RECORD OF SOCIETY OF ACTUARIES
1984 VOL. 10 NO. 4B

MICRO-COMPUTERS

Moderator: JOHN T. CLARK. Panelists: MICHAEL J. AKERS, ALAN P. BLACKWELL, JOHN E. MAC MILLAN. Recorder: JOY PILLARD*

Microcomputers can be used in a variety of ways in the insurance industry. Several topics will be discussed:

1. Programming languages.

2. Using the micros as an actuarial tool.

3. Use of micros to support local agency operations and to prepare policy illustrations in the field.

4. Spreadsheets and other application packages.

5. Communication between micros and mainframes.

6. Downloading from mainframe to micros.

MR. MICHAEL AKERS: Our first speaker is Alan Blackwell. He is responsible for all phases of actuarial work in life and health lines of business at his company. Al has been programming computers for a dozen or so years, and during that time, he's worked with a varied assortment of machines and used a varied assortment of computer programming languages. Al will be speaking on the general use of the micro as an actuarial tool.

Our second speaker is John MacMillan. John's 20 years of experience covers policyowner service, underwriting, group, and marketing. As Director of Individual Marketing, he works closely with other departments to develop marketing support for all the company's individual products, as well as computer proposal support. John will be speaking about the uses of micros to support local agency operations and to prepare policy illustrations in the field.

I'm your third speaker today. I have worked both in the development of actuarial systems and as an end user. This includes experience with pricing, projection, and valuation systems for traditional and nontraditional products. I've worked in the mainframe, minicomputer, and microcomputer environments. Today, I'm involved with our ongoing work to enhance the ability of our user community to move among these various computer environments and systems. My topic is going to be communications between these various kinds of computers.

* Mr. MacMillan, not a member of the Society, is Director of Individual Marketing at the Manufacturers Life.

MR. ALAN BLACKWELL: I've identified an assortment of languages including Basic, Fortran, APL, Pascal, C, Modula-2, Assembler, and Cobol which are the prime ones used on micros. There are also some that I call "semilanguages" such as data base programs (Dbase II to mention one), program generators and spreadsheets.

Basic. That's the language that most people are familiar with because it's been implemented on more micros than virtually any other language. Having worked with an assortment of micros, there's no set definition of Basic that I can find. There is a standard out now which tries to encompass all implementations of Basic, but I think it's too big to implement on any micro, including the IBMXT. A very general purpose language, some of the limitations are:

1. The precision of real numbers. This doesn't seem to pose a problem until you try to do something like a universal life proposal for 60 years with interest compounded monthly, and find out that the result is $10,000 off at the end.

2. There is a lack of structure in some implementations of the language. The initial version of it was designed to run on very small machines. Because of that, there was no forced structure and it's very easy to write almost unmaintainable code. I'm sure you are all aware of writing spaghetti programs where GO TO statements go to other GO TO statements, and they go to other GO TO statements.

3. It is not portable among the micros due to the extreme differences in the dialects of Basic. We've moved our programs twice. Most of them only get 90% converted. Every program we've had has had to be slightly rewritten when we've changed to another machine. It seems to lack in some good system tools. And generally, subprograms are not available. Most of the time you find it as an interpretive language, meaning every instruction is converted into the machine language and then carried out. When it hits that instruction again, if there's a loop in the program, it'll have to interpret it again. And it tends to run slow in that mode of implementation. There are some compile versions on micros, and I understand that there is a lot of structure available on some micros and a lot more capability on some.

I've seen quite a variety all the way from an Atari I have at home to a Hewlett-Packard Basic which looks like a superset of everybody else's favorite language. So it is out there, but it does have those limitations.

Fortran. If you've programmed in Basic, you can probably program in Fortran very easily. The code is very similar. It does require more structure. There is "data typing" where you can identify some of your variables as integers. This does speed up processing because it takes smaller amounts of memory and it does run faster. The biggest drawback with Fortran is someone has to learn how to properly handle a compiler which is, in some cases, not a simple process. It is not always available on smaller micros because it was originally designed to run on mainframes and has only recently moved to minis and micros. The biggest benefit of Fortran is it does not require line numbering other than reference lines. It has generally better matrix handling because it will allow several-dimension matrices where most basic languages limit you generally to two.

APL. That's short for A Programming Language. APL is a very powerful scientific language. It does have very outstanding matrix handling capabilities. For example, I've been recently working on tax reserves. In APL, as opposed to generating one reserve factor for one age and one duration, you can generate all the durations for that age at once, and then you can do all the ages for all durations, and then do that for all the tables. And if the code is written properly, you can have it all run through the same sequence. APL seems to not have a deterioration in speed as much as you would think when you increase the number of variables in the matrices. So it does have that type of power. It also requires special symbols on the display. APL seems to have a very long learning curve. And I think the biggest drawback with it is that it's easy to write in a fashion so you can never read it again. If you're a fan of Byte Magazine, one of my favorite authors in there has mentioned it as a write-only language. It seems to be very useful where you need a quick and dirty programming effort.

