RECORD, Volume 25, No. 3*

San Francisco Annual Meeting
October 17-20, 1999

Session 31PD
Data Warehousing For Actuaries

Track:	Computer Science
Key Words:	Computer Systems, Modeling Tools

Moderator:	CHARLES E. RITZKE
Panelists:	KEVIN JOHN PLEDGE
	NEIL RADEN†
Recorder:	CHARLES E. RITZKE

Summary: Many life insurance companies have some data warehousing project in progress. Much activity in this area has been focused on marketing, sales, and accounting. Less work has been done in financial management areas, but this is expected to grow over time.

This session focuses on data warehousing applications primarily of use to actuaries, such as experience studies, profitability analysis, and performance measurement. The panel discusses practical applications of data warehousing for actuaries, including examples taken from real implementations.

Mr. Charles E. Ritzke: This panel will educate us on some of the practical applications of data warehousing for actuaries. Our first speaker is Neil Raden. Neil is the president and practice director of Archer Decision Sciences. He is an author, lecturer, and consultant. He founded Archer 15 years ago to solve problems in analytical applications. Archer is one of the most well-known and respected consulting organizations in the industry. Neil started his career as a casualty actuary.

Our second speaker, Kevin Pledge, is an actuarial vice president with the Independent Order of Foresters in Toronto, Canada. Kevin is a Fellow of the Institute of Actuaries (FIA) as well as an ASA and an AAA member. Kevin transferred from the U.K. to Canada three-and-a-half years ago to manage the Corporate Actuarial Department for the Independent Order of Foresters (IOF). Kevin took responsibility for financial reporting, which at the time was mainframe-based with little consideration for the coordination and consolidation of results. Convinced that there must be a better way, Kevin sought the assistance of Neil Raden and between them they introduced decision support system disciplines to the IOF.

*Copyright © 2000, Society of Actuaries

†Mr. Raden, not a member of the sponsoring organizations, is President and CEO of Archer Decision Sciences, Inc. in Santa Barbara, CA.

Note: The charts referred to in the text can be found in the pdf file 9sf 31pdch1.pdf.

Well, if any of you are like me, you may not know that much about data warehousing, but you've been hearing a lot about data warehousing projects that are being implemented in various companies. So you may be wondering what all the buzz is about. After all, actuaries have been collecting, storing, manipulating, and analyzing data for probably the last 50 years and we would often 'warehouse' that data. Of course, the kind of warehousing technologies we're going to learn about today are a little more sophisticated than that. I know that I'm interested in better understanding the differences between these new technologies and what we've been doing in the past.

Mr. Neil Raden: I don't want to tell you very much about Archer Decision Sciences because that's not very interesting. I guess what is interesting is we've been doing this for 15 years. I started this business not long after abandoning my career as an actuary or, as Kevin's boss told me, as an actuarial analyst. He was very clear to make that distinction. But we've been trying to solve the problem of how you do analytical work for well over 15 years. How do you understand your data? And how do you solve problems with data? Thank goodness data warehousing came along a few years ago. It really helped, and we'll try to explain why.

Many, many years ago when we first started host-based reporting and decision support, tools like Focus led to this notion of executive information systems— "Wouldn't it be nice if we could just distill this data and put it in a nice, happy format?" I call this the Fisher-Price approach. Executives could punch big, pretty buttons and just get all the data they wanted. Problem was we didn't understand at the time that the real underlying problem was one of data management, and the data was a mess. So, instead of trying to solve that problem, we did what most technologists do, which was to move on to the next technology platform instead of trying to solve the problem. Most people at that point jumped into PC spreadsheets. That's an epidemic that is beginning to slow down but it is still a problem for us.

We moved into client/server applications just because they showed more promise and more scalability. Data warehousing luckily came along and rescued us. We understood the analytical problems. We understood how to solve those problems. But we really didn't understand how to deal with the massive scale of data or how to be good stewards of data over a period of time. Data warehousing really created that methodology. The problem is, if you look at the Chart 1 time line, that it seems data warehousing is declining. Now, I'm not saying that data warehousing is dead. I think that data warehousing will last a while. As a company our role in data warehousing is beginning to diminish because we've been there, and we're ready to take our clients to the next step, which is to build these closed loop decision support systems and to integrate the analytical processing and the operational processing in a way that people can begin to integrate analytical thinking and data into their daily decision-making process up and down the organization. We used to think that up the organization was the most important thing. I'm beginning to think the most important thing is down the organization, and probably the best analytical application interfaces are the ones that are completely transparent. For knowledge workers in your company at whatever level, the absolute key is to integrate data and models into the decision-making, and this is where I see closed loop decision support systems heading. At the end of Kevin's presentation, when he takes you through the case study at IOF, I'll try to tie this together with some current trends and give you an idea of where it is going.

I had a few comments I wanted to make as I was thinking about our session. The point I'd like to make is that data warehousing for actuaries is a solution to your problems. Now, there are a couple of things at play here. First is that there is a perceived problem—one of handling of data, of auditing, of cleansing and merging, and of reconciling data—but data isn't really the problem. That's only the perceived problem. The real problem, or what's actually presented after you look at the problem, is that you don't see the opportunity to improve your entire process by doing this. Kevin will explain how this happened at IOF. Just to give you some highlights, what used to take five months in terms of assembling data to do a valuation has now been reduced to five days. The valuation process was only annual; now it is quarterly. The staff has realized vastly improved knowledge about the business, and I think that this is key.

At the end of the day, the most important thing about a data warehouse is not the database. It is not the data. It is not the machines. It is not the technology. The only thing that's important about data warehousing is that, at some very fundamental level, you change the nature of work in your organization. If you don't start with that vision and aim for that benefit, you might as well not spend the money because anything else is just transitory. The kind of benefits you can get from something like a data warehouse exist at some low-level, operational improvement of productivity and it is temporary and localized. You want to attack the fundamental nature of work. You want to look at your organization, like Kevin did, in a visionary way and say we can take this group of people, and we can turn them into a group of energized, motivated professionals who enjoy their work, who collaborate, who work as a team, and who are freed from the daily struggle of doing grunt work so that we can have them do the work that we're paying them to do as professionals—that's what you have to aim for. Anything else is too low, and you might as well not waste your time.

