RECORD, Volume 23, No. 3*

Washington Annual Meeting
October 26–29, 1997

Session 104IF
Actuarial Careers in Software

Track: Actuary of the Future
Key words: Product Development, Computer Systems

Moderator: DOUGLAS S. VAN DAM
Panelists: DAVID N. KARO
JOSIAH LYNCH
MARK S. MAHANY
Recorder: DOUGLAS S. VAN DAM

Summary: Software development remains a growing business in which the U.S. maintains a commanding lead, dwarfing all other countries combined. The following topics are explored in this session:

- *what makes a career in software unique?*
- *the actuarial skills in the industry,*
- *other skills needed to succeed, and*
- *who should consider this option?*

Mr. Douglas S. Van Dam: We are talking about actuarial careers in software. There's a big intersection between the people that work with computers and those that are in the room. David Karo is with Ecta Corporation, which does illustration software. Josiah Lynch is with Lynch Bell Systems and he works on pension systems. Price Waterhouse does valuation systems. The membership in the computer science section has been fairly level—between 2,100 and 2,300 members. There has also been a steady increase in software development

Mr. Mark S. Mahany: I'm the managing director of the Insurance Software Division of Price Waterhouse. I started as an actuarial student at John Hancock back in 1980. I spent three years with John Hancock and went to work at Transamerica as an actuary in the research unit. I spent four years at Transamerica and joined Chalke Incorporated. I got my FSA and started marketing and selling PTS software

*Copyright © 1998, Society of Actuaries

at a time when we had probably 25 companies that sell. I spent six-and-a-half years with Chalke Incorporated and in March 1994 I joined Price Waterhouse to start off their Insurance Software Division, which was the start of valuation software in the individual insurance marketplace. We also do some property/casualty software as well.

I'm going to talk about the sales and marketing component of our presentation. In the last 14 years, I've really been functioning as a salesperson. I spent six and a half years at Chalke doing asset/liability management (ALM). I started with the Regulation 126 marketplace and it grew into what we now know as cash-flow testing and asset adequacy analysis. The valuation marketplace is a relatively new marketplace, although a couple of companies have been in it for a long time. From the individual insurance component, what's happened in the mid-to-late 1990s is sort of what happened with the asset/liability marketplace in the late 1980s. The new regulations that have come about from the states, have sort of rendered software development to be a commercial venture these days. Companies have found that an incredible amount of time and revenue are spent to do it in-house. For the most part, the administration companies that used to do the valuation component have sort of washed their hands of the valuation piece, and so the companies either have to continue to develop their own legacy system or go out into the marketplace to buy new software. We sort of hit on a nice marketplace, at least at Price Waterhouse. If you're a salesman, you need a market and there's a real viable marketplace. There's a couple of other entrepreneurial opportunities that we see that exist. The Internet is coming of age and I think people will learn how to take advantage of the Internet, whether it's through sales or developing technology that companies can take advantage of. Dave will talk about sales in that marketplace.

We really employ four types of actuaries in our division. The first one is a technical actuary. We have people that sit down and actually write out specifications from which we're going to build software. Whether we're doing a bug fix or a customization for a particular client, a new regulation, or just supporting some new product feature, we have an actuary that sits down and actually writes out the formula that's going to be incorporated into the code.

The second piece that I missed, because we don't have any actuaries doing this function at our company, is taking the specs to a developer and having a developer code it out. The third type of support manager is a core manager. These are actuaries that are functioning as people that train on software and have a fair degree of knowledge about what the software does and how it goes about doing it. You can just teach clients how to use the software and continue to manage the clients once they get up and running. So we're a little bit unique. When you bring the package in-house, you get some level of training and then you hand it off to a support desk. With our support function, the people that train and the people that are responsible for implementation of our products, at a particular company, actually stay with the client. It's nice because they recognize a familiar face. They have some degree of technical knowledge about the software and so I would say these people aren't really functioning as actuaries, but they certainly know enough to be dangerous and know something about the software and the hardware. They are really jacks of all trades.

Then the final type of actuaries are the sales reps. I would say I haven't been functioning as an actuary for about 13 years, almost since I got my FSA, but I think a sales function would be incredibly useful because I think people give you the benefit of the doubt because you have three initials after your name. They get to know me, and they don't really give me much benefit of the doubt after that.

For some reason, over the years, there has been a negative connotation attached to sales. I like being a salesperson and I'm not shy about saying that I am. You might not realize that everyday you are a salesperson. If you're trying to make a case for something that you want to do in your job, you're going to try to make the case to do things the way you want. Your ability to be a salesperson is going to determine whether or not you're successful in your career. So even if you're not in a pure sales position, I really believe that you're measured by your ability to convince people of your ideas and your ability to implement them.