Pascal. The language seems to be used mostly in academic circles. It's probably the best teaching language for new programmers. Paraphrasing from a similar Byte Magazine, a lot of professors that teach programming would wish their students coming out of high school had never seen Basic and learned all those bad habits because they have to spend the first part of the time unlearning the bad habits. They say Pascal is the best for the new students learning it because it forces the programmer to write structured code. You have to do a lot of documentation. You can't just generate variables as you go through your programs. There's a special section that says "variables", and you have to identify each one and assign it a data type. It's been called a voluntarily worn straightjacket. Probably the best feature of it is that it's very easy to maintain because it is almost self-documenting. Now it does have some GO TO capabilities, so if you really try hard, you can write a bad program in Pascal. But it discourages it greatly. The ones I've seen all work under a compiler and require fairly sizable micros. Under the 16 bit machines that run 256K+, it seems to be no problem. And there are some very good Pascals out for the IBM micro. For actuaries, my biggest complaint is that it lacks the specific exponentiation operator. And I've heard that it does not seem to do well for extremely large programs. It may be that it was written as a learning language and does not have the power because of that.

Modula-2. It's a superset of Pascal, and it was primarily written by Nicholas Wirth, who wrote Pascal, to cure most of the problems that Pascal had. It has a lot more structures for input/output, some of Pascal's limitations. It still does not have an exponentiation operator.

C. A fairly new language hitting the scene is C. It's what I'd call a middle-level language, somewhere between the machine language and all the high-level languages that we're used to working in. It's useful for writing operating systems and things of that nature. Its power is transportability from one CPU to another.

Assembler. That's programming using the machine's instructions. It takes an extremely long time to learn and a long time to code. You have to learn how macro assemblers work. You have to understand how the operating system

of the computer works. You have to understand peripheral handlers so you can tell it how to turn on the disc drive. It's not one that you'd be intending to use for actuarial work.

Cobol. There is an implementation for the PC. I don't know if there are implementations for other machines. It requires quite a bit of space to compile Cobol. It's an extremely verbose language. Cobol seems to be good for accounting and inventory programs. It has special functions for those types of programs, such as sorts. You have to know how to use compilers. It's harder to do complicated matrix functions in Cobol. Matrices are really considered to be tables and not matrices. And it has fixed formats for variables, which tends to cause some rounding problems on small numbers unless you assign 24 decimal places, which most programmers do not want to do. It does tend to be self-documenting. The advantage for actuaries is that it's used in mainframe computer applications. Your ability to understand it could help you debug the code of the mainframe computer programmer who's trying to write something for you.

Those are the main languages. I'll now touch briefly on Dbase II or data base-type programs. I must say I've not used them for actuarial applications. I haven't come up with an application where we need that kind of manipulative abilities on a data base.

My next topic is using micros as an actuarial tool. I identified nine general areas: product development, model office, management information, reserve factor development, maintenance of small segments of the company's business, project control, spreadsheet replacement, rate increases, and mainframe interface.

Product development. In product development, generally we will set up files of all the assumptions -- mortality tables, withdrawal tables, interest rate assumptions, expense assumptions, commission scales, death benefit schedules, etc. Draw it together with a product assumption set describing which tables to use, write the program and generate the results. It's very easy in that type of environment to try many runs under various assumptions and see the trends of results. And that's the real power. Before, you were somewhat limited; you had to make do with three or four runs.

Model office. The pricing produces cell by cell results of profits. Next you can weigh the cells based on anticipated sales, bring in surplus information, federal income tax assumptions, etc., and produce model office results. I've found that telling somebody we're going to make 16% return on equity doesn't seem to be as meaningful as to show them yearly results over the next several years of the product, which you can do with the model offices.

Management information. I think because the actuaries do have the micros, they've been charged with providing, on a periodic basis, key indicators on their particular company's business. We've used a micro on a mainframe interface to provide those indicators. We used to do graph books showing all the key movements. It used to take one person about five days to work

all these graphs by hand. Now, with the graphics package on our system, the graphs run in about an hour. I think that's one of the better uses of micros.

Reserve factor development. Most companies have mainframe systems to produce reserve factors or at least store them for valuation. If you've got a system where you have to produce the factors and load them out on to the mainframe, obviously the micro makes an excellent place to develop reserve factors. Card images can be generated on your micro and loaded on to the mainframe directly, which eliminates errors.

Maintenance of business. In the past, it's been typical for actuarial units to keep cards on very small blocks of business, such as annuities in the payout stage. If you have a simple application, you can take the card files and put them on a spreadsheet on a micro, and it's much easier.