The second point is that you need a data warehouse even if you have one. Let me just ask a question. How many of you work for insurance companies or in consulting organizations where you have your own data where you would need a data warehouse? OK, quite a few. How many of you have a data warehouse currently? OK, some of you. Most of the data warehouses I see are temples to databases that are built by information technology (IT) with very little interest in the business process. I suspect that many of you already have something called a data warehouse; it might have been extremely expensive, but it is probably not useful for you. You do need this, and we're going to explain what an actuary in a life insurance company needs in terms of a data warehouse very shortly.

The third point is that you also need some help, and you need some help because it is too difficult to discover all of this yourself. What you don't need is for someone to come in and live in your building and do this for you and say, "Just step aside.

We know how to do this. We'll do it for you." But what you do need are experienced people who can give you the vision and direction and bring to you the best practices in the industry. Most of all, you need someone who can design the architecture because that is the trickiest part. A data warehouse is not a database. It is not a piece of software. It is an architecture-based solution that's designed to be scalable. It is designed to be maintainable and usable in an environment of change. That's not something that an amateur should attempt to do. The architecture is key. If you don't get the architecture right, you're going to waste a lot of money and a lot of time. That's where you need help.

Now, alternatively, perhaps you need help with project management. Perhaps you don't. Perhaps you need help with certain specialists or certain pieces of technology. You might even need help with certain subject matter experts. Maybe you're just shorthanded of staff. If you attempt to do this on your own, it will take you far too long because it is a solved problem, and there are a lot of people out there who already know how to do it.

Now, I think in the long term success in any IT implementation is not based on the quality of the technology implementation or the volume of data that you can access. Ultimately, success is going to be driven by how efficiently you can access and integrate relevant, in fact the most relevant, information because the amount of data available now is just proliferating riotously. Every day there are ten times more data than there was the day before, so it is not a question of getting access to it.

The problem is that most people approach these problems of data with a great deal of fear because they don't know what is difficult, and they don't know what is easy. When they look at a problem and they can't understand it, their immediate response is to shy away from it because they can't assess how difficult or how easy it is going to be. It is analogous to if I push this button, what is going to happen next? That's a very simple case. But most people won't ask for what they really need because they can't tell if it is going to take you a month or a minute to give them that. So, the first thing that we have to do is to set up this environment where it becomes more intuitive for people to understand what can happen quickly and what is easy to do and what is not.

Second, your goal should be not merely transferring knowledge, but building differentiated and self-regenerating professionals because they're going to move on. You all know that you have turnover in your organizations. In the next 20 years half of the professional workforce in this country will be in nontraditional employment. You should not expect people to take one or two years to learn how to use tools that you build in your organization. They're not going to spend the rest of their career with it. You have to focus on collaboration and motivation. I like to use the metaphor of an artesian well. All you have to do is dig the hole, and the hydrostatic pressure keeps the water flowing. It is a self-generating process. That's what you need to do.

Now, there are going to be a lot of unforeseen challenges. If you think that e-commerce or e-business is something that won't affect you in, say, the life insurance business, you're quite mistaken. It may not reach you as quickly as it reaches some other businesses, but it is going to vastly complicate your job. If you're not ready with your own data you're going to have the problems of temporary and transient alliances with third parties—both your customers, your suppliers, your reinsurers, and everybody in whatever chain that touches your organization—you're going to be exposing more and more data, and that creates a lot of opportunities for mischief. So, you have to get your arms around it. It is clearly the most important thing because this whole process is getting very porous.

Now, I want to turn it over to Kevin because I want Kevin to explain to you how we went through this process at IOF—the problems they faced and how we solved them—because I think you'll find it very relevant, even those of you who aren't in a life insurance company.

Mr. Kevin John Pledge: I'm going to discuss various aspects of the project we did at IOF. I'm going to talk about the project management, the problem we set out to solve, and the end solution as we arrived at it. An important thing to note is that the IOF reports in more than one country across a range of products, and we had systems that worked, but they didn't work particularly well. The basic process that we had before we introduced the data warehouse was a series of individual extracts that led to individual expertise.

On Chart 2 I have various mainframe systems which were extracted using individual customized extracts mainly written in A Programming Language (APL). The important thing was that it took an individual who understood those programs to run those reports. This situation is extremely limited, and it becomes even more complex when you consider some of the other applications that we're trying to deliver, such as experience studies, asset/liability, and other functions associated with our day-to-day job. We end up with a complete mess of extracts being managed by various people with interrelated applications that are difficult, at best, unless you have the same person to perform the work each year.

So, we relied very heavily on staff expertise, but the staff expertise that we relied on was very specific to a particular process or a particular source system. We were trying to develop expertise that was not dependent on having to know these particular processes and source systems; instead, we concentrated on actually doing the work that we needed to deliver. As Neil emphasized, we set out to solve a few problems such as data management, yet we ended up discovering that we not only solved the data management problem, but we also introduced a better, more flexible way of doing analysis and changing the way that people worked with the data. That was a most important deliverable that we had from this.

Initially, we didn't know where to start the project. We attended a conference just like this one about two years ago. I met Neil and some other people who suggested that a data warehouse would be the solution. Neil at the time was insisting that our problem was not with an audit trail. It was not with managing our data extracts. It is how we use the data, and that's what a data warehouse is for.

It took me a while before I learned that. We started off looking at just solving our first problem, which was to organize our data. The valuation system we use is a PC-based system called Axis. The original plan was to take the data from our various source systems, consolidate it into a detailed data warehouse, and then use that to feed our valuation system. We extended that original plan to take data back from our valuation system and build a number of additional analytics or analyses of the data warehouse. So, we would actually analyze the results of our valuation system through the warehouse. Things like experience studies and profit analyses were done consistent with that.