When Doug asked me to give this speech, he asked me to not only talk about what I do, but what I felt determines being a success in an organization. I was always fairly certain about my relative strengths. I was always able to find out what was important to me and turn that into a career path. When I was going through the exam process, I was at Transamerica. We used to have a study group because there were so many actuaries taking the exams at once. I sat in a room with about seven or eight people, and we quizzed each other. I guess my strength as an actuary is that ultimately I would always guess, but the problem was I was always the fifth or six one to get it in that room. I think that when you expand the focus, you always feel like you are in the bottom 25% of getting problems, being a technical actuary or being in that role. I always had a good career being an actuary. I passed the exams and did my work, but they couldn't really find a place for me to stand out. The sales opportunity was something that was given to me and I took advantage of it. At Transamerica, I had opted to be an actuarial director. I was running the student program, even though I wasn't an FSA, and dealing with upper management and negotiating and doing reinsurance treaties on the ceded side. That type of work appealed to me because it was an opportunity to get to know other people within the industry.

In my junior year of high school, I loved sports. I went to my math teacher and said, “I’m happy as an athlete, but I don’t think I can make a career out of it. What can I do?” She told me to become an actuary.

As I mentioned, I was with Transamerica when Shane Chalke approached me about selling software. It was at a time Shane had developed PTS, and we probably had about 25 clients. It was only a liability-based system. He asked me to join him at a time when he was adding the asset model, at a time when Regulation 126 was just coming out, and when New York was asking clients to run ten scenarios. People were starting to look at the commercial marketplace. The opportunity that he offered me was to sell, but to really run my own shop. Even though I was an employee, I was compensated on a commission basis, the same way the prior salesman was. So I took a salary draw, but I really didn’t get a cent on the salary draw until I earned the commission. It really made you run your own business. I felt like I had some money in the bank, and could afford to take a chance with the opportunity. The nice thing is that I felt like it would give me a lot of exposure to other companies, so that I could see different clients and different environments in the companies. Even if it didn’t work out, it would expose me to those companies that I was impressed with. When you go out and do presentations for about 40 companies a year, you see some companies and you wonder what they’re still doing on the face of the earth. You can tell whether they are going to be successful by the quality of people. I don’t really determine my priorities; my clients determine my priorities. Every morning when I come into work, I probably have about 40 things on my list of things to do and my clients prioritize those for me. Whoever is going to make me the most money or make the company the most money goes to the top of the list. I’m able to cross about ten things off the list after another 20 things get added to the list.

When I joined Price Waterhouse, my position was that of a managing director. So even though I don’t write a line of code, I’m responsible for not only the sales of our line of software, but also the management of the division. It has really provided me with a conflicting role in the past. I just used to sell, sell, sell, and I couldn’t understand why the development division couldn’t keep up with me. My biggest enemy as a salesperson is the Director of the Division who had to get the work done to deliver to the client and vice versa. So the managing director in me wants the salesmen to go out and take a vacation for a couple months, while the sales guy in me is really looking to try to sell as much as possible.

As I said the market sets the priority. If you like managing your life based on what other people tell you to do, you’ll like sales because everybody has an opinion about what you should do. You should have a little bit of entrepreneur in you. If you’re willing to take a risk, even though actuaries aren’t well known for their ability to take risks, it's a great opportunity. The best advice I was ever given as a salesman was to shut up. The ability to listen is very important. Many times, when you're dealing with clients, they'll tell you what you need to do to make the sale. You just have to listen. They'll tell you what their priorities are and what they want the software to do. That makes your job very easy. I think you need the ability to integrate information. You must try to turn what you learn from one client into what other clients may want. I also think you have to be tenacious enough to never give up.

As far as success goes, I think that the most important criteria that we have in this particular world is the idea that there is coordination between marketing and sales. Even though I'm responsible for all of the puzzle pieces, if any one piece of the puzzle falls apart, the whole puzzle falls apart. If you get new clients and you don't take care of your existing clients, it's not going to be very long before your perspective clients find out that your existing clients aren't too happy. The mood within our shop is that our existing clients sort of determine our priorities and we listen, especially when people are buying $100,000 or $200,000 worth of software. If you're not taking care of your existing clients, and people will find this out, you're not going to have any existing clients.

Finally, I'd like to discuss the tools to achieve the communication. We have tracking sheets. You need to be able to track what you're doing for every sales goal. At Price Waterhouse, we put every request or every initiative that we want to do with the software into the same form. Whether it's a bug fix or clients screaming for customization or a new regulation that's becoming effective December 31, 1997, we continually validate all of those priorities with one clear picture, so that we don't have the left hand doing something the right hand doesn't know about.

My final sales story actually happened when I was doing a presentation at a company I used to work for. We had been trying to sell PTS and they were depending on me. They had the room set up so that there were 40 actuaries and systems people in a complete circle around me. I was doing a presentation and sat there for about I'd say three-and-a-half hours, first doing my presentation and then getting grilled. They asked me, "Can you support this?" I said, "Well no, I can't support that now, but we plan on doing that in three months." I went on to discuss something else and I had to say, "No we're not doing that, but we plan on doing it in six months or nine months." The company didn't buy software anytime soon and they didn't buy anybody else's software, but after year-end, the same group brought me in again. They said, "Let's see what you have now." I went through the presentation and there were two people that sat there with the entire list of all of the things that I said we were going to do. I ended up closing the deal within a month because everything that I had promised that we were doing to do, whether it was to be done in three months, six months, or nine months, we followed through and completed. This was a lucky break because we all know priorities change from one day to the next, especially over a year. The bottom line is, if you're honest with people and you don't sell "vaporware," or the new term is *software futures*, you'll be successful. The sales world can be fun.

