RECORD, Volume 23, No. 3*

Washington Annual Meeting
October 26–29, 1997

Session 71PD
The Knowledge Discovery Process: Mining the Data Warehouse

Track: Computer Science
Key words: Computer Systems

Moderator: MICHAEL F. DAVLIN
Panelists: HERBERT A. EDELSTEIN†
DAVID P. MARIANI‡
J. BRADLEY MURRAY
Recorder: MICHAEL F. DAVLIN

Summary: Data mining deals with analyzing large databases to identify possible relationships, interdependencies, and other useful information. Its technologies and techniques can profitably be applied to a variety of problems encountered by insurance companies and actuaries. These include identifying customer behavior patterns, enhancing the effectiveness of target marketing, and improving internal operations.

Mr. Michael F. Davlin: We're fortunate to have three very knowledgeable speakers in the area of data mining. Herb Edelstein is an expert in data mining and data warehouses. If there is anything related to the knowledge discovery process, I think Herb has done it. Brad Murray is from Tillinghast, and he's going to describe some of the work he has done with existing clients. He will speak more from a user perspective than from a technology perspective on implementing some of the ideas that you'll hear from Herb and Dave.

Dave Mariani is our third speaker. Dave and I go back to the late 1980s when we developed data mining Online Analytical Processing (OLAP) applications for banks

---

†Mr. Edelstein, not a member of the sponsoring organizations, is President of Two Crows in Potomac, MD.

‡Mr. Mariani, not a member of the sponsoring organizations, is Chairman, and Chief Technology Officer of MineShare in Santa Monica, CA.

and savings and loans. You often hear about actuaries branching out into other areas, but you rarely hear of any concrete examples. There is an FSA who Dave and I used to work with in California named Peter Garretson. He worked for us back when we did the savings and loan work. Peter branched off with Dave a couple of years ago and founded a company called MineShare, which has a new relational OLAP data mining suite of products, which Dave will describe to you. It's refreshing to see somebody in our profession go out and make a real mark in a completely different domain.

Mr. David P. Mariani: I would like to give you a little bit of perspective about OLAP. I'll discuss what the technology is, how you can use it for your organization, and, in particular, how to use the right tool for your organization, as I am very familiar with OLAP. After reading all the different articles out there, you can become very confused because you hear all these different and confusing acronyms: OLAP, Desktop Online Analytical Processing (DOLAP), Relational Online Analytical Processing (ROLAP), and Multidimensional Online Analytical Processing (MOLAP). I'll try to give you a little bit of perspective about what the different architectural styles are and what the trade-offs are, from the standpoint of what your needs are and what architecture would best fit your organization.

First I'll discuss OLAP. Why is this technology applicable for your organization? Second, we'll go over the different OLAP architectures. I'll try to describe a little bit of what those acronyms mean and how they apply to you. Next, I'll give some guidelines about how to choose the right tool. It's a short checklist of things you need to take into account when you choose your tool.

After working with financial services corporations, primarily banks, I've experienced a lot of dos and don'ts. Unfortunately it's the don'ts that I've experienced firsthand. I hope I can share what to do what and not to do to make your project successful. Then I'll give a brief demonstration to show you the technology and what OLAP really is.

What we found in implementing reporting systems and client server systems for many companies is that structured and banded report writers are very powerful. Crystal reports, such as business objects and the like, are powerful tools. But what I found in my experiences was that they are too complicated for the end user. Report writers have to manipulate data and know what the data structures look like. You need to know what the tables are, how to join the tables, and how to group the data to produce meaningful reports. OLAP simplifies the report process and provides the user with the components they need without having to know the entire system.

Also, it's very easy to spot trends in data because you can quickly reorganize data. For example, you can quickly intermingle a dimension, like geography or location, or time (months, years, quarters, etc.), or product those to find out how Coke versus Diet Coke is doing in the eastern United States versus the western United States this month versus last month. You can answer that question quickly without having to know anything about programming. I call this a speed-of-thought query. OLAP gives you the ability to drill down and get results quickly. What I mean by speed-of-thought is the ability to be able to answer or ask your next question just as your last question gets answered. It's a free-thinking, free-flowing thought that lets you explore a particular path and find out the scope of your data.