Since starting the project, our company (as are a lot of companies) has been looking at customer relation management. A lot of the work we've done in defining the data and the data extracts is obviously reusable, so we're now looking at potentially leveraging our work in terms of supplying the data to the customer relationship management system. The project structure that we set up to solve the problems takes a bit of explaining. The first thing to consider is whether you have the infrastructure in place. For example, do you have the hardware? Do you have the server? It is normally an IT responsibility, but it is very relevant if you don't have the hardware to run it. And then we have a number of roles that we defined around the project. We have the extraction transformation and loading tool (ETL). This is a specialist tool that takes the data off the systems, combines them and loads them into your data warehouse. The project managers are specialist roles. The architect is like the designer, the person who looks at how you're going to structure data and structure your project, whereas the project manager is concerned with day-to-day work within the project.

Then within the three stages that we talk about within the data warehouse, we have acquisition of data. This is the extraction and transformation of data, loading it into your warehouse. That relied quite heavily on consultants on ETL tools and some people with subject matter expertise who know the source systems. The next stage is the storage of the data. Again, the database analyst is required to set up the database structures and ensure the data management and organization works as you expect it to. It doesn't actually run in this order because all three parts run simultaneously, but the final team that we had working on this were the people coding the analytics with access to the data, largely actuarial staff who know how to do the analysis working together with people who know how to use the tools.

There were three original objectives, and we satisfied them. We are able to manage data extracts through the use of tools that have a graphical user interface (GUI). Metadata are the basic definitions around your data. There are basically two types, the first type is a definition of what a piece of data means. So, for example, a premium has a particular definition. It is very common to find that everyone thinks they have the right answer, but it often has different definitions. And then there is the other type of metadata that are automatically generated within the tools that we use. This will tell you, for example, when the data was loaded, whether it was transformed, whether it was cleaned, or whether you've converted it in any way. Then there is the tool itself, which actually manages all the extracts, does the transformations, and keeps a log as to when the extracts were run.

The audit trail was our second requirement. Our auditors had insisted upon this. A big part of what they basically wanted was that the extracts run as expected along with a list of any errors found in the source data. It is all pretty boring, but it is all generated as part of the tools and part of the standard implementation and it's very straightforward. The data itself is stored in the star schema. Do you want to explain it, Neil?

Mr. Raden: Has anyone in this room ever heard this term star schema? Is that familiar to anyone? Surprisingly, a few of you have heard of it. What you're looking at is a logical model that you implement in a relational database. In Chart 3, each of those boxes represents something called a table. In more general terms one of those boxes is called an entity, and the things that tie entities together are called relationships. That's the technical talk. The point is what you have is something in the middle, and in this case we're talking about policy facts. Typically these are numeric values. They're things that you can add up. And the things around the sides are the dimensions. If any of you have worked with APL, and I know most of you have, you know what I mean by dimension. I might be looking at additional premium, base premium, or issue age and so forth as facts, and I may be constraining on which of those I look at or I may be aggregating or disaggregating those based on the dimensions that surround that fact table like plan code, reserve type, and so forth. So, those dimensions become the row and the column headers. They become the control breaks and so forth, and they also become the qualities at which you look at information.

The whole concept of the star schema is to take business data and to separate it from its original purpose and to put it in a purely analytical kind of schema. This allows for fast analytical work so that we can look at information within a relatively large database without having to extract big chunks of it to different pieces and create a management problem. So, this is the star schema. It is a wildly simplified version of what a real dimensional model would look like for a life insurance company, but it is good enough to give you a flavor. So, the point is that you might look at that table up on the left where you have a plan code and say, "I'm interested in product type x and fund type b, and I'm interested in a particular reserve basis in a particular country." Let's take a look at the base premium and additional premium by month. Any query of any combination of those things is possible with this schema. That's the whole idea. Now, if you can imagine taking data from your organization from 12 or 15 different sources, like we did at IOF, from different lines of business, different jurisdictions and so forth, and putting that together into a single schema like this, I think you can begin to understand the complexity of the job. The point is, once you have it in that schema your analytical capabilities are virtually unlimited.

Mr. Pledge: The actual star schema is much more complex. We actually have 20–25 fact tables in our data warehouse and probably about 35 dimension tables. They work in relationship to each other. Some of the structure depends on individual requirements such as how frequently you need to load your data, how frequently you're going to do analysis, whether you can load some data less frequently, and the level of detail you want to hold.

Mr. Raden: The sum total is that most of your databases, even if they're relational databases, are incapable of providing any kind of analytical query. If you've already installed SAP or you have some of the other administrative systems, the vendors at one point or another probably crowed about the fact that the data itself is stored in a relational database; hence, that means you should be able to query it freely. But you know that's not the case because an operational schema is simply incapable of swimming through this kind of data and giving you quick answers. That's why it requires a physical transformation. I think that what you'll hear are some complaints saying, "This all sounds great, but you know what? We have a lot of data with TPAs, alliance partners, and so forth. We could never actually do a data warehouse because the data would always be incomplete." This is not true. Provided that you can actually physically get the data, you can model it and get it into a data warehouse, and that can become part of your data. The other part of this, and I think the part that's even more interesting is reusable analytical applications and components. The point is you want to get to something that I like to call "late assembly by stakeholders." That's akin to giving your actuaries and other workers Legos that they can snap together. The point is that the data is of no value at all if it isn't presented to people in some sort of a context. So, it is not just the data. It is the models themselves. It is the rules. It is how these things are actually calculated. And instead of building one report after another or building one massive spreadsheet after another, what we aim to do is to create reusable components to create subpieces of models that will behave predictably and snap together. This is really the Holy Grail of data warehousing. This is the end game, and this is where you really start to pick up velocity in terms of getting benefits out of what you do.