Project control. Using some simple programs, the department manager can keep track of the status of all his projects. Using a spreadsheet, you can store things like received date, due date, completed date and source. This information can be used over a period of time to measure the department's strengths and weaknesses and personnel needs.

Spreadsheet replacement. Most actuarial departments over the years have had to do a lot of simple calculations on columar pads (spreadsheets). All of these applications can be moved to the micro using a so-called spreadsheet program.

Rate increases. Actuaries working in the individual health area are painfully aware of the need for frequent rate increases. It is possible to get a summarized version of loss ratio data put on a micro and programs developed to track loss ratios by form by state. Also, when a rate increase is needed, programs can be written that will massage the data and produce the required actuarial memorandum.

Main frame interface. In situations in which the actuarial department takes data from mainframe printouts and enters it into a microcomputer, it is often better to transmit the data directly to the micro via a modem.

Some mainframe programs used by actuarial departments, such as programs to build GAAP factors, require extensive amounts of input to be run. In some instances it makes sense to write a program on the micro to generate the input to the mainframe program and then transmit it directly.

MR. EDWARD COWMAN: Which of those languages are you actually using in your shop? Is there another that you would prefer to use having looked at what's available today?

MR. BLACKWELL: We're using compile Basic in our shop, but hoping to get a better Basic. I must say, I'm no longer working on a micro, I'm working on a Prime mini. And the Prime's major limitations are availability of software. We're using the Basic they have, hoping for an implementation of so-called True Basic to come out which has most of the capabilities that Fortran has and the structure that allows you to write nice readable code

and still fits in with our shop. One of the things we have that limits us is code that we don't want to have to rewrite. Other than the languages I programmed in, I'd be interested in Modula-2 if an exponentiation operator were added.

MR. BRIAN JENKINS: I program in most of the languages that you went over; unfortunately, I think you did a very bad review on "C". "C" is at least as powerful as Fortran, if not more so. The only problem is for some items like matrix operations, where you either have to do a little of your own creative work or else you have to buy a package. It does not have explicit exponentiation functions, which I agree is a problem. One of the languages that you did miss that is starting to be used with micros is PL1. It is actually quite a good combination of a Cobol, Fortran and "C" type environment.

MR. HENRY RAMSEY: You said that the ANSI Basic is coming. I've read just this month that the two Dartmouth professors who developed the original Basic are developing the True Basic. They see that as two warring factions of the many variants of Basic to come out. The example of the code that they had in True Basic looked like Pascal to me.

The text essentially said that they'd almost made GO TO statements illegal. Is that the primary reason that you're looking for it, because it's so much like Pascal? Or are you looking just because you want to turn your existing Basic into something more documentable?

MR. BLACKWELL: I want to turn the existing Basic into something more documentable. But we would like to go to structured programming, top down, section by section. Our particular implementation, the prime Basic, is very old. Programming in letter-number variables is grossly unreadable. But we would like to go to a more documentable Basic. And when you get there, you'll notice, except for some things like specific data typing, that Pascal and Basic and Fortran do all look alike.

MR. JOHN MACMILLAN: My topic is the use of microcomputers in illustration preparation in the field. First, I want to tell you who my company is, where we are in the use of microcomputers to produce policy illustrations, why we made the decisions that put us in this position, what some of the problems are that we are having with that decision right now, and where I see the future going in micro applications to illustrate products.

In the United States my company sells only life insurance and annuities on the individual side. Our group area does have pension products, and group life, and is moving into group disability. But I represent only the individual insurance side of the house. The company is totally divisionalized, and I'm in the U.S. division. Our method of distribution is a branch office system. We sell through a whole time agency force, but we also sell a lot of our business through brokers. We expect about 40% of our business measured on annualized commission to come from a range of brokers. Right now, we do business with stock brokers, primarily annuity business. We do some business, also fairly heavily life, with the kind of producers who call themselves the big independents, which usually means they have whole time agency contracts with a dozen companies. And we also look for surplus business from agents from other companies where we have a product that the agent appears to need that is not available from their primary company.

Our market is the upwardly mobile, the affluent, the person with tax problems. Primary emphasis is on the small business owner in the United States. The other thing that we have done, however, is try to look for market niches where we think the company can do well, deliver a product that meets a specific need, and make a profit. One of those is, for us, the annuity marketplace, particularly court settlements. And another that we have done quite well at is the supplemental compensation marketplace, the kind of specialized high cash value product that the Fortune 500 or the large corporation likes to buy to fund that kind of employee benefit program.

In terms of products, we have three major product lines. Our traditional product line is a participating portfolio average product. We do use direct recognition of policy loans in setting dividend scales. We have a nonparticipating, interest-sensitive product line, including two universal life products and another interest-sensitive whole life type product. We just introduced our variable products, variable life. We're now working on variable annuities.