Mr. Josiah Lynch: I'm going to talk about how we got to where we are now and what patterns we've seen evolving over the decades. As I was saying to Doug earlier, I'm not sure I'm at the right session because I'm a pension actuary. I feel very much out of place in the group that's wandering through these halls. Everything I say will be from a point of view of a pension person, who specializes in software for pension people.

The first generation of computers dates from the late 1940s to the mid-to-late 1950s. They were characterized by vacuum tubes and machine language programming. It was a very primitive beginning and actuaries played almost no role in this generation of computers.

The second generation of computers got more interesting. The second generation was characterized by solid state equipment, transistors, the invention of languages, and assembly language. COBOL and Fortran were the first two user languages; COBOL was for business, and Fortran was for mathematicians. Underneath it all was the six-bit characters which meant that, with a single cluster, you could have up to 64 possible characters, which means you had an upper case, but no lower case, numbers and a few other characters.

Now the second generation computers ultimately became very powerful and advanced. What was probably the most advanced was the Control Data cyber super computer series. They had the Cybernet and multiprocessing time sharing and quite a few actuaries used those computers to great advantage. The third generation embodied within the IBM 360 and the 370 and the RCA Spectra series, while it lasted, had new architecture with an eight-bit byte. The eight-bit byte still carries on to this day, allowing 256 possible characters, which is reasonably rich. IBM instituted the most cryptic of all possible languages called Operating System Job Control Language (OSJCL) and that was a career maker in some cases and a career breaker in other cases.

The mainframe environment was very forbidding. These computers had limited access. They always had their own room, usually on the floor. In a rigidly controlled environment, the temperature and the humidity and the air quality were carefully monitored by the hour. They were always carefully enclosed and, as I said, admission was restricted. There were fairly frequent references in literature to the priesthood that guarded those computers and who had a level of culture, roughly seen now at the CIA or the Secret Service. From the actuary's point of view, the actuary's role was pretty much restricted to advising the software designers. There weren't many actuaries who did "hands on" programming.

Let's discuss the role of the actuary in developing actuarial software, compared to the role of the programmers and the people who are involved in technology. Technology, at this point, represents a small proportion of the total effort because it is highly computer-related and it didn't vary much from project to project or from system design to system design. The people involved in technology simply monitored the operating system and made sure that the computer kept working. The programmers did the bulk of the work on mainframe software, and the actuaries provided the objectives and worked with the programmers to make sure the results were right.

In my own career, I've had the good fortune of combining these two roles—the actuarial role and the programming role so that nobody ever programmed something that I worked on. I did all the programming myself.

In the early to mid-1970s, the minicomputers came on the scene. The minicomputer had the charm of being a more personal kind of an entity. A small firm could afford one and it didn't quite require the scientifically monitored room that the mainframes needed. Each minicomputer was frequently served by a modest number of terminals. These machines cost less, and their use did lower the cost per calculation. In the mainframe era, there was a rule of thumb: the more expensive the computer, the cheaper each calculation was. The minicomputer, for the first time, turned that rule of thumb on its head. The minicomputer also had a more flexible approach. In as much as the minicomputer was usually serving a single company, then the operating system could be much looser in its accounting than it would be, if it were a main computer serving many companies or many, many huge divisions within giant companies that were constructed as if they were many companies within a single firm. So they were much more personal in their appearance, in their use, and in the way people thought about them and used them.

So in the same subjective evaluation that was used before, there was a much greater involvement of the actuary in the use of a minicomputer than was the case with the mainframe. There was more technology required because the minicomputer lent itself to more fussing than was possible with the mainframe. So some purely sophisticated routines were put together for the minicomputer within a single firm.

Now we come to the beginning of the PC era. I'm going to bypass the non-DOS machines, the Apple, and the old CPM machines. The DOS PC first came out in

1981 and has been getting better and better over time. It had the lowest cost per calculation ever seen in history. It had the advantage of being a personal computer. It was right there for the user to use. If he didn't use it, nobody did. It wasn't working when he wasn't using it on the assumption that one person had one computer. He could actually lug it in his car from office to home if he wanted. It provided a liberating effect. The reason why I'm concentrating on the DOS PC is because IBM, recognizing that they couldn't possibly capture that market by themselves, opened up the architecture and made it freely available to anyone that wanted to create software for it. As a result, there was an explosion of third party software. Geniuses in garages were working around the clock to develop software they'd sell you for $100 or $200 to make your work easier. So the productivity became fantastic on each one. It was literally a personal computer.

This sort of unleashed the actuary's power on that computer. He could do programming himself. It came with a free version of Basic. He could, for a modest amount of money, get a spreadsheet program or his employer could provide it for him. So while many of the actuaries still had other people writing programs for them, more actuaries were doing hands on work on their own computers. With spreadsheets and database programs, it was not uncommon to have a guru or two, but it is here again, more like a personal guru, rather than a fixed asset. So that was the DOS PC, which was a stand-alone machine.