Mr. Pledge: The movement analysis that we were doing was an example of the benefits of reusable components. We were able to build the profitability analysis onto that work by substituting policy count by reserves and then adding extract components of other items that you need to bring into profit analysis. In fact, one thing that we found very interesting in this project was actually looking at profitability and profit analysis. We went back to a number of old actuarial papers, one in particular by David Keller and Sue Collins, which gave us examples of how we could build a profitability analysis from reserving and pricing data. We were able to generalize that and move it into our model. What is interesting is that the volume of data, I think, would have been a real obstacle if you tried that before.

We delivered added functionality for our experience studies, providing both added speed and flexibility. The approach we took was really something that we've wanted to do for a long time. One other thing that we added is an executive information system. As Neil described, it was a Fisher-Price computer interface with big buttons. We find that this is really useful to communicate some of the issues and work closely with other departments. Our challenge now is how to make our tools more useful for other departments and to work closely with other departments. This is so important that I have one position in my department that's focused on information communication and how we display results to people and involve them in the results; that's a full-time position that we wouldn't have considered before.

I'd like to quickly go over the components that make up our data warehouse. We start off with the extract and transformation piece managed with a transformation tool. The one we selected is Informatica, but there are a lot on the market, and what you choose would depend on your individual needs. We have a staging area for our data and we bring up data off the mainframe basically as a copy of the data. We bring down about 20 gigabytes of data every month and just load it in and start working with it on the server.

The next component is the storage. Again, we're using a specialized data warehouse database. The one lesson that we learned is that, even if your organization has a corporate database standard, it may not be the standard that you need for your data warehouse. So, you need to be very careful of that. And we also have our metadata, the data that tells people the definitions and gives information where the data is stored. This metadata is stored in an open relational database. So, it is open to everybody. Other people in the organization can read this metadata and use it, even if they're not using our data.

The third main piece is the access of the data. Again, we've tried using a couple of tools. We're now using Microsoft online analytical processing (OLAP) services, which comes with Microsoft Office 2000 and SQL Server. But in total, we're using, at last count I think, something like 170 tools. So, it is worth seeing what is the best fit for your purposes. We're using one that's very simple to get up to speed with and provides us with the flexibility we need. It also has the ability to deliver the results over the Web, so we can really cut down on network traffic. When I say the Web, I mean we can use Web-type technology over our own network and avoid network traffic and slowing the network down by just delivering the end results and having all the work take place back at the server.

The final components are our valuation system, which we implemented across all lines of business at the same time and the product information system that we integrated with this. The latter system took advantage of the leveraging that was possible from the Web-type delivery system. Those are the components.

An important distinction to make is the difference between how the actuarial department and IT perhaps views a data warehouse. Neil has some views on this.

Mr. Raden: Is anyone here part of an IT organization? You can close your ears now. We're going to say some nasty things about you. IT people sometimes have a stilted view of data warehousing. They look at data warehousing as a database. It is a good excuse for them to buy some nifty hardware. I don't mean to sound so cynical, but the problem is that people in IT sometimes just don't get it when it comes to data warehousing because it is something they've never understood.

Analytical processing is always something that's been done by knowledge workers, not by IT departments.

Now, I know that there are exceptions to this, and, in fact, I remember working in an insurance company where we had an IT organization that was staffed with actuaries specifically to support actuaries. That was a rare and unusual situation, I think. But for data warehousing to be effective it has to be about your business process. So, instead of starting with "What data is out there? Let's go get it, and let's put it in a big old bag. Then let's slap some tools on in front of it and let people go query the data so we don't have to write all these reports anymore. Some of you are nodding your heads. I think you've already seen this. That's backwards.

The way you build a data warehouse is by starting with the thought, How good is this going to feel when I'm finished? In other words, what is your vision? What kind of an organization do you want to be? What sort of problems are you trying to solve? Work backwards from that. And when you work backwards like that, it will suggest the kinds of analytics and the kind of models that you're going to need. That will inevitably lead to the kind of data you want. Don't start with the data because data is pretty much irrelevant. Picasso said that art isn't truth. Art is a lie that makes you realize the truth. Well, you could say the same thing about data because the data isn't even real. Most of the data you get are generated by some operational process. It is questionable how much of it is "real". Some of it is a snapshot. Some of it is a guess. Some of it is an allocation. What makes it real is the way you apply it.

The problem is, if you're going to get support for a data warehousing effort from IT, it is probably going to be at the wrong time and in the wrong place, and will involve the wrong kinds of things. So you're going to have to develop some sophistication in the way you deal with IT to get what you need from them. They don't understand your business process, so it is really important to hold them at arm's length and to use them as service providers. Now, there are a couple of things you have to do. First, at IOF, we had a problem because we simply didn't have the infrastructure to even start this project. We had to build our own network, we had to buy our own servers and software, and so forth. We had to educate the IT department in what we were up to.

The IT department may act one of two ways. One possibility is that over a long period of time, they will adopt a posture of passive aggression, which means that they'll stay out of your way for a while until the worst possible moment, and then they'll raise their objection and stall things. You have to watch out for that.

The second possibility, and this is the biggest trap that you need to watch out for, occurs when you roll out your first prototype or your first working application. By the way, when you build a data warehouse, you don't try to boil the ocean; you do it in baby steps. You start with a small piece, and you build incrementally. You don't go into development for three years and say, "Listen, just leave us alone, and in three years we'll have something that'll just be so wonderful. Get out every 90 days with some new functionality." But the problem with that is that the passive resistors will wait for that first prototype, at which point they will become your biggest ally. They'll talk about what a wonderful application you've built, and then when you're out of the room they'll say, "Well, we don't have to spend any more time on that data warehouse. They're finished. Didn't they do a great job?" So, you potentially can have a lot of trouble with IT, and it is not because they're bad people. All people are bad people when you put them in a bad situation. They have their own requirements, and they really don't mesh very well with yours. So you're going to have to get some sponsorship outside of your organization. You're going to have to get some senior people in your company who understand your business case and what you're up to because IT alone isn't going to be enough as an ally.

