Editorial

The Software Dilemma:

How is it possible to simultaneously make software widely available (and low priced), yet reward the producers of good software with adequate compensation for their efforts?

By Carl Helmers

Conventional wisdom has it that proprietary software must come at extremely high prices, commensurate with concentrated work on the part of a small number of dedicated and thoughtful programmers. After all, this wisdom has it, we'll only sell a few copies of package X anyway, so why not keep a tight lid on it and charge as much as possible?

This conventional wisdom has worked well in the past, when the typical computer system might cost upwards of $10,000 or $100,000. But when the typical computer system comes in at a price on the order of $1000, paying prices which are of this same order of magnitude for software packages is not a very likely move on the part of the individual purchaser with his or her personal budget.

In the personal computing field we are participating in a market phenomenon characterized by a change from the situation which supports the conventional software wisdom, to a new situation which has its own characteristics. More and more people are getting into the swing of things with computer use, and thus more and more people have needs which can and should be filled by specialized software products. Where computers are concerned, when we talk about a 100,000+ person active individual user market as we do today, we are for the first time talking about the potential for mass marketing of software in ways unheard of in the conventional wisdom of computing. Establishing a new "conventional wisdom" is clearly required; as a step toward that goal, this paper provides a survey of the prospects for mass marketing of software, and a solution of the software dilemma posed above.

Let's Draw Some Parallels: Woodworking

Like many individuals, I dabble a bit in the arts of crafting furniture. Suppose, for example, that I want to build a nice, neat contemporary rolltop desk for my study. As an individual with limited time available for such leisure crafts activity, I'd probably want to start with an existing design rather than working out all the details myself. In seeking the end product of a rolltop desk, I'd be in the same situation (as a wood craftsman) as the owner of a computer system desiring a compiler, assembler, application product or peripheral. I know in principle that rolltop desks exist and that in principle I could design then fabricate one, or use an existing one as a mental model with my own variations. But to save time and possible mistakes I might want to find some source of a "proven" design with detailed information on achieving the goal of a rolltop desk. Well, in the world of woodcrafting, as in the world of photography, the world of live steam model engines, or the world of backpacking, there are numerous sources of information including ready-made designs and techniques. I refer, of course, to books which are just published products with specific orientation or theme.

Similarly, when I have a computer system and I know that some neat language or software development tool exists, I also know that in principle I could write such a package myself using my own design or general design concepts taken from any

Continued on page 68

This editorial consists of the text of a paper delivered at the First West Coast Computer Faire in April of this year.

About those missing mailing wrappers and the May issue:

A strike at the printing plant was responsible for May BYTEs arriving late to subscribers and for May and June issues being mailed without the customary brown wrappers. The wrappers will be restored as soon as our printing situation is restored to normalcy. ▪

Continued from page 9

number of existing computer science textbooks. But I'd really like to somehow buy the design in a completely documented form so that like the plan of the hypothetical rolltop desk, I could implement it literally with custom modifications. Tutorial and "how to" plans books for specialized fields such as those mentioned above are widely available already, and at prices well within the range of an individual's budget. They are marketed in large quantities because large numbers of individuals use the information; outlets range from mail order book services to retail stores. Drawing out of this parallel between individualized computing and individualized "anything else," it should be obvious what the solution of the software dilemma is: Publish detailed plans and tutorial information for software, on a scale commensurate with the size of the market. Publishing ideas is an activity which has a long and distinguished history, and yields both personal and financial rewards to those who engage in the practice, as well as real benefits to those who purchase the products. Let's turn now to the application of this concept to the software designs of the computing world.

The Ideal Model of a Published Software Product

When we talk about publishing software at the present state of technology, we are talking about a product which is akin to the detailed design of the rolltop desk mentioned earlier. It is a product which serves as the starting point for the home software craftsman, not a recipe which will fit without thought into every conceivable system. This will change a bit as the systems in the marketplace become more refined, but the nature of the computer as an intellectual amplifier tends to require a certain level of technical familiarity on the part of its user. (This is the element which distinguishes the general purpose computer from the applications oriented dedicated computer such as a 4 function calculator or oven controller.)