Then the local area network came along. The actuary still had an intimate involvement with his or her PC. There was still programming being done, but on the network, the role of the technologist was enlarged because the network required a tremendous amount of attention, tweaking, educating, etc. We now have Windows 95 or Windows NT PCs. These machines, in order to support Windows, are much more powerful with a greater capacity than ever before. You have a less verbal and more graphic interface with icon access and so forth. It's getting very friendly, while getting more difficult to fathom. We have dynamic linking, fourth generation language, and visual object programming; it is immensely powerful but, in some ways, it's a step backwards for the actuaries. So what we're seeing now is that the proportion of the actuaries' task to the programmers' task is about what it was, but the role of the technologist has grown by leaps and bounds. The systems person has a much more powerful role than he had, even in the mainframe days or the minicomputer days. In my experience, it has been not unheard of for the actuary to do this function and this function, it's much rarer for them to do this function. But here again, I'm talking from the point of view of pension actuaries.

David N. Karo: I've spent almost the last ten years working as an actuary, dealing with illustration systems and computer software. I started out as an actuarial trainee right out of college at a small life insurance company. I had no programming background when I started to program the company's traditional life insurance illustration system on an IBM 5110. From there, I moved on to the product development area for a medium-sized life insurance company. I was responsible for pricing its universal life (UL) portfolio. I also was involved in programming its in-house APL (A Programming Language) illustration system, as well as making state filings and dealing with compliance issues. When we purchased the third party illustration system, I was the main actuarial contact. That proved to be a fortuitous relationship because a few years later, another company was coming in and was going to take over the insurance company where I was working, so I left to go work for the software vendor. I have now been working there for a little more than five years. Our main products are life insurance and annuity illustration systems, along with Internet and client-server solutions.

All our new systems are Windows-based, however, we still do a good bit of DOS work in maintaining our Legacy Systems. I mainly deal with the calculation engine portion of the universal life and variable life projects and I generally see the projects that I work on through from start to finish. I start off with a sales presentation, a product demo. From there we move on to a face-to-face meeting where we fire out all the specifics of the project in great detail. At that point my actuarial background is a huge help. From that information, we assemble a comprehensive technical manual that the client must sign off on before the programming actually begins. Once they've signed off, I begin programming the calculation engine and testing home office values against our own values. The final step is assembling the input and output modules in hooking up the calculation engine to them and retesting everything against the home office values.

Now the remarks that follow are based mainly on my personal experience and may vary between firms in the software industry. However, there're many factors that make a career in software unique. One of the most noticeable differences is the more casual work atmosphere. This might show up in a more relaxed dress code, which can sometimes be a hidden cost of employment for some people because their casual clothes might not be suitable for polite society. There also might be a lack of formal rules and procedures. That can be very desirable for those people that are fed up with office politics. However, sometimes the lack of rules can be a source of frustration. There tends to be fewer levels of management, at least in smaller software firms. Far less time is spent in meetings, allowing for uninterrupted working conditions and increased productivity. Plus there tends to be more dependence on teamwork within a software firm than within an insurance company. That's due to the large scope of the projects and the tighter deadlines.

There are so many different facets of the projects that it's impossible for one person to be an expert on all the pieces. We usually break down the project into a lot of little pieces and divide them between team members. Software firms tend to believe in a more equitable sharing of wealth than insurance companies. This philosophy leads to better employee benefits. Some typically enhanced benefits might include better compensation because software firms have a lot more competition when recruiting and they entertain highly qualified individuals; there might be profit sharing plans or production bonuses, as well as more vacation time and fully paid medical benefits for your entire family.

The work environment in a software firm can differ drastically from that of an insurance company. Some of the differences could be more flexible work hours. I have several co-workers who come in before six o'clock in the morning, so they can be home in time to meet their kids when they get home from school. There may be loud music playing in the hallways or a liberal use of profanity. That profanity might be directed at co-workers, as well as the clients; it just depends. There is also a more permissive policy towards office decorating. Individuality seems to be encouraged. For instance, when I moved into my office, the entire back wall had hundreds of Pepsi cans glued to it from floor to ceiling. Since I don't drink Pepsi, one of the first things I did was take it down, but a lot of people at our company still kind of look back at those days fondly. In a software firm, there's a lot more likelihood of being involved with projects with an international focus because you're dealing with many clients instead of just your own employer. It's not unusual for many of the nondomestic firms or large multinational conglomerates to look for the best software that fits their needs, regardless of its company of origin. Because of this international focus there is greater potential for travel than within a life insurance company. However the chances of going to some exotic locale are probably no greater than what you see now. However, you might have a better shot at convincing your employer to send you to Hawaii next year for the meeting.