Mr. Pledge: As Neil said, we had to address the infrastructure ourselves. It is a lot of fun buying a lot of hardware and some of these big machines, so I do have some sympathy with IT there. The important thing to appreciate is that we've moved the analysis away from the source system. We want to reformat the data based on the way we want to approach it. We have all these fact tables and dimension tables in a data warehouse, but that is not the end solution. You need to organize those fact tables and dimension tables in a way so that the end users can easily use them for the applications that they're actually building.

I want to discuss a very simplified example of 3 subject areas out of the 20 or so that we've consolidated within the warehouse. The first one I want to start with is the valuation system. That feeds information into our certificate fact table. You put valuation results into the certificate, and the certificate table itself also feeds the table that's used for the movement analysis. I'll discuss why that is later. There are a couple of things I'd like to note. I've referred to the results of the valuation system, what we actually file, as a reporting data mart. I'll explain a bit more about that later, but basically there are many ad hoc adjustments that we actually get at the last minute and at different levels of the data. The analysis that we do on these adjustments is more limited and less revealing. So, we've focused on keeping the data, filing it, and making that process simple.

If I start with the certificate star, we have the star schema again, and we have the certificate fact. Joining onto that we have currency type, which will tell you what currency your information is in—things such as premium reserves. Then linked to both of them is a time dimension, because obviously you need to know the time in order to determine your exchange rate and to determine your data as-of date. One of our big lessons that we discovered was in certificate status. I don't know whether other companies would have the same problem, but on a lot of our systems, when a policy became paid up, for example, a new paid-up policy would be issued on the system. So, you never really could track a premium-paying certificate through to it is paid-up status. So, we changed that, and now for a status we've introduced two primary status keys—a premium-paying status and benefit status. That's an important thing to take away if you're thinking about doing this.

The certificate dimension includes a lot of the nonadditive information around the certificate, things like age-at-issue, their name, their address, or anything we might use to trace back to the source system. The geography of the hierarchy is based on where the certificate was issued, so it would include the state, the country, and the continent. Then there is the product hierarchy. This would be the plan code, the plan grouping, the plan, the fund, and so on. So, if we wanted to ask a question such as an age distribution of reserves, we would connect the certificate fact to currency type, because we'd need to know what currency the reserves are in, and to time, because we need to know which date we're talking about.

So that's the certificate star, and it is designed so that people can reuse very simple queries. Now the movement is basically the same as the certificate but at two time periods. You can answer all the same questions off the movement star, except it is much more complex. So, we try to keep the basic views for the users as simple as possible. And, again, this just has two time periods, two product hierarchies, and two geography hierarchies. We use the geography as of that period, or the product hierarchy as of that period.

The final subject area I want to discuss is the financial reporting. It really isn't based around the star schema. We can take the data from the data warehouse into report extracts, but we may get additional data that we don't have at the certificate level. For example, adjustments at fund level are not always available at the certificate level. For analysis at the certificate level we set a small data model or data cube—bring that data in, and the analysis is available at that level. Of course, the data can be added into the final report, but then we can't do much more with it. There is also a risk of inconsistency.

I have one final note on the workload of the project, say, from a project management perspective. We went in expecting to spend a little bit of time on data quality. We had an existing process that looked pretty good, and we didn't have any reason to assume that our data quality was anything to be concerned about. We'd done the acquisition of data before for other processes. So, again, we thought there'd be more work there. We had some indication of that. We're bringing in external experts to design the storage of data, so we knew we really could have a pretty good grasp on that. And, finally, there was the analysis. We took a stab at that.

For the end experience, most parts were actually less of a problem than what we expected, particularly the analytics and the acquisition of data. Once we started delivering some results, they kind of snowballed. You can deliver more and more through the re-usable components. But the data quality was the real issue. When we sat back and thought about it, we realized that it was really the first time that anybody had gone back and combined source systems. It may not be that any one source system is incorrect, but you're actually addressing issues such as this source system treats something one way, but this other source system treats it differently. But we wanted to be consistent. So, it is not so much whether your data is right or wrong. It is whether you can get it consistent across the organization, and I think that's something that is easy to under-estimate unless you've actually pulled your data into one single model.

Mr. Raden: Here is what we would like to do. We can give you a live demonstration of some of these capabilities. I think that will be useful because it is nice to talk about these concepts, but it is even nicer to actually see something to get a sense of what it is that we're talking about. We're not going to show you all the ugly, dirty stuff. We're going to show you the real cool, pretty stuff.

There are a lot of things Kevin said that I wished I'd said myself, but I think there are a couple things that he didn't mention that I'd like to bring up. If you attempt to do an actuarial data warehouse like we're discussing here, it is going to be hard to get organizational support for a lot of different reasons. There is going to be a great deal of professional jealousy. I think that's number one. And second, data warehousing, as a practice, is a very political thing to do anyway because it crosses over so many boundaries in any organization and for an actuarial department even more so.

The other problem is that no one else really understands the depth and breadth of the analysis that you do with your data. IT people, marketing people, and salespeople understand a little bit of slice-and-dice, but they really don't understand how complex some of your models are. The problem is that you guys are so smart, and you've been working with this so long, that it is so obvious to you how these models work that you fail to communicate to people how complicated they are. So, they scratch their head and wonder why the actuaries take so long to do things or why they do things that people don't understand.

The other problem is that data warehousing as a discipline is relatively mature now. There are a lot of so-called best practices, but most of them don't apply to you. The reason they don't apply to you is because most of the analytics and the tools out there are very, very thin, and they simply can't do most of what you need to do. You've heard those names: OLAP tools, query and analysis tools, multidimensional databases, and so forth. So, you're going to have real problems making people understand your specific requirements and why you need to do things a little bit differently.