Given our marketplace, and the range of products we offer, illustrations are particularly important to us and our illustration usage is particularly heavy. Perhaps more important than usage is that we use highly complex illustrations that usually illustrate tax advantages to the buyer. If there are not the traditional tax advantages available (for example, we don't believe you can minimum deposit universal life and obtain a tax deduction for interest payments), we create other kinds of tax advantaged illustrations. Right now, for example, we are working on an asset exchange approach to selling universal life as a single-premium product. This kind of illustration is usually very complex.

About three years ago, we had an illustration system that ran on our home office mainframe and had been developed, enhanced and supported by in-house programmers. Our home office officially supported no outside illustration services. Many of our offices had made individual arrangements with outside suppliers to supplement what the home office illustration system provided, but we did not sponsor any form of outside illustration service.

One of the problems of this for us was the cost of developing and maintaining very complex illustrations. With only 700 agents as a user base, and our brokerage solicitors, the cost of developing any new illustration was spread over a very small user base. It used a lot of our money to develop a complex illustration and the development was usually fairly slow. None of our programmers were conversant with how life insurance was sold. We had to have people who could define for the prgrammers what the illustration system had to provide in terms of output, what the concept of the illustration was and how the agent might use it to give the programmer a stronger sense of what the final objectives of the program were. This meant our development was not only expensive, but quite slow and we had a problem using the few people we had in our marketing area to define illustration output. Our branch hardware consisted of first, basic telex machines communicating with our home office over a time-sharing system running at 300 baud. We subsequently moved to word processors used essentially as dumb terminals. This improved the quality of the output

since the agent could produce letter quality illustrations and had some ability to word process the output to add comments or change column headings to some degree. But this was a very limited customization capability. One capacity they did have was the ability to store output on disc and print later, which many of our agencies used, but this did not really meet the needs our system of agents was telling us they had for illustrations.

So about three years ago, we set up two task forces in our home office, one to look at hardware and where hardware was going over the next five years and another task force to look at software -- what kind of software did we need, how were we going to use it, where were we going to be selling business, what were the agents going to be doing. The results of both of those studies were that we decided to decentralize illustration preparations. We would move illustrations to the field out of the home office main-frame environment which meant we had to look for a vendor that was going to be in the micro business for a while. And we chose IBM for several reasons -- faith in the big blue letters and that IBM would take a major segment of the personal computing marketplace, provide good communications capability with the home office, and was not likely to be in and out of the business. But we did choose PC's and we had some concerns. There were an awful lot of vendors out there with micros. We wanted a vendor that was going to stay in the business as long as we were going to stay in our business. We wanted a vendor that had a range of machines because we were looking at a movement or migration of programs up from a micro to a mini to a mainframe. That would be the best of all possible worlds for development to moving programs up and down. Regrettably, that has not happened yet. And the other reason why we moved to IBM and moved to micros was that we felt that technological advances were going to happen a lot faster in micro technology. Micros were going to be able to do more, faster than we were going to see advances in communications technology. In other words, if we wanted to save money, we would be better going with a machine that was going to be able to do more real fast than expecting the cost of communications over lines to go down. The advances have, in fact, come very dramatically in micro capabilities and not at all in communications as far as I can see because we still use some time-sharing applications. Our unit prices for time-sharing have not gone down.

The other area we looked at was the development of software. I've already alluded to the costs of developing software for a small user base. The problem was maintaining a group of people who could understand what the user of the software wanted and translate that to a programmer. And our conclusion was that we couldn't do that. So we decided that we would no longer maintain an in-house proposal system developed by our own programmers, running on our mainframe. As of today, we no longer have one. We went to a vendor and bought a system, which is what we are currently using for all of our illustrations with two exceptions. The reason we went to the vendor is primarily cost containment. The vendor had a much wider user base to spread development costs over, the vendor also was willing to go on the risk with us for development. The vendor is prepared to do a lot of the development for you because their forecast is they're going to make it back in selling to other users. And I found that very comforting for two reasons: one, I kept my cost down; and secondly, it gave me at least some feedback that somebody looking at the marketplace other than us was agreeing that that was a reasonable development.

So as of today, we have moved from a time-sharing system using a home office mainframe, communicating at that point some at 300, some at 1200 baud, to a word processor to PC's, XT's, and I think as of last week, one AT in the field. Also in communications mode, our vendor produces a microsystem that we think is about as advanced as you can buy.

Technologically, it is not the most elegant programming. It is, however, in our opinion, the best advanced illustration system we can find, and it has two advantages. First, the system is operating on micros. There are six program discs. So it has a lot of capability of illustrations and a lot of customization capability, which is both an advantage and a problem. And I think we have somewhere between 13 and 18 discs of rates, so a lot of our offices who do high volume move very quickly to first the XT and then the AT to use the hard disc storage for the programs and their frequently used rate files to give them operating ease. We can do almost anything we want to do on that particular program. We have one more piece we will add this fall, and that is the ability to change column headings, and customize the words within the program. The person using the illustration doesn't have to run the illustration, move it to disc, and then word process it to make changes.