In terms of the architectures, there are three main types of OLAP.

The term OLAP, was coined by Arbor Software. It was a marketing ploy, playing off OLTP which stands for Online Transaction Processing, which was used for handling large transactions like those at your bank. It was coined because there was a need for analytical people like us.

OLAP, the whole tool space, was created was because relational databases were not geared toward decision support queries. I'll explain that after I describe what MOLAP is. MOLAP is the first category that was created. Another technology is desktop OLAP or DOLAP. Relational OLAP, or ROLAP, is the latest technology. There's actually one more called HOLAP, which is hybrid OLAP, but it is not well-defined yet, so I won't cover it right now. MOLAP is defined as proprietary data storage.

Again, OLAP was created because relational databases were geared toward transaction processing. When you make an airline reservation, the transaction is typically an OLTP transaction, where someone has to look up your customer record, and reservation, and pull it back quickly. In contrast, if you're looking at the sales of your products over time within a location, you will have to aggregate or roll up a lot of data into just a few lines. That touches many rows in the database. Traditionally relational databases were not good at that kind of activity.

What the MOLAP vendors did is create their own proprietary data storage. Usually what this looks like is a set of arrays or vectors that are physically stored that way on disk. I'll go into greater detail in a moment.

They also precalculate the data, so when you have to add many rows to crunch down the fact that you want to know about your Diet Coke versus Coke for the eastern versus western region, the query is actually precalculated. By the time you ask for it, it's just a matter of getting those records and returning them to you, the

user. It's very fast. Some of the vendors who are using this kind of technology are Arbor Software, which was one of the first vendors, as well as Pilot Software, which uses a MOLAP type of architecture.

DOLAP is defined by a relational database store. DOLAP vendors will use a relational database for access. However they will create the cube or these specialized structures on the desktop, on the fly. Imagine taking the data in the raw form from the relational database to the desktop, either in a local cache or memory. They'll construct these queries, so the user can quickly drill down and look at the data.

ROLAP also uses a relational data store. The real distinction is that the queries are calculated on the fly. Rather than constructing the cube in memory or on the local machine, ROLAP will sum up all those records in large tables on the fly and return the answer set back to the user. Vendors that are employing this kind of architecture are MicroStrategy and MineShare.

What are some of the pros and cons of these different architectures? As in everything, there are always trade-offs, and this is where you'll have to choose functionality over speed over the number of users. I'll get into some of those trade-offs shortly. One of the defining characteristics of MOLAP is fast response. A fast response is as quick as snapping your fingers. Such a response occurs when you do a query. Again, that's because everything has been precalculated. The data already has been stored in a form requested by the user, so it's very quick to access. Also, MOLAP tends to allow you to read and write from it. This is very important in budgeting or modeling applications. What many users like Arbor do with MOLAP is use it as a collection methodology to collect budgeting information. The people out in the field input their information in an easy-to-understand interface that gets stored back to a specialized database or a MOLAP database. These tools are also very easy to use. They feature Graphic User Interface (GUI) front ends that offer a short learning curve.

DOLAP gives you full access to the relational data. That's good because you have access to any data that you have in a relational database, which is typically much of the corporate data or the information in your data warehouse. That's in contrast to MOLAP, where you need to carefully choose what data goes into the separate data store. It's also relatively inexpensive because in DOLAP, most of the work happens on the client side, so you don't need to actually establish a server and have it in a tree type of architecture. Really it's just the front end talking to the relational database, so there's no need for an intermediate server to get your results back. The desktop allows fast response. Since you're building these cubes on the desktop in memory or in a local cache, the navigation of that data is fairly quick.

Again, ROLAP allows full access to the relational data. It also is good for handling larger quantities of data. Typically, ROLAP tools are being chosen over the former tools in retail situations. You have many transactions. Take a Wal-Mart situation, where every single sale that hits the register is recorded. That goes back to a centralized database; that data needs to be mined and accessed. It can't be reconfigured and put into a proprietary database format because it's too big, so ROLAP tools are well-suited for a large amount of data. It also leverages your relational database architecture, which means you can take advantage of your database architecture that are on-site, such as your investment in the hardware and these relational databases.