The other thing you need to understand is the massive scale of data required when building a data warehouse. IOF has under a million certificates, so it is not a huge company. Even with a company under a million certificates, you have a problem of relatively massive scale, pooling data from 15 or 20 source systems, looking at data at the coverage level, and doing valuations on a quarterly basis on a seriatim level. It is not something you can do in Excel and Access. It takes you into the realm of real data processing. Now, if you're a company with 5 million, 10 million, 50 million, or 100 million certificates, all I can tell you is that scalability is not purely linear in this business; that you hit the wall at certain levels, and things get very, very complex at certain levels. Those are things that you're going to have to consider.

Mr. Pledge: Neil, just one other thing. When you're sizing up your data warehouse and considering it, you need to consider that you're not just loading active records. You'll also be loading records on certificates that have lapsed. We load everything that was issued in the past six years. So, our load is well over one million records each time we do a load every month.

Mr. Raden: You're right. There are a couple other things Kevin didn't tell you. First, I don't think we really talked about this ETL tool very much. There's too much work to do in this kind of data warehouse for people to actually have to do anything manually. We can tell you that in this environment, there is virtually no programming. You could search high and low in this entire project to find a line of code anywhere. We don't write scripts. We don't write programs. The only programs I know of, and correct me if I'm wrong, are some light COBOL programs to extract some data in a simple format from the feeder systems. The idea is to take those transformation rules, those business rules, and to remove them from program code. We implement that transformation and that definition of data purely in an open metadata repository.

Now, that may sound like a lot of gobbledygook to you, but what that means is that in a GUI environment, in a point-and-click way, we say this piece of data from this place goes to that place. The rules that apply to how that data moves can become very complicated. There is no programming. If any of those schema change at one end or the other, the system will reconfigure itself and rewrite its code at the time it executes. The other thing the transformation system will do is actually run the process for you. You don't need an operator sitting there moving files, copying files from one system to the other, firing off jobs, writing JCL, writing scripts, submitting batch jobs, or running loaders on databases. The whole idea of the ETL tool is not only to map the data, but to manage the entire process and to manage it in a scalable and maintainable way. You can't expect, as time goes on, to maintain an army of programmers or developers in your organization to keep this running. It has to run on its own. That's something I don't think we covered.

Another thing is that this project was actually implemented with 6 to 8 people, not 50 or 100 people. It was a very small group of dedicated people composed of a nice blending of consultants and actuaries. Another thing that I think that Kevin really didn't stress enough is that I've seen a complete transformation in the department in the two years I've been involved with IOF. Two years ago I think that the people in the department looked with a great deal of dread and loathing at the end of the year, knowing they were looking at a lot of sleepless nights and a lot of grueling work to get through year-end. They viewed their job as being an actuary or an actuarial student: one person doing the valuation for variable life, and another person doing annuities. This view has completely disappeared. Now, you have an organization of people that have a lot of pride. They have improved their skills. They really understand how to collaborate with each other, how to lift each other up, how to train each other, how to hand off to each other, and how to get the job done. It is really a wonderful, wonderful thing to see. And in my position it is nice because I don't get to see it that often. I hate to tell you the truth, but the truth is that very few data warehouse projects ever get to that point because they don't have the kind of visionary leadership needed from the beginning. That is absolutely the most important thing. Somebody has to be keeping their eye on the ball from day one and not get all tangled up in technology and politics and so forth.

They have to say, "If we're going to be successful, we have to transform the way we work. We have to create an environment for people where they can expand their skills and they can rise to the level that they need to be at." I think that's very, very important.

I think that another thing we didn't mention is that IOF has a single system that's designed to process and support valuation in a statutory reporting process for U.S., Canadian, and U.K. business. I think Kevin was being too modest when he didn't mention that. One last point I'd like to make is that the data warehouse is not about automation because automation is really just a way of getting a machine to do the same dumb things you have people doing. The idea is, you automate processes that can be automated, but the most important thing is to create information-enhanced processes. That means that in the process of replacing human processes by a machine, you create something that's bigger than the process you're replacing. You create something that has leverage that can lift you beyond the original aim.

Regulatory filing can be a very simple process because it really becomes an adjunct of the whole process. In other words, now that we've become good stewards of this information and we have a manageable process, we can generate the statutory stuff more or less mechanically. It doesn't change that much from year to year. Not only can we generate it, but it is traceable and reproducible because all of our data, models, and assumptions are stored and maintained as part of this process and they're instantly recallable.

The system we've built can help you better understand your true sources of profit. I think that it is fair to say that even at IOF, they haven't really fully tackled this problem. But I think they're in a much better position to understand it than they were before. I think it is clear that, whenever you look at any organization, whether it is a bank, an insurance company, or a mutual fund, a great deal of the systems and data work that you do is driven by the need to fulfill statutory and regulatory requirements. At an insurance company, and I know that I'm preaching to the choir, it's really about solvency, not about profitability. What kind of prescriptive information can you get from that? Does that really tell you how to run your business? Does that really tell you that you're making a profit? Or does that just mean that you're fiddling around with reserves one way or the other? The point is with this process you can separate the data from its intended use. You can fulfill the regulatory requirements, but you can also get the information-enhanced process by using that data to really analyze at a low level what is going on in your business.

Mr. Pledge: I want to mention one other thing that we skimmed over. We don't just load back a reserve number. We load back the components of the reserves. We load back reserves and the cash flows that contribute to them under different scenarios. And we can do the analysis around that.
So are you ready to take on a data warehouse? This goes back to one of the first points that Neil made. If you just want to extract data, have a smooth process, and produce a report, you're probably looking at the wrong solution. You have to

want more from it than just a process to manage your data and produce a report. You have to want to work interactively with your data with the confidence that it is going to be correct and to compare across scenarios and time periods and really get into it.