In telling you how we got to where we are now, I've really given you most of the reasons why we made the decision. The primary reason was cost containment. We estimated we would break even, or be slightly ahead on cost at the end of one year. It also had the added advantage of putting the control on productivity in the agency. The manager would make decisions about illustrations based on what the agent needed to sell business rather than on what it was going to cost him to get the illustration since each illustration run on time-sharing produced a unit cost. Our forecast at the time we made the microcomputer illustration about a year ago, looked like we would be in money or saving money at one year of operation. At this point, it looks like we might just make it.

I have also made some references to why we chose an outside software vendor. Again, primarily to spread the development costs over a much wider user base, but also to have a supplier who would go on the risk with us for developing complex illustrations. For example, developing a deferred compensation illustration program that would provide individual life illustrations, allow flexibility in assumptions and composite cash flow illustrations, was very expensive. If a time-sharing supplier was willing to produce it, we would much rather rent it than develop it.

We also wanted a vendor who had a backup system. At the time we made our first commitment to micro illustrations, we were really a little worried about what might happen so we also looked for a vendor who had a time-sharing system that allowed us a fall back in the event anything went wrong with a branch system. We found that the chief value of the time-sharing system was not forming a backup to the micro system, but rather to allow us to change interest rates on interest-sensitive products rapidly, introduce new products quickly and all across the country and make changes like dividend scale quickly. With the time-sharing system, we had a system we could set up a product on, test it on, and release it to all of our field at once without having to go through the process of creating it for a micro, duplicating discs and mailing them.

Where do I think the future is going? In terms of micros, my forecast is much wider use. Currently, it's our experience that agents are buying the things like crazy. We have made an offer to our agents for, in effect, a one-year, interest-free loan to purchase their own IBM PC's with the loan repayable over 12 months. We have not gone any further than that and that seems to be perfectly acceptable. We believe that the technological advances we hoped for are occurring. The development of new chips for the micros are producing more speed, better quality output and major improvements in memory and storage capabilities to allow a much broader range of complex illustrations to be produced on the machines quickly and easily. In the same vein, the developments of communications technology have not begun to occur so use of time-sharing systems has in fact not shown any decrease in communications expense.

We also see a major change in distribution of products. This started with universal life because it was impossible to produce it on paper to illustrate all of the capabilities of a universal life product. So to distribute universal life, you're forced by the competition and by the constraints of the product to distribute it on discs. I believe that this will continue to occur and that the major method of distributing new products will be single discs or one- or two-disc systems that will allow the agent to plug the product into his machine, manipulate it and make decisions on whether he likes it or doesn't like it based on the system, rather than reading brochures. In fact, we are currently negotiating with our time-sharing vendor for constrained single-disc versions of the system to allow us to distribute products widely to any producer who has a PC. We believe, for example that hand-helds will replace rate books as a primary method of giving rate data to agents and brokers and that proliferation of single-application discs will continue. We are now working on the second version of our universal life disc and working on an annuity disc.

Having painted this rather rosy future, what are the problems I see? Well, one major problem is control, particularly updates or changes. How rapidly can we change universal life interest rates in a down market? How rapidly can we change annuity rate bases? We have a major concern with the extreme decentralization of illustration preparation, we could be in a position of having our products being illustrated on bases which the company is not prepared to support. That's one major issue I think that anybody looking at micro applications must really consider. We don't have an answer for it, but are leaning to putting the change capability, if you like the disc that will allow changes to be made, in the hands of our managers and giving them the accountability of ensuring that when rate changes occur, everybody who has a disc gets an updated one immediately so we don't have bad proposals floating around.

Another issue is to ensure that our illustrations conform to the NASD or NAIC requirements for solicitation. State commissioners have not been particularly active in policing illustrations, not to the extent that the NASD has, but we feel this is a potential problem that will have to be addressed. It boils down to the manager in the field being accountable for the illustrations produced in his or her operation, knowing the seriousness of changes and policing them.

One problem we thought we would have that did not occur was in agents creating their own products. We felt that once they had a computer with real data entry processing capacity, they could, if you like, invent their own dividend scale. That has not been occurring, so it is a worry that we had and we don't think we have to really concern ourselves with at this point. Misleading illustrations with changes in column heading or word processing out footnotes that we feel are important is an issue, but at this time, we don't know how big a one it is. I should point out that all of our illustrations are reviewed by our own actuaries to ensure that they feel they conform to the ethical standards of the Society in how products are presented as well.

There are some technological problems that we think will be resolved quickly, but that are currently problems when you deal in a micro environment. One is downloading of changes. At this point, we have really not solved how to change our agency illustration discs electronically. We are still working in an environment where we duplicate them and mail them out. Although I think technology will improve to a point that will allow us to do that effectively a year from now.