While MOLAP gives you the fast response, the rewrite access, and the ease of use, on the negative side, you have very slow load times. That's because you have to precalculate everything. In a typical multidimensional OLAP environment, you have quick data expansion rates. Imagine if you're calculating a multidimensional array using the three dimensions I was talking about before (time, location, and product). We'll take the number of products, times the number of time dimensions (number of months), times the number of geographical locations or branches. There is suddenly a big array.

It takes a long time to load these databases. It also requires special skills to manage them. It's not easy to find a database administrator to manage a MOLAP database. It'll be much easier to find an oracle or a side-based. Also, the data must be kept up-to-date. MOLAP is using a different data store, so you need to synchronize that because now it's outside your normal warehouse realm. For DOLAP, these tools tend to be difficult to deploy. This is the whole fat client problem. You need to get your database drivers on these client machines. You need to have a lot of RAM on these machines. These machines tend to be unwieldy, which is not good for handling large databases. Just think about your PCS RAM and disk space. At some point, these models begin to break down as you throw large quantities of data at them.

ROLAP tends to be very expensive to deploy. If you look at some of the implementations that are done with ROLAP, they tend to go into the millions of dollars because you have to pay consultants a lot of money to configure your tables for relational OLAP. Then you have to define the Medidata. It's a costly and lengthy process. It's also slow for really large amounts of data because you have to perform these queries on the fly; you cannot take advantage of the precalculated queries. It also tends to place a heavy load on the server. If you have many users, they will be doing what relational databases find difficult—summing up rows and doing lots of group buys. This can lead to unpredictable performance. The more

data you have, the slower the database operates. Unfortunately, you cannot predict how much data each user is going to manipulate.

When choosing the right tools, there are some questions to ask yourself, given these trade-offs. How sophisticated are your end users? How much can they do by themselves in terms of report writing? You want to opt for an ease-of-use tool if your end users are uncomfortable doing it themselves. How much data are you dealing with? What's the data volume? What's the complexity of data? How deep are the data dimensions? How many users are using the system? How often does the data change? If you're in a daily change, and you want to get the latest data, you need to take that into consideration with the architecture deploy. How long is your update window? Again, how fast do your users need to see the next or refreshed data? How many dimensions do you have? How deep are those dimensions and how many elements are in those dimensions? How many cost centers are there? How many time periods do you have? How many products do you have? These factors will slow down your calculation times if you choose a MOLAP architecture. Look at the different factors—complexity, number of users, the amount of data, the frequency of change, the length of your update window, and the number of dimensions—to consider which tool to choose.

MOLAP supports a high level of complexity. MOLAP tools are feature-rich. DOLAP tools, being less geared toward report writing, are less so; whereas ROLAP tools are a hybrid of MOLAP and DOLAP tools. With ROLAP, some tools are feature-rich, but other tools allow you to access DOLAP's relational database capabilities, which enable you to navigate all the way down to the lowest detail level of your data. In MOLAP, you're dealing with a separate data store, so there's a disconnection between that database and the rest of your warehouse.

In terms of number of users, MOLAP systems tend to be scalable because the data are stored in a form that is easily accessible; those accesses are very fast, meaning they can score a larger number of users. DOLAP and ROLAP, on the other hand, tend to be driven by the relational database, and the types of queries you're doing can severely affect your performance. In terms of the amount of the data, this is where MOLAP and DOLAP lose efficiency. As you deal with larger and larger volumes of data, the time that it takes to calculate the models versus the times it takes for DOLAP to calculate or form that cube on the desktop becomes unreasonable and unbearable.

One client of ours had 150 meg worth of data that they loaded into a MOLAP tool; MOLAP expanded that 150 meg of data into 6 to 7 gigabytes when it precalculated it. Furthermore, it took six to seven hours to calculate or create that database. That was a special case because they had many dimensions. There were actually only

four dimensions at that point. They wanted to do 17 dimensions because their trees, or their hierarchies, were very deep and had many elements. That exploded the size of the database. These are all considerations to think of when you're thinking not only about the amount of data, but how your data looks as well. It's not always a clear-cut decision.