Mr. Raden: Do you want to share information? Well, sharing is an interesting idea that presumes you have someone to share it with. It has to be a two-way street. I think that in most organizations that's a slow process. I think that you start by getting people's attention, and once you get their attention, if you're lucky, you can get their permission to really begin to talk to them about important issues. Once you get their permission, then some learning takes place between the two parties, and eventually some trust will build. This is something you have to pay attention to though. You can't just shove this in people's faces and say, "Look, here it is. You can have this data if you want it." You have to work in a cooperative way with people.

The support of senior management is key. How do you get senior management support? Senior management is good at giving you a lot of verbal support at the beginning of any project, particularly one that requires the approval of the board of directors, but they're also remarkably good at disappearing until the project is a success. You really need senior management to be committed to this, that is part of the vision, that has bought into not just a technology project, but has bought into the whole transformation process. We already talked about support from the IT department. I want to leave you with a sense of hope about IT. I think that IT itself is rapidly transforming. They've been clobbered by a lot of things in the last five years, and they're not finished yet because e-commerce is coming, and they're not ready for that either. They're beginning to understand that it is no longer important for IT to be aligned with the strategy of the business. IT has to be part of the strategy of the business. IT <u>is</u> the business, especially in your kind of companies.

Finally, are you ready to address data quality? Data quality is like making sausage or violin strings. It is something you don't want to look at too closely because it is ugly. It is not something you really want to talk about. Data quality is a real mess, and the reason it is a mess is because data may be consistent in the uses to which it is applied, but the minute you try to marry data from different sources you have timing differences, interpretation differences, and domain problems. Then data quality becomes a giant problem. You simply can't solve your data quality problems by gathering the data and fixing it. It usually requires a back-flush procedure where you work cooperatively with the people who are in charge of those systems of record to clean up their problems. That's going to make you very, very unpopular because they don't want to know just how dirty their data is. So, you have a lot of tough work to do here, but the good news is in the end it is really worth it if you keep your eye on the ball and you have some vision.

From the Floor: I look at data separated into two different types of data. One is the transaction data such as premiums or claims that happen during a period. The other is a status type of data that shows what the information is at the end or the beginning of the year. I've always had problems trying to rectify differences. For example, if you have something at the end of the year and the beginning of the year, you should be able to figure out what transactions happened during the year to get you there.

Mr. Raden: Wouldn't that be nice?

From the Floor: But that doesn't always help. Kevin, you kind of alluded to a problem like this when you talked about how complicated the movement star is and I was wondering if you could elaborate on that complication.

Mr. Pledge: We focused initially on the point-in-time view of the data and then reconciling between two periods for that point in time, and we took nothing as assumed when we set up the first movement analysis star. One breakthrough for us was the new status definition that allowed us to track changes in policies instead of seeing a new policy being issued to reflect the change. When we built the movement analysis we discovered things in our data. We took nothing as assumed. We start off with the previous load and end up with a new load. There were no balancing items. In the process, we discovered that a few certificates have been issued on policies that no longer exist. It is really a question of building each item that changes and then tying them together at the end.

I have some limited data to show you on my laptop from our data warehouse in the first release. I'm not going to go into a great deal of the underlying data because a lot of it relates to our business, and it is not something we want to show. Everybody in the department has a basic screen that is available to them. It goes to the life database. They can choose the time period at which they do the query.

Mr. Raden: Kevin, is a subtext of what you're saying that the data you're going to present is correct?

Mr. Pledge: The data is correct. It has been cleaned. Our first release was rather limited in a small subset. The thing to consider and one of the questions we asked is, "Why can't you just do this with Access and Microsoft Excel?" I've been building pivot tables for years in Excel. And the answer is it is just not powerful enough. It just can't handle that volume of data or the complexity that we need to build into this. So the goal is to build something that looks like a set of pivot tables and maybe embed them in reports and provide the ability to drill down. A quick and easy example should be the movement analysis. We're going against 1.5 million records on this particular example. The first prototype just looks at one table. Release 2, which is out this week, actually took two weeks to build after this. It takes three independent queries. They could be against standard tables. Then we could build another query against them. We could look at a particular country such as the U.S. or Canada for male lives for two years. We just put the dimensions into drop-down boxes to make it simple. If we wait 10 to 20 seconds, which is better than an overnight run, a new set of results comes back on the screen. Now what I'm doing could be done on Excel with a particular summary data set, but the data warehouse does it in that 10 to 20 second response time with all the underlying data behind it and more besides.

We've also built other graphic interfaces. We can look at a product summary or geography summary with just a few clicks. And then we also have a function to show the detail reports that will show gender and other breakdowns. We can show gender splits by product line, by state, by anything we define in the data.

Mr. Raden: Kevin, one thing I think you need to point out is that these aren't extracts that you're looking at. If there were 10 million records in the system, it would be querying those 10 million records at the time you hit the key. So, this isn't a series of reports that have been pregenerated and downloaded to a PC. The point is that if you had a much larger data warehouse, the architecture itself can scale. We add more processors. We add more powerful database servers and so forth so that the response time stays the same. It scales linearly no matter how large your organization is. That's the whole point here because when you subset, preaggregate, and compartmentalize things you immediately limit all of the kind of analysis that you can do. You want to be able to have access to all of the constraints in all of those dimensions to frame your analysis.

Mr. Pledge: I guess the other thing to point out is we've introduced this so that we have a very linear learning curve for people new to using the system. So, this is all sort of point-and-click, the Fisher-Price approach. But within the menu, someone who is more ambitious can go straight into the pivot table view of the results and click or drill anywhere. If you have the dimensionality there, you can add it. The next level is where people start designing the queries themselves against the existing schemas. The next level up from this is where somebody would then start designing a query based on other queries that have been built. And then the next stage in the transition is where they actually start designing the star schema that they're going against. They say that the current schema is not sufficient for us. I need to define a new one. They'll build an application and it will be available to everyone. Then someone else will come along and say I need a bit more information. I want to add an extra field. They can go back to the data that's being pulled in and put in more data and really work on it like that. It is sort of an iterative learning process.