We also are looking at micro-mainframe communications extensively. One of our major concerns is conservation and particularly the illustration of inforce products. Currently, most of the values for our inforce products are resident on our home office mainframe. We are trying now to find the answers to accessing that information effectively and moving it down to a branch micro with the attendant rate file so that an inforce illustration, that we hope will keep our product in force if it comes under attack, can be prepared quickly and effectively for the producer. At this point, we feel that the technology exists. Our major problem is we're not sure whether the cost is manageable, but we think so and that is a development we will be working on in 1985.

MR. MICHAEL AKERS: Up to this point, you've heard mostly about today's microcomputer application. My role on the panel is to share with you some examples of one of the newest directions in which computers, both micros and others, are moving and my topic is Microcomputer Communications.

Even in the compressed time frame of micros, it's only recently that communications has become a topic of serious interest to the typical micro user. Today, I'll be talking about some of the reasons why you should be interested in this rapidly developing technology. I'm not going to say that you should rush back to your office and order communications equipment for your machine, but there are many potential uses for inter-computer communications of which you should be aware.

Also, I don't intend to recommend specific networks or communication packages. There are too many options on the market and the package that you eventually choose must reflect your company's specific situation. What I'd like to talk about are some general topics on communications -- the evolution of microcomputer usage that has led us to communication applications, the environments for computer communications (microcomputer networks are not the only option that we have), applications for

inter-computer communications, and probably most importantly, the data processing and management questions that must be addressed before implementing an inter-computer network.

Originally, micros were brought into the work place and used as little more than super calculators. As the microprocessors are becoming increasingly powerful, the microcomputers have in turn become more effective in fulfilling this role for their user. Today, we heard from our other panelists a number of interesting actuarial and nonactuarial applications currently in use. We can only look forward to things continuing in the future whether it's accomplished with programs that are written in APL, Basic, or one of the other languages or through an off-the-shelf spreadsheet program.

The second major work place application for micros is in word processing. The evolution of the word processing packages has also been quite rapid. There are packages out now that have the added attraction of emulating the functionality of one of the commonly used dedicated word processing systems. More importantly today, the programs are powerful enough and easy enough to use that more and more professionals are finding it effective to learn how to use at least one of these available systems.

The third major work place application was the development of data base systems for microcomputers. Of the business applications that have been brought out to date, this is probably the one that has sparked the least interest amoung actuaries, but I think this is a tool that is looking for an application. I'm sure that applications are certain to emerge over the next couple of years.

There are many other types of programs that have been introduced over the past two or three years, but the ones that I listed are the major forces in the proliferation of micros in the work place. All of these are useful in isolated computers, but there are insufficiencies that have arisen as departments and companies install additional microcomputers. As more users acquire computers, we find that there is duplication of the costly peripheral equipment (hard disc drives, printers, plotters, and so on) that may not be fully utilized by any one user.

Equipment is not the only area where expensive duplication can occur. The duplication of data entry can be an even greater cost in the long run. Several users may be running applications on similar data, but each user loads his own data on his own microcomputer. The data that is commonly accessable must be transported on floppy disc from one location to another or the user must go to the machine that has the hard disc. A related problem for a micro user is the necessity of loading data on to your device that is already available on either a mainframe or a minicomputer.

One solution to these duplication problems is to develop communications capabilities for microcomputers, either networks of micros or communications with larger computers. As one of the companies working in an environment that has micros, minis, and mainframe, we found it necessary to develop these kinds of communications ability. The primary combinations that we focus on are first, the micro to the minicomputer communications;

second, micro to mainframe; and finally, micro to micro, or networks. To this point, I have spoken in some generalities about communications among computers. I'd like to move on to some specific applications that have been implemented which require the ability to move data between computers. The examples that I have described are obviously not the full range of what can be done, but they might set you to thinking about what you can do inside your own company.

The computer connection that we have found the most fruitful to date has been in our micro to minicomputer link. There are four areas where we're currently making use of this. First is terminal emulation with a microcomputer, second is uploading the data files from the micro to the minicomputer, third is downloading of data files from the mini to the micro, and the fourth is the inter-office and intra-office communications.

Terminal emulation is a pretty straightforward application with some obvious advantages. Usually none of the computing powers of the micro are called into use, but you do enjoy some increased flexibility of adding terminal capacity to your system without having to buy more single use units. This is not what I would call real communications between computers, though.

The ability to upload files from the micro to the minicomputer or to move them down (download files from the mini to the micro) is probably the key functionality for the micro to mini environment. An important point that I'd like to interject here is that although there are many communications packages out on the market, there is a very common limitation on many of those. Not all packages have protocols that verify data transmission between the sending unit and the receiving unit. We found that it was absolutely necessary to have those protocols in place to maintain data integrity during file uploads. The alternative is to keep sending until you get it right. Before you decide on a package for your own company, you'll have to consider whether or not you need these types of checking protocols.

