These packages drop rapidly in price after about 10 sales and tend toward an asymptotic price between 5 and 30% of the individual package selling price. The highly discounted software tends to have a high single user price in the $1,000 to $5,000 range. In both regions, prices appear to continually decrease with volume.

Figure 2 plots the data as a function of total cost for multiple users; the dashed line represents the no-discount line. This figure can be considered as the integral of the prices shown in Figure 1, as it represents the total cost of the software per additional user. For "market share" software, the total cost is directly proportional to sales, because the multiple-user discounts are modest (in the 30% range). The "highly discounted" software appears to have a more asymptotic behavior than the "market share" software.

There are many marketing strategies in promoting a product or for gaining market shares, as well as approaches for discounting older software versions before new versions, with new and improved features, are released. For the limited data shown here, there seem to be two marketing approaches that are commonly employed. One could speculate on the nature and reasons for the two different regions that are observed, but this speculation is left to the reader. The single user price does seem to be a good indicator of multiple user discounts, although exceptions do occur. One should take care in applying these data to other software products, first verifying that your software is similar enough for comparison.

CONCLUSIONS

The modified replacement method of valuing software has been proposed, together with the idea that by taking 5% of that value, an estimate of a nonexclusive selling price can be established. The "modified replacement cost" is defined as the R&D cost required to create the same software today, knowing what one now knows. This cost is totally application-driven, and represents the cost to recreate the product directly. This method is realistic because it is quite possible for a staff member to be hired by a private company to rewrite the software. Other "forward" pricing methods should still be employed to validate or change the software price. The evaluation of competing products on the market is essential as a pricing reality check and must be used in establishing a market price.

The 5% approach was developed over two years of licensing and evaluating software, and it should not be assumed to be a finalized rule. This process will certainly continue to evolve and will be modified and reevaluated as more experience is gained. The majority of technology licenses are not "home runs" bringing in millions, but "average" licenses that get the technology into commerce and may be very important to the licensee. The modified replacement method, as well as the multiple-user data, was developed from those average licenses. It should be noted that value and price are not always synonymous and that technical support, user-friendliness, well-written user's manuals, etc. are also very important in setting a selling price.

The three prices for a license, (1) the licensee's price, (2) the licensor's price, and (3) the final negotiated price, still exist. However, with the modified replacement method of calculation and an understanding of multiple-user pricing, the licensee now has another rational basis for negotiation.
REFERENCES


2. AUTM Technology Transfer Practice Manual, Association of University Technology Managers, Inc., Vol. II, Part VII, Chapter 4, February 1994 (pp. 4.1-4.70)

3. Personal communications with Ronette Stec, Argonne National Laboratory.

Table 1
Summary of Replacement Cost Software Costs and Pricing

<table>
<thead>
<tr>
<th>Software Case Number</th>
<th>Total R&amp;D Cost</th>
<th>Modified Replacement Cost/Estimated Selling Price (5% of MRC)</th>
<th>Revised Selling Price</th>
<th>Competing Market Products</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$1,500,000</td>
<td>$350,000/$17,500</td>
<td>$20,000/yr¹</td>
<td>None</td>
</tr>
<tr>
<td>2</td>
<td>$650,000</td>
<td>$260,000/$13,000</td>
<td>$15,000²</td>
<td>$2,000-$50,000</td>
</tr>
<tr>
<td>3</td>
<td>$47,000</td>
<td>$11,000/$540</td>
<td>$500³</td>
<td>None⁵</td>
</tr>
<tr>
<td>4</td>
<td>$190,000</td>
<td>$32,500/$1,625</td>
<td>$500²,³,⁴</td>
<td>$1,000-$10,000</td>
</tr>
</tbody>
</table>

1- This software will be sublicensed as a yearly lease.
2- This software may be embedded in other products, and royalties may be negotiated as a percentage of the total package sales price.
3- Will be sold as a shrink-wrapped package.
4- To be sold in conjunction with another software package as an add-on. May have educational and other discounts.
5- No software package was found, but a web-site entrepreneur provides something vaguely similar at $25 per use.
Figure 1
Normalized Price Ratio per User

Normalized Price Ratio (P_{multiple user} / P_{single user}) vs. Number of Users

- Photoshop
- Director
- Notes
- CC:Mail
- SQL
- WordPerfect
- Informix
- Lotus
- Deals
- Saber
- Acrobat
- Solares
Figure 2
Normalized Total Cost Ratio per User

[Graph showing various lines representing different software programs such as CC:Mall, WordPerfect, Lotus, Notes, Informix, Director, Photoshop, Saber, Solares, Deals, Acrobat, SQL, and No Discount Line. The x-axis represents the number of users, ranging from 0 to 60, and the y-axis represents the normalized total cost ratio per user, ranging from 0 to 40.]