Movement analysis is one of my favorite applications. This is interesting because we start off with the certificate count at the start of the time period. We then bring in the new certificates that have been issued and then the certificates that have been revived. That's often a common source of error in a movement analysis. We did the extract last year, and it was down as lapse, but then back premiums were paid. We also have a line for increases. That doesn't change the count, but it'll change the benefit and the fund value. And then we have two lines representing exits, from in-force and new business. We could go into this query and just group the two lines, but we're still in a development stage. So, we wanted to break it out. And then finally we have decreases to balance the increases and then the records at the end. Each line is derived independently from a separate query. Then we just have the error line at the end, which thankfully is zero.

So, the measures we have in our movement report are certificate count, death benefit, and fund value. We've added reserves onto the certificate fact in our second release and can now also drop this into the movement report. One of the data errors that is quite apparent is we have a fund value in the U.S. that was there at the start, and it exited. That's something we would go in and investigate, and, again, for payout annuities we had a fund value that shouldn't be there; we investigate it, and we can explain it. And, again, we can just switch between the product line or select all product lines, and the results are pretty much instantaneous. If we discovered a serious error and we had to go back and load the warehouse, we would do that, and they would change.

One nice feature of the system that we're using here is that you can disconnect. I've done it for my laptop, so when I reconnect my laptop to the server, if the underlying data has changed, I'll get a warning and a question: "Do I want to refresh my query?" I can work on it and plug it back in. If any changes have been made to the source, I will be asked if I want to update my results.

From the Floor: What price do you pay on a system like this, the hardware, software, tools?

Mr. Raden: It is really a tough question, but to do anything of any value you're looking at seven figures. This isn't something you can do for $500,000. If you had the right infrastructure in place and you already had the tools and someone had already addressed some substantial problems with data quality, then it could conceivably be done for hundreds of thousands of dollars. The other thing is that it is not linear. It might cost a couple million dollars for $500,000 certificates, and it might cost a couple million dollars for 10 million certificates. It is not just the size of the company. It is really the complexity.

From the Floor: You keep going up. How far down in size would make sense?

Mr. Raden: In terms of the size of the company?

From the Floor: Yes.

Mr. Raden: Well, that's a good question. I'll stick with what I said. Depending on how you count the cost, I think you would be hard-pressed to develop anything of value for less than some hundreds of thousands of dollars.

Mr. Pledge: It is going to depend on what you've done in the past and how you've approached problems. The software we've used, the database itself, is a database that runs right on a Windows NT server, so the server itself is far cheaper than, say, a Unix server. We spend, I think, $60,000 on an NT server, and now we have two NT servers. So, that's about $120,000 in hardware. The software, well, the price tag on the database itself is higher than, say, something you would run on Unix. So, we paid $78,000 for the database engine. The transformation tools run around $100,000. And the front piece depends on how you buy it. You can buy it per user at about $1,000 dollars per person or you could buy a server version for about $30,000. You might want to buy two different pieces of software depending on the analysis you want to do. That's really the bottom end of software.

Mr. Raden: There is a bottom end though. The bottom end is you can buy a low-end Windows NT server for a few thousand dollars with a copy of SQL Server 7.0 on it. Microsoft offers rudimentary tools that fill all those categories, but they are rudimentary. They have an extract transformation and load tool. They have an analytical tool. They obviously have the database itself. To the extent that you can fit into that Microsoft environment, your scale and the kind of functionality you're looking for, then you could conceivably be looking at a hardware investment of under $10,000. That might take you up to some hundreds of thousands or even millions of records depending on what it is you're trying to do. But it is not just the hardware and software. There is still a great deal of work to do.

Mr. Pledge: The new Microsoft SQL Server 7.0 is I think somewhere around $10,000, and that has every component. It has the data transformation loading. It has the storage and the analytics at the front end. That wasn't available when we started this project, but I don't think it would have been sufficiently scalable in hindsight. But the real work is the analysis and the work in developing it. The software is a small part of the total cost. If you're approaching it for the first time and some of the work has already been done and people understand they have clear data definitions and if you're dealing with one source system, then the cost comes down a lot. But it is really the work that people put into it that makes up most of the cost.

Mr. Pledge: We're not here to sell a particular product, but the approach of data warehousing is something I've been really turned on to, and I think it is something worth investigating.

Mr. Raden: What was presented here presumes that one or more production systems are serving as the source data systems to feed data to the data warehouse. Is it possible to build a data warehouse to be used by users for analytics but also as the central data repository to feed production systems? I'm a little confused about the terminology because to me the production systems aren't necessarily fed data. They're the ones that create the data. So, can you clarify the question for me?

From the Floor: Suppose you build a data warehouse, and now we're going to issue new business. Can you take the new business data, put it in the data warehouse, and when the production system needs it, go to that data warehouse to get it?

Mr. Raden: Give me a specific case so I understand what you mean by the terms.

From the Floor: Let's say you're calculating a cash value and it needs an interest rate.

Mr. Raden: OK. I think the question really boils down to this. Is a data warehouse a one-way street where data is pulled from operational systems and then used for analytical work but not fed back or made available to other systems? The answer is typically yes. That's the way data warehouses are built, and that's a problem; that's what I meant about the end of data warehousing in closing the loop because as a design that simply isn't going to work anymore. We have clients where certain tables we maintain in the data warehouse have become the system of record for the company. One cosmetics company has 2 million product codes. The changes to that product code master happen in the data warehouse, not in the manufacturing system or the logistics system. So the answer to your question is, yes, it is possible to do that, but most of the methodologies and the best practices that you'll read about in the books and hear about at the data warehousing conferences won't ever address that.

Mr. Pledge: I actually have a couple of words of concern about that. Don't take on a data warehouse project to solve another problem like data quality. Data quality problems or other problems in legacy systems have to be solved. But I think you have to address that problem as its own problem if those are real concerns. That being said, we can use our storage to feed other systems outside of the actuarial function. That's a real possibility. We can feed some of the components back and reuse some of the calculation routines in the source systems.