There are a number of ways we've made use of the upload and download abilities being used by the actuaries in our firm. One of the first ones was the profit study system which is available to many computer users through a time-sharing network. Some of our pricing actuaries have developed programs on their microcomputers for calculating special input data or for manipulation of the output data from those profit studies. The communications program allows them to do this special purpose calculation on their micros and still be able to use the time-sharing system for the standard number crunching type of operations.

Our pension actuaries have access to a valuation system on the time-sharing network. We developed a macro which was a set of instructions, in Lotus 1-2-3, to serve as a preprocessor for data to be used in the valuation system. The macro prompts the user for the various data elements and reformats them into the proper structure for the main programs and then you simply upload the data to the minicomputer.

Another area of use is APL programming. Many of our actuarial users have the APL on their microcomputers and also use APL on the minicomputer. We've had some success since there is a certain degree of compatability

between the two versions of the APL. People can develop programs on the micro and then upload the data files and work spaces and run the full job on the minicomputer. That gives them access to the number crunching power of the mini once again. We have had several cases where the uploading and downloading capabilities were used to transfer data developed in one of our offices to another office that needed access to the results of the same input data.

Numerical data files aren't the only types that we find necessary to move through this micro to mini link. Often times, our users will prepare a document on a microcomputer word processing program with the intention of uploading that task to the minicomputer. The mini serves as the controller for our electronic mail network, so we can direct a document file to other micros or even to our own word processing equipment to obtain letter quality documents.

The microcomputer to mainframe communications are a different environment from the micro to mini, but the communications modes (terminal emulation, uploading of files, downloading of files) are essentially the same as for the micro to mini links. The types of files maintained on the mainframe open up a lot of new possibilities. On the mainframes, you start moving into the area of policy master files, accounting files, and large corporate data files. The file architecture is different between the mainframes and the minis, which makes for an added problem to be solved by the people who are going to build and maintain your network. However, once that problem is solved, the potential applications are quite interesting.

One application that I've been told about is in a company that has had some serious delays in the issuing and underwriting processing. To make matters worse, they were preparing to introduce a new annuity product and were concerned that their sales campaign was going to be derailed by this uncooperative issue system. I'm sure a lot of you encountered programs that just will not take on any new functions without some massive modifications. This company got around that problem by programming a microcomputer to handle the issue and underwriting functions for the new product. Once the contract is issued, they upload the necessary data to the policy master file in the mainframe.

The sharing of mainframe data base information is an application that has considerable promise for the future. In the trade papers, we see an almost continual stream of microcomputer software for accessing mainframe data bases being announced almost weekly. In conjunction with these systems, the high-level data base query languages are being modified to operate on a microcomputer. In a sophisticated shop in the not so distant future, users will choose either to work on their main data base in a terminal emulation mode, or to download a piece of that data base to manipulate on the micro. They'll be using the same language in both cases. They could conceivably take the data, work on it, and subsequently return that information to the data base.

The examples that I have given are just an indication of possibilities of microcomputers linked up with larger computers. I think it's one of those areas where the more you exercise your options, the more options you will receive down the road.

The final inter-computer communications link that I did list earlier was networking of microcomputers. A network is a direct method of attacking the duplication problems that I spoke of earlier: the duplication of hardware and duplication of data entry. An ideal micro network would allow any user to sit down at any micro in the network and work on it as if it were his own machine. Hard disc capacity, high speed printers, and communication controllers would be shared by all the users.

For all its promise in advertising, we haven't found networks to be as useful as the linkages that we have to our minicomputer and to our mainframe. One of the major arguments for networks, sharing of these expensive peripherals, has been eroded by the drops in prices of hard disc and printers over the last couple of years. The cost of purchasing peripherals for these work stations has to be measured against the cost of purchasing a controller and the additional hardware for each micro that is going to be in the network. Also, the sharing of discs and printers is an obstruction to access that is one of the main advantages of microcomputers in the first place. We've tried a couple of networking packages and will continue to try others as they come out, but we have not identified any solid reasons for committing all of our micros to that type of network.

Before you lock yourself into any sort of network, I'd advise you to review the data processing technical expertise on your staff. Our assessment of the networks that we have encountered to this point is that maintenance requires either a knowledgeable micro technician in your company or a contract with a microcomputer consultant. In fact, any move out of the single user micro environment will force you into a data processing role that most actuaries aren't equipped to handle and are just not interested in dealing with.

One reason that we're not as interested in networking of micros as others might be is that we're installing a different type of communications device which is called a "smart switch". In effect, the switch gives us the ability to communicate between any of the devices that are hooked up to the network. This includes our mainframe, our minicomputer, the microcomputers and even our word processing equipment. Local users will configure their micros to fit their own processing needs, but they'll be able to transmit data to and receive data from any other user that has access to our time-sharing network. This communication works at the inter-office level as well as the intra-office level.