In order to make a software package which is optimally configured for the customer's standard or customized use, there is a certain minimal level of documentation which is required. This level of documentation is not necessarily needed by all users all of the time, but is in many respects akin to the reference books for integrated circuits: When a question needs to be answered, it is good to have the information needed to zero in on the answer. Here is what I consider to be adequate documentation:

- Users' manual textual materials concerning the "standard" uses and limitations of the software. Here is where we find such information as standard IO patch points, relocation tables, etc.
- Complete object code, preferably machine readable along with machine readable relocation information.
- Complete source listing of the package including source language and generated object code for each statement.
- Program logic manuals and tutorials on the design of the product are an excellent option.

The idea is to include enough information to allow the user to do routine field alterations, including relocation. The idea of a published software product is to compile all this information together in a comprehensive book form, to be sold at prices characteristic of books, as opposed to the past history of software prices for applications and system software packages. The technology of printing covers all the portions of the "complete" package except machine readable code, at least in the minds of most people. However, as we have demonstrated with experiments published in BYTE, printing technology also covers machine readable representations as well.

Varieties of Machine Readable Representations

User convenience demands that a software product be made available in some form of machine readable representation. While it is certainly possible to take an object listing in printed form and type it into a processor by hand, this is a long, tedious and error prone process. To complete the functional definition of "adequate documentation" given above, we need a form of machine readable object code at minimum, along with machine readable relocation information. Fortunately there exist several technologies which can be employed for this purpose, which I'll review here.

ROM Releases

This is the most expensive medium presently available for reproducing software; however, it has utility in the convenience of use provided by built-in software. In terms of practical products, however, this form of software will most frequently show up in manufactured products preloaded at the factory, rather than user integrated software. A ROM program is difficult to achieve in a relocatable form, and has a certain permanence which is both its advantage (convenience) and its disadvantage (difficulty or impossibility of patching). This method has been successfully employed in several desk top calculator packages in the form of ROM software options, and in such personal computing products as built-in BASIC interpreters and monitors. For end user markets, this form of software can dispense with all but the user oriented manuals in most purchases, since modification is impossible.

Magnetic Digital Media Releases

As more and more floppy disk products, large and small, come to market, the use of the magnetic diskettes for software releases is becoming common. Similarly, the Philips and 3M digital tape media with standard digital recording techniques can be considered as vehicles for release of machine readable data. However, the price of the media and the costs of digitally recording and verifying each copy tend to make this method of delivery limited. It is used, quite naturally, as the vehicle for delivery of floppy disk operating systems from the manufacturers of drive interfaces, but there the writing and testing of a diskette full of data falls out of the expected quality assurance tests prior to packing and delivery. The same is true for the other forms of hardware which have related operating system software products that can be recorded.

Paper Tape

This venerable medium has been in existence longer than the modern electronic computer. No survey of distribution media would be complete without mention of it. Very reliable means exist for reading paper tapes into computers at quite economical prices, well under $100 for the peripheral. As a distribution medium, however, paper tape in my opinion suffers from several disadvantages: It is inconvenient to store, bulky, prone to create a messy tangle due to manual handling with inexpensive peripherals. Whether my opinion is supported in the marketplace is another question altogether.

Audio Media Releases

One of the most useful and practical vehicles for the distribution of software is likely to be the use of audio recording media. Here we can identify two principal methods of distribution; recording on tape cassettes or other magnetic tape audio media, and recording on the audio equivalent of a read only memory, the phonograph record.

The technology of recording on tape results in a product with a fairly high unit cost for each copy of the information. To this must be added the cost factors associated with normal printing of the rest of the documentation. Cassettes, as a recording method, are a logical choice for custom software, or small volume situations, but the high degree of manual labor associated with each copy argues against the practicality of large production runs in this form.