One of the greatest benefits of working for a high-tech firm is staying on the cutting edge of the latest technology. You're continually forced to update your knowledge of computer hardware, software, and programming languages. This process will broaden your skill set and make you more marketable. Actuarial skills needed in the software business are similar to the skills most of you already possess. They would include the basic knowledge of actuarial mathematics, general insurance knowledge, product knowledge on an industry-wide basis, as well as the knowledge of pertinent regulations. Your insurance background is more specialized than your computer background. This makes you more desirable because it is easier to pick up technical computer knowledge on the job than it is to pick up insurance knowledge.

Other skills needed to succeed in a career in software would include a basic knowledge of computer programming principles and an understanding of logic.

Learning the computer languages is something that can easily be picked up as you go. Organizational skills are important because you'll probably be juggling multiple tasks for multiple clients, all of which are likely to have an aggressive time frame. Communication skills are much more important when you're dealing with clients in another location, rather than your peers under the same roof. Documentation skills go hand in hand with communication skills. When dealing with clients, it's always important to be able to document any details that have changed on the fly. It's so important to cultivate the ability to work effectively with less supervision. You should expect to assume more responsibility for projects of a larger scope. You should be able to shift from a task orientation to a goal orientation. This may happen when a project is in jeopardy of not being finished on time. The overall goal of delivering by a certain date may supersede some of the individual tasks that make up the project. This caused some of the tasks to be left out for subsequent deliveries, so that the overall goal of delivery could be met.

People skills are always important. You need to be able to deal with a wide variety of people who work in different departments of your client companies. Each group may have a different background and perspective on the project. The different groups may not always communicate effectively between each other, so it's up to you to make sure that the communication goes smoothly.

Many of you might want to consider the idea of working for a software firm. Career ASAs or other actuaries that have stopped taking exams could benefit since promotions aren't tied to exam level, and there's less emphasis put on studying for exams. However if you decide to continue taking exams, don't necessarily expect that you'll be getting study time from your employer. People frustrated with working in a corporate setting should thrive under the looser constraints set up under a software provider. If you have a strong interest in the latest technology and programming, a software firm would definitely be right up your alley.

One thing to consider before jumping ship to your favorite software vendor is the existence of the dreaded cross-hiring agreement. Many vendors and their clients have these in place to prevent their best employees from leaving. You don't want to burn your bridges behind you if you go in case things don't work out and you want to return to your former employer, or in the event of an even touchier situation, such as when your former employer becomes a client. In that situation you're almost guaranteed to be asked to work on that project due to your familiarity with the corporate culture and the people. So you really want to maintain cordial relations with them.

Many things are special about my current position in the software field as opposed to other actuarial work. They include dealing with a broader range of clients from many different backgrounds. It's not unusual for me to be dealing with home office staff, consultants or even other software vendors all on the same project. Once I was in a meeting with 17 other people, and only two of them worked for the client. The rest were from six different consulting firms. As you can imagine, this can cause great communication problems. My work has allowed me to broaden my horizons and learn about different insurance products that are being sold abroad as well as the regulations apply. My employer's liberal Internet policy has allowed me to learn how to use the Internet as a business tool. I use it for keeping tabs on competitors, researching regulations, or even recruiting candidates to fill in an open position.

Strong client interaction has improved my interpersonal skills. Since actuaries are still relatively rare in nonactuarial software firms, you could assume a higher position, both inside the company and outside of it. Due to my background, I'm continuously sought out from my coworkers for advice on insurance issues and mathematical questions. Due to the smaller size of the software firm I work for, I'm able to assume a lot more responsibility for projects than I ever could in a larger insurance company. In a software firm, you're not going to be constrained by your title like you might be in a corporate situation. This is going to allow you to be judged based on the quality of your work. It may seem like a very unusual concept to some of you, but believe me, it works. With the increased responsibility comes increased control of your career path. Since there's so many different projects going on at one time, you may have more of an opportunity to volunteer on a specific project that's more in your line of interest.

Mr. Van Dam: Mark, you mentioned an entrepreneurial bent. Joe, you didn't mention it explicitly, but that's what you've been doing for many years. Do you see any new opportunities out there for the people in the audience who feel they have an entrepreneurial bent?

Mr. Lynch: Actually, I see the cost of entry going up all the time. When I entered, there were no competitors. So anything I did was original and hopefully of value. I was the first person to found a firm that specialized in pension valuation software. I was looked upon as a pariah in the profession at first because I didn't have a job. I was building my own company, but it has worked out for 33 years. I've been responsible for meeting the payroll I was on. But I have noticed that the cost of entry is going up for a new firm because the people who develop the software that is in commercial use have been working at it longer and longer and gaining more and more experience. The technology is getting stronger, so the programming standards have increased. The interface standards have increased. It's very difficult to break in as a new entrepreneur. I think you will see what you've seen in other professions. Software firms will grow from within, by acquiring people. They will grow by expanding their offerings, but as far as the entrepreneurship, or starting a brand new firm with a brand new product in an untried area, the chances in the actuarial specialties are becoming fewer and fewer. That's not to say it can't happen, but as I said, the cost of entry is higher than ever and it is going up.