The ability to communicate among computers raises a number of questions that management must consider. The first is simply whether or not there is a sufficient reason for instituting the types of communications that we're talking about. Second, will you only link certain standard micros or will you try to devise a comprehensive link for all devices that are in the company. This decision is simplified if you already have a company standard for microcomputers. Do you have the technical expertise that's necessary to keep the microcomputer network or something more sophisticated operating smoothly. Actuarial departments are seldom a depository for that kind of knowledge and we found even many systems and programming departments don't have the technical skills of networking . Who will control, and how will they control the accessing and modification of data files by users. It's one matter to allow access to information on a data

base, it's something else again to allow modification and reloading of that data. This may be one of the most crucial questions to consider. Another consideration is what control can and will be exercised over data download. If information is stored on a floppy diskette for later use, data can easily be lost or possibly sensitive information can wind up in an area where it's not intended. Lastly, will there be a coordinated and concerted effort to develop applications or will it be strictly a user-driven environment.

For anyone who has an interest in the micro-mainframe link, I recommend an article that was entitled "Micro-Mainframe Navigation" that was in the October 3 edition of Computer World on Communications magazine. The article is short and it's not too technical and does give an overview of some of the micro-mainframe packages that are available today. Even more valuable than those overviews was a discussion of some of the issues that you must face before you can seriously consider linking microcomputers and mainframes.

In closing, I'll say that I find a potential use of inter-computer communications to be exciting and something that is well worth keeping pace with. Even now, we've decided in our company that the computer literate individual must know how to operate the three basic microcomputer tools: a spreadsheet program, a word processing program, and finally, our communications package and I think somewhere down the road, you should all be considering the same things.

MR. JIM DOHERTY: Do you have any communications between outside data bases and your inside micro or mainframes? You addressed security on data and data between micros. Would you talk about security on software purchased outside, such as Lotus 1-2-3?

MR. AKERS: We are, among other things, a software house so we had to identify very strongly with the concerns of people with Lotus and its company policy that you do not make unauthorized copies of copyrighted programs. I know some companies where it's grounds for dismissal if you make an unauthorized copy. I don't have a lot of experience in dealing with mainframe data bases and I don't think I can really comment on that, especially between Canada and the U.S.

I think security is one of the big questions that could be particularly difficult if you get into networking, because you can't put something like Lotus on a network and have several people using it. But as networks become more common, they're going to have to work out a group licensing arrangement.

MR. DOHERTY: Mr. MacMillan, you described your company as having a large array of programs that allow agents in your branches to produce illustrations on sight. I got the impression that for every product you had a large illustration service. You then expressed concern about the large number of paper ratebooks that you were printing. I wondered about the compatability of having these arrays available on computer and still printing paper ratebooks. The more intriguing comment was you suspected that paper ratebooks were on their way out and that you would have hand-held computers doing this rate file. I wonder if you would expand on those remarks?

MR. MACMILLAN: First, let me deal with the microcomputer and the size of the program. The program is a very complex one that produces a wide variety of illustrations. There are six discs which are basically program discs that allow a lot of customization. You can move columns around and display them in different ways. There are function columns which allow you to express columns as a percentage of each other and so on, add, subtract, etc., which accounts for the complexity of the program. The product complexity results from trying to put all of the variations of all of our products, with the exception of annuities, in the field. Our basic whole life product is male, female, unisex, smoker, and nonsmoker, with a dividend scale that varies with the policy loan interest rate. So, you're putting a lot of data on floppy discs. My comment on the paper ratebook is that I'm responsible for maintaining it and it's getting expensive and it's less and less used. Agents are using illustrations. They are not using rates. The hand-held computer with chips will be the ratebook.

MR. WILLIAM LOUCKS: Now that you have your ratebook on disc, what about your application?

MR. MACMILLAN: We haven't started to think about that. We have toyed with not bothering to get an application until we actually have issued the policy. In effect, saying to the agent, as long as you have a signed application in the field, that meets the legal requirements. You can input the policy underwriting data that will generate a decision and generate a policy which will be assembled in the field. Please send the paper later. We're not quite ready to do that and so we're still handling a fair amount of paper. I think, in fact, the issue will not be the application, the issue will actually be seeing the underwriting documentation before we will agree. I think that's what will keep us in the paper chain for much longer. I would suggest with non-underwritten products, annuities in particular, that we will move much more rapidly to a piece of paper coming later to satisfy legal requirements rather than as a requisite to getting a policy issued.

MR. JOHN KIRKMAN: I think I can supplement your previous answer on the ratebooks. We've gone through almost exactly what you described. The two differences are:

1. We still maintained our in-house personalized sales illustration system. The key reason for that was we thought we had to provide service to the so-called average agent.

2. Several years ago, we stopped printing ratebooks. I just thought I should let you know that it is possible to do that.