The technology of making phonograph records is, on the other hand, a well established mass production technique which can be adapted to the software distribution without much variation from standard methods. To illustrate the point and to test the concept, I made a test in the spring of 1976 at the suggestion of David Fylstra, an associate of mine who is also a homebrew record maker. He arranged for the cutting of a test record with the audio format of my personal monitor program, circa March 1976, to test out this method using a master record cut on standard recording industry equipment. Depending upon size and quantity of pressing, the costs per record run well under $1, which is hard to meet with the cassette duplication method.

Machine Readable Printed Media

As a final option, there is the use of machine readable printed formats for object code and relocation information in the release of software products. This is a form which was suggested to us at BYTE by Walter Banks of the University of Waterloo, and with which we have been experimenting in the pages of BYTE. In this method, an optical reader is used to scan printed materials which have been formatted into a series of bars corresponding to the digital information. Because of constraints in the design of the layout and the method of scanning, it is possible to simplify the scanner designs to the point where a very inexpensive peripheral is used together with some adaptive software which takes care of the speed tolerant input scan. The beauty of this method is that it "comes for free" in so far as actual production costs are concerned. Why is this true? The reason is that the 200 to 300 pages of documentation needed to support a systems software product with perhaps 12 K bytes of object code require only an additional five to seven pages of machine readable bar code copy, hardly affecting the economics of the book at all. These 200 to 300 pages of documentation are required by the product concept, whether or not there is any other form of machine readable code made available.

Economics of Publishing

With media established, and a product concept outlined, what about addressing the problem of rewarding the producers of software products? Here, as in any area of publishing, the answer is quite simple. The publishing house judges whether the particular software package is in its view a readily marketable product with a certain minimum press run potential. If so, the publisher puts up production capital, where the author puts up intellectual capital in the form of his or her work. It is a risk situation in which both parties are making a speculation that readers will purchase the product; as in numerous parallel situations throughout industry, authors and publishers work on an agreed upon split of any rewards from success in the marketplace.

Applied to the software publishing variant of this business, the author's intellectual capital is in the form of the program, its source code, its object code, and its documentation; the publisher's contribution is the marketing organization, the technical editing of the manuscripts, and the technical details of book preparation. Other than the specialized content, the method of operation and the details of the arrangement are not much different from publishing any item. Rewards to authors now become a small amount (in absolute terms) of royalty recovered from orders of magnitude in sales for successful software book products.

Proprietary Products

The problem of protecting and keeping software proprietary is no longer a major "new" issue when publishing of software is contemplated. How many people extensively copy from books? Very few, and if they attempt to make a regular practice of it they would tend to be prosecuted by publishers under copyright law. In publishing software, an implicit or explicit license to copy the copyrighted materials for personal use and modification is part of the bargain; the price is low enough so that if you want your own user documentation, you buy your own copy of the book (even if you may have been using object code derived from your neighbor's computer). Since the documentation is a necessary component of use, no sales tend to be lost in the long run due to the fact that object code can be swapped around.

Conclusions

What I have endeavored to show is that there is quite some potential for the sale and distribution of software using conventional publishing techniques with modifications to suit this type of product. By publishing software along with machine readable code, we end up with a way to make the products widely available, yet retain the desirability of compensating authors for their efforts in proportion to the success of the product.■

Articles Policy

BYTE is continually seeking quality manuscripts written by individuals who are applying personal systems, or who have knowledge which will prove useful to our readers. Manuscripts should have double spaced typewritten texts with wide margins. Numbering sequences should be maintained separately for figures, tables, photos and listings. Figures and tables should be provided on separate sheets of paper. Photos of technical subjects should be taken with uniform lighting, sharp focus and should be supplied in the form of clear glossy black and white or color prints (if you do not have access to quality photography, items to be photographed can be shipped to us in many cases). Computer listings should be supplied using the darkest ribbons possible on new (not recycled) blank white computer forms or bond paper. Where possible, we would like authors to supply a short statement about their background and experience.

Articles which are accepted are typically acknowledged with a binder check 4 to 8 weeks after receipt. Honorariums for articles are based upon the technical quality and suitability for BYTE's readership and are typically $25 to $50 per typeset magazine page. We recommend that authors record their name and address information redundantly on materials submitted, and that a return envelope with postage be supplied in the event the article is not accepted.■