Mr. Mahany: I guess I agree with Joe. My point would be that you need the ability to step back and look at what's important to companies. One clear thing that every company is trying to learn is how to get products developed in the market as quickly as possible. We need to get them through the regulatory approval stage, get sales illustrations, build Internet access and that type of thing. If someone is able to get a product from the gestation period to the initial idea to the market and in the agent's hands, there is going to be a market. If you look at ALM tools, at least on the individual side, from an actuarial perspective, those tools are really being used as regulatory vehicles. I'm a little bit removed from it because it has been three and a half years since I was in that marketplace. There're just not a lot of companies using the information that comes out of the PTS systems to manage their company on a going forward basis. They use that information for regulatory stuff and then they do other models, investment models, for example. So I think there are opportunities out there, whether in a stand-alone shop or getting someone within your company to evaluate what's important to the company and come up with a strategy.

Mr. Karo: One of the two areas where I've seen a lot of opportunity are the Internet, which is obviously the hot topic on everybody's mind right now. Many people are talking about selling insurance or annuities on the Internet. Session 103 is on that topic. So we're dealing with many clients that have a lot of interest in that. It hasn't necessarily translated into sales yet, but we have put some systems out there that are doing that. There's even one insurance company that has no home office per se. They're doing everything on the web. There is a great deal of interest in the Internet right now.

The other thing that's been a real boom for us has been the illustration regulations. Anybody with knowledge of that has some opportunities. It has kept us very busy and in Canada, they just passed some guidelines that are somewhat similar to the U.S. regulation. We anticipate seeing a lot of work coming out of that as well.

From the Floor: I've actually worked with software from all three of you guys. One thing that I noted is that all three packages that you've been involved with that I've used have been written in different languages and they have very different philosophies as far as providing the source code or letting you modify it or customize it. I wonder if you could address the question of programming languages and what you see as being the future, if there is a future?

Mr. Karo: For us it seems like C and C++ is the language of choice. We're somewhat flexible in what we program in. Some clients dictate a specific language. Right now we're just embarking on a large project where the client wants it to be in Microsoft C, which we've never used before, so we have vast quantities of a library source code that are in C and Pascal, and we're in the process of figuring out how to make them accessible in Microsoft C. So the client sometimes dictates it. Other times it's just a matter of what your current generation of products has evolved to. Right now, C seems to be the most popular by far.

Mr. Mahany: As you said, I've run the gamut. PTS is written in APL, but we had some people who just would not even consider buying software because it was written in APL. They had an attitude that they just didn't want to go near it. The clients that like APL, like the ability to customize it and do their own thing and create "what if" type analyses. The biggest problem with that is that you come out with a new release. Perhaps it had new regulatory support or support for any type of CMO. The company would not implement it in any kind of timely process because they were the people responsible for implementing the customization that they need from the previous versions into a current version. So you have clients frustrated about a bug in this version. While we fixed that bug two versions ago, we never upgraded to the current version. So you constantly go back and forth.

In our developing environment, a big component is that database and being able to not only do the calculations, but give you the intermediate calculations and the final calculations in a format that you can play around with and use to do what if analysis and what we call ad hoc querying. The code itself is written in C++. We've taken a real aggressive approach about the cost of customization, as well as the timeliness of the turn around to try to get companies to hire us to do everything, so that we can keep the versions current. When we send out a new version, almost all of our clients implement it right away because we've regression tested it and the prior customization began and was carried forward in future versions. We'll still give access to the source code. We have a provision in the contract that says it. I don't think we've ever sent it out. I'm not sure what we're going to do when someone calls up and wants it. It's there, but it's there more for comfort because we really don't want you to touch it. If someone is going to screw it up, let it be us. Everything we do is in C++. We find that C is a little bit faster in some of the crunch routines, but we're doing things in C++ and using a database.

From the Floor: What database are you using?

Mr. Mahany: We're using a product called Century. It's a sequel-based product. The reason that we chose it four years ago is because it was, at the time, the only front-end database package that operated with both the stand-alone machine environment, as well as in the network. You don't have to have two versions of the code. The second thing is, when you think of using a database, you use it so little. It's just for all of the plan assumptions. So a big plan database might be 20 or 40 megabytes for us. It's very small compared to the resulting files that we produce. Having to install an oracle in-house to house your plan database is like killing an ant with an elephant. We've gone to the lower end of that. But if the client wants us to, we'll customize the application for an oracle or something with bigger high tech engines.

Mr. Karo: One other thing about the source code that you had mentioned is it seems like clients are spoiled in that respect. We will license out our source code now at an additional cost to a lot of clients. Over the last two years, we have had a large in-house effort to really boost up our documentation, so that we can give a training course and have key people at the client's office who will sit through that training so that they're familiar with the engine and the interface and everything like that and learn to program new products. We'll kind of take them through a mentoring phase where we'll give them technical advice, but not all of our clients are comfortable with that approach right now because there is a great deal of movement towards outsourcing and a great deal of uncertainty in the information service department. Some people would rather just leave the whole responsibility with us. They know they have dedicated people that are familiar with it, rather than having to worry about turnover among their own computer programmers.

From the Floor: I just want to say, one of the reasons people like my source code is because they don't know if the company is going to be around in a few years, so they have the source code to sell that product.