Frequency of change is an issue. In the case of that one client I was talking about, it was six or seven hours, and they wanted to turn that process around in an evening so it would be available for the users the next day. In that case, MOLAP wasn't applicable for them. A DOLAP or ROLAP architecture, where the data is already in a relational database, is already easy to update and to load. It will be ready a lot sooner. Again, the update window is closely tied to the frequency of change. You need to be able to quickly update it and, again, DOLAP and especially ROLAP are good for this. Finally, there are a number of dimensions that are a factor in the expansion side of the MOLAP databases. DOLAP actually falls into this category as well, whereby the number of dimensions grow, again, building the cube. Precalculating all the data becomes more and more difficult.

While implementing these tools, I've learned certain guidelines. Make sure that you have successful implementation if you choose this route. You should solve a real-world problem. Get users involved to solve a problem; do not just try to employ the technology because of its initial appeal. What I mean by this is, I've been involved in a lot of processes where the IT shops led the purchase of these tools. We build a tool and the users end up using it. It never happens that way. Start with a small application.

One client I worked with, a large regional bank, decided to solve the problems of the corporation. They were very decentralized. They included all the regions, of which there were 10 or 20. Suddenly, everyone had to have his or her say. They added to the list of requirements and they ended up trying to build the Taj Mahal, but nothing was ever built. So I would recommend starting with a small application, something that will make your OLAP vendors prove to you that their tool will work. Make them implement a prototype. Most of the vendors will do so if asked. Before you buy the tool, perform a test drive. You would do that if you were buying a car, so you should do the same if you're buying OLAP. Also make sure that you assemble a team that has both representatives from IT and the end user community because end users really need to drive the functionality needs of the system and the IT, or technology guys, need to make sure that this is a stable platform that's going to be scalable. It must be one that can support a large number of users in the future. Plan for several iterations as well.

Again, this is the quick-strike, easy-chunk-to-bite-off scenario. If you're successful with that first scenario, other questions will arise, and you want to expand the use of the tools. You need to be able to change and modify the tools and to show end users real progress.

Finally, roll out the system slowly. Another problem with that one regional bank I identified earlier was that they planned to roll out the system almost simultaneously to hundreds of users. That was fraught with danger because you don't know if the product is going to sell or the problems you're going to encounter. Start with five users. When you're successful with five users, go to ten users, and then you can scale up from there, but start small.

One of my clients is a very successful eastern money center bank. The IT staff decided that they were going to build the data warehouse for the corporation. They started this multi-year project that cost millions of dollars. It never saw the light of day because it was driven by an end user. It wasn't driven by any real needs, so it ended up placing or collecting in a relational database tables and data that nobody really wanted or knew how to access; thus, it was a big waste of time and money.

Make sure that you don't set unrealistic expectations. Don't say that this is going to solve the world's problems, because it won't. Solve a small problem first and then use the technology further once you become familiar with it. Make sure you do not ignore the functionality needs of the tool.

When you're looking at these tools, there's a lot of difference between the built-in features and functions in one tool versus the other. Make sure that you prototype the tool, and make sure that it can handle all the kinds of functions that you need. Otherwise, you'll have a great technology, but no one will use it. Don't let IT drive the project. Again, you need a real team that is working together on this.

Mr. Davlin: Dave talked about OLAP, which is a great way of organizing your data in a multidimensional or hierarchical way, which I'm sure is the way you naturally do it. It doesn't matter whether those data are coming out of your modeling systems, your valuation data, or your budget reporting data. It's great for filters, or getting reports, but eventually you'll hit a point where you're going to want to do some heavy-duty statistical or other forms of analysis on your data. This is what Herb is going to talk about next.

From the Floor: You mentioned large end users and a large database a couple times. Can you give me an idea of exactly what you mean by large? Are you talking about millions of records or thousands of records? At what point do you cross over into the large domain?

Mr. Mariani: Yes, when I say large, it's a cop-out because what is large? Large usually pertains to the number of rows. Many people will use gigabytes to define the size of the database. I think that's misleading because what really matters is the complexity of the data. How many dimensions are there and how deep are they? In the example, 150 megabytes blew up into 6 to 7 gigabytes. That's not a lot of data to start with, but it did blow up into a large database. So there's really no easy answer to that other than to say what Arbor would say: they can handle databases up to 50 gigabytes after the cube is calculated. You have to move backwards from 50 gigabytes to figure out how much source data would result in 50 gigabytes. That's typically the rule of thumb that they use.