Mr. Mahany: I have found that it's a safety blanket. If we didn't put it in our contract, everybody would ask for it. It says that within ten days of your request, we'll send you the source code. We've been doing it for about four years now and we've never had a request so it's a good safety blanket. Every company out there has been burned by a vendor in some way, shape, or form.

Mr. Karo: It sounds good on paper that you have source code, and that you can make the changes. I'm dealing with clients that we've given the source code to, and they've brought in contract programmers because they were sure that they could just take these contract programmers and update the source code. They just retain us on an hourly basis to give them advice. The contract programmers call me up and say that they've been looking at some line of code for two weeks, and they don't understand what it's doing. How do they make this change? It'll be an unbelievably trivial change. So just having the source code isn't necessarily a panacea.

From the Floor: We have been using a lot of vendors' software and I'd like to make a general comment that the documentation is pretty bad. It is confusing at best. I think one of the actuarial careers could be writing better menus.

I don't want to contradict what you just said about C++ because I don't understand languages very well. Some companies might have a corporate home with ten different companies and they don't want the different companies to have different vendors of software. They want to have one vendor and the information going, freely, from one company to the next. The best is to have a spreadsheet and a system that can connect all the spreadsheets together.

Mr. Mahany: We haven't.

Mr. Lynch: We've studied the languages quite a bit over the decades and we have stayed with Fortran for our valuation engine in part because the strong part of our software is the capability of doing entry point modifications. The Fortran language is much easier for clients to understand and read because it is more actuarially oriented than any of the other languages. We've stayed away from C and C++ in part because of the power of Delphi concept engine. It is an object-oriented engine that allows one to generate Pascal automatically, although Pascal has recently come out with a very good C++ that uses the Delphi concept engine. We stayed with the Pascal. I have felt that Basic has been underrated as a language. It's an easy-to-use, easy-to-understand language. The chief problem is because major systems have not been written in it, there is not the body of people who have put in extended powers that stop your developers, press the compiler makers to add.

Mr. Karo: I'd like to respond to your question about documentation. I'm probably speaking for the other panel members when I say that we probably would love to turn over a software product with manual after manual of great documentation, but I think the two forces are always tugging at each other. The marketing department has a very aggressive deadline. I've never come into a software project where they said, "Take as much time as you want, just as long as you give us good documentation on this system." Rather, we hear, "We need this very quickly." If we ask for more time to get the documentation ready, I don't think we'd be given that time.

Mr. Mahany: I guess I know more about the documentation. I think it depends on the type of application or software package that you're writing. I spent six-and-one-half years with Chalke working with PTS. The technical reference documentation was not up to speed, shall I say. The market demanded that we implement something very quickly, and I think we got off to a bad start with it and it never caught up. We wanted to bring something to market and the technical reference manual sort of fell behind. At Price Waterhouse I think we're dealing in a valuation marketplace that is very specific. Companies are worried about a number that makes up 60–90% of their balance sheets, so they are very concerned about the quality of that number. Price Waterhouse is an audit firm. There was a fundamental design component that I'm not responsible for. I wasn't responsible for developing it; I'm responsible for maintaining it. If you have our technical reference manual or out of detail report in a calculator, you can reproduce any calculation that's done in the system on a given policy. So the technical reference manuals for our products are anywhere from 1,700 to 2,100 pages and we approach the specification pages the same way, whether we're fixing a bug, or whether we're doing a new regulation as I mentioned. We actually have an actuary sit down and fix the code on the spec page and that's what's handed to a developer. Those spec pages are turned over with our release.

So I feel like I've seen the extremes in terms of documentation. Clients never liked it with PTS, but it didn't keep them from buying the package or at least it didn't influence the ultimate decision. In today's marketplace, in my valuation marketplace, it will keep companies from buying it if they feel like there's not a real strength in the documentation relative to the other packages that are out there.

Mr. Van Dam: There are careers in software that do not involve programming. Mark is one example. I have an independent consulting firm for which I do independent actuarial consulting. I do work for an administrative software vendor. My job did not involve writing code. There are a number of different things that my job did involve, and there're a number of different roles for which actuaries with different skills can play. There's a support role for the sales function that Mark talked about. There's also a project management role there, if an actuary was interested in project management. The administrative software companies are desperate for people that understand insurance. If you have an interest in project management, you'll find that the salaries are good for a career ASA, but they may not be what an FSA would want. They might be fine for a new FSA, but the FSA probably would not want a career there.

There's also room for people who can understand the system in terms of what the program ought to really be doing. You have an understanding of insurance and most actuaries have a basic understanding of what's involved in programming because they've done a little programming along the way. They may not be programming a very big application, but just having the knowledge and understanding helped in terms of working with clients, in terms of getting the specs in the system, and in terms of making sure the system can handle the products that come in. There's another role, in implementing the products.

We talked about the outsourcing. I don't how many of you work with administrative software. There are a lot of actuaries that are involved in trying to get products up on the administrative software. Some of that work is also going on in the actual software houses. These roles have basically turned into consulting type roles where you're billing your time by the hour. Some of those roles have a lot of similarity to what you think of in terms of a traditional consulting role.

Mr. Lynch: I don't know how it compares with other software firms, but in my software firm, we have a function called Service Bureau Computing. Smaller firms or firms that are considering leasing our software or firms that have overloads send us pension cases to do the valuations in our office. It is similar to the way they would do it in their own shop if they were leasing or had purchased our software. We run the valuation for them according to specifications that they give us and then return the results to them. This is an area that we're going to be expanding in the not-too-distant future. This is not an offer by any means. We will be running ads and expanding this capability in the near future in our office; that's a growing area. In our office, we have seen a growing need for a full-time function for quality control and our quality control person is a very powerful person within the company. People jump when she talks.

Mr. Van Dam: All of you raised your hand when you talked about software. Can anyone here talk about work you're doing in terms of Internet software? Dave, are you doing anything with Internet software at this point?

Mr. Karo: Yes, we've released one system, where this one company's sales is done via the Internet. They have no agents or no field force whatsoever. So they have a calculation engine up there doing their proposals through the Internet and we've also started work on a couple of other similar applications for other clients, as well.

Mr. Mahany: We've done research and we've actually implemented it in at least one case. We find that a lot of companies right now are still dealing with the security issue of the Internet and access to people's computers. We're looking at it as more of a client support role. We've done some experiments where we have gone out to the client's site to fix the software or to try and recognize what problem they might be having because of their data; we'll actually go out on the Internet and take control of their machine and are able to debug it and use it that way. Everybody thinks about selling new products, but as far as a client support avenue, its capability is unknown.

From the Floor: There was a session about rapid application development (RAD), and I wonder if any of the panelists would comment about the use of RAD techniques in their shops.

Mr. Mahany: We've looked into some of the tools. We've looked into Delphi and Power Builder, but we haven't actually built any projects yet with them. Some of our clients have been experimenting with that and they're in the process of building calculation engines with them. I'm not sure how successful they've been up to now.

Mr. Lynch: We use Delphi to accelerate the development of our Windows 95 Software. We converted our DOS product into Windows 95 without ever having used Windows 3.1, except via a DOS box. We converted our entire system into Windows 95. It included a change in Fortran compilers, and it included rewriting all of the interfaces. Changing the Fortran compilers was fun because we had switched. It does now. We switched to Microsoft Power Station. Just after we had made the complete conversion, Microsoft announced it wasn't going to support it anymore. We're going to be turning over some rights to it to Digital Equipment Corporation or DEC. So we switched them to a DEC Compiler, which had bugs. It's all debugged now and everything works.

Mr. Mahany: I know we've used them but the good news is, I have no clue which ones we have used. I can give you the name of someone in our shop that can tell you which ones they use. I just sign the check for the items they buy.

From the Floor: Can the other two panelists comment on Joe Lynch's graphs about the changing relationship between the programmer, the actuary, and the technologist in software development. One of the charts showed how the technologist has become the most important person in the equation. First, do you agree with that, and second, how has that affected your work.

Mr. Karo: I do agree. I haven't put as much study into it as Joe has, but I do feel that technology is becoming more important. It's certainly changing much more rapidly, and you're asked to be able to support a number of different platforms that are constantly changing. We have a number of people in house whose job it is to basically tinker with different technologies and find out what works and what doesn't work. The line programmers don't really worry about those things. They know they have the resources that they can fall back on to find out the answers to the really hard technical questions.

From the Floor: I guess I agree that technology is becoming a more important component. I don't think you can be a dope about computers and still be fine and rise to be president of your organization. I think it's just a matter of accumulating talent. For instance, I have a client support role. At this time, I have six client support managers. Those client support managers range from someone who can barely turn on a computer to the opposite extreme which includes people who spend time at home on the Internet and time at work on the Internet. I can tell you that the people that know more about the technology play a more important role because when the clients ask questions and when we have meetings, these people play a relatively more important role than people who don't have that expertise. In my organization, it makes them more valuable because I don't necessarily have to go out and hire someone with that skill set. They built that skill set, along with the other basic things that they need to do to their job. So they're relatively much more important.

Mr. Van Dam: Another thing that I had written down as an alternative career that actuaries would be good at in a client server environment is a database administrator (DBA). I think a lot of the actuarial skills would translate well into a database administrator position. I'm not that familiar with it, but it was a very good paying job for those of you who are interested in technical types of careers. That's something that you may want to look into; however, it doesn't really use a lot of actuarial training per se. One last question I had is, has anybody here in the audience worked with some of the information technology consultants like Price Waterhouse? Have any of you had any experience working with that group of people? I haven't necessarily seen a lot of actuaries move into that role. Mark, do you have that experience? Do you see any cross fertilization with the group within your firm?

Mr. Mahany: More of the cross fertilization has been on the life side in terms of the data warehousing and that type of thing. Even though I would say that it's life versus property/casualty the life industry has a much better feel for commercial software and the value of it relative to developing in in-house. The property/casualty actuaries are still of the mind that no one can do what I can do and I'm better off building it myself.