Mr. Herbert A. Edelstein: I'm going to talk about knowledge discovery and data mining. Let me ask a couple of questions. How many people have ever had a statistics course? How many have never had one? How many here have ever built a logistic aggression model? A few of you. How many of you have ever built a neural net model or a decision tree? Why do we want to do data mining? Fundamentally, there is money in data. In some ways, data is more important to an organization than cash.

If I take away all the money that you have, you can get it all back if you have information about your policies and your customers and their practices. But if I go in and wipe out all the information on all your disks, tape backups, and everything else, you're in big trouble. Data are as important to most organizations as actual money. There are many definitions of data mining, and some of you may remember the old definition from graduate school: data mining was when you were having trouble writing your thesis; you got a lot of data and looked for a conclusion.

I define data mining in a very simple fashion. Data mining is a process (and I want to emphasize that it is a process, not a product) for finding patterns and relationships in data. Surrounding data mining is a larger process called knowledge discovery, which is essentially the process of building and implementing a data-mining solution.

We're going to talk now about models. I want to define a model in a particular way. A model is an abstraction of reality. Now you probably understand modeling much better than many of the other groups I talk to. A model is not reality. There's a statistician named George Bachs who I think said something very important. I'm going to paraphrase it because I don't know what the exact quote was: All models are wrong, some models are useful. We're not striving for 100% accuracy. We're trying to devise something useful. There are two kinds of models that we talk about in data mining—predictive models and descriptive models.

A predictive model is one you build explicitly to predict something. Is this person a good risk or not? That's a classification model. What is the return on this class of policy? Those are explicit predictions. Descriptive models are looking at patterns and describing things that you see in the database. When I first thought of descriptive models and I was using this definition for a while, one of the questions that naturally occurred to me was, why would anybody want a descriptive model? I want a descriptive model because I'm going to post it on my wall, and, if it's a particularly naive model, I'll invite people over and say, "Here would you look at this model. Isn't this lovely? See what I built today?" That's not very useful. People want descriptive models because they are used for implicit predictions. There is an implicit assumption that the behavior of your existing data will be reflected in the future and, without having an explicit prediction, you're going to take actions. You're going to do something. What good is a model if you don't do anything with it?

There are three types of predictive models. The first is a classification model in which I'm going to take a look at what's called a categorically variable good risk or bad risk and predict it. Which class do you fall into? With the second predictive model, a regression model, I am going to predict a particular value, typically a numeric continuous value. What is the probability that this person will die within the next 14 months? Probability is continuous value? That is related to regression and time series. A time series model, the third predictive model, is one in which time is one of the independent variables. There are certain properties that I can take advantage of when I'm using time to predict.

Descriptive models are clustering models, and are so common that I don't know how many clusters I have. What I want to figure out is how they all clump together. There are a number of tools I can use to identify clumps in my database. What are the smallest distances, and what groups are there with small distances between the members? In some models, I want to find 10 groups or 20 groups. In other tools, I do not let it limit the number of groups. I then look at these clusters and interpret them. It's often very useful to build a predictive model on a cluster. If you think about it, that makes a lot of sense because if I desegregated my database into clusters, I'd wind up with greater accuracy. Other descriptive models are associations. What events are coupled together? The big driving force behind the development of association models was, in fact, scanner data.

Grocery stores have scanners. A grocery store can build an association model that says, if you buy low-fat cottage cheese, there's a high probability that you will buy skim milk. I can go even further. How many of you have a card from your grocery store that you are encouraged to use every checkout? Many of you. So when you use that card, they can build these association models over time. That's called

sequencing. I can find out how your behaviors are related over time. It is scary how much data we have on people. In a recent issue of *The Washington Post*, there was a letter responding to an article by somebody from the Direct Marketing Association (DMA) saying how good it is that all this information is available. In the letter, the individual firmly stated, "I'd do everything to keep my name off that information because I don't want to be bothered." If you ever went to any of these DMA meetings and look at the things that are available, you would cut up all your credit cards. You would never leave your name.

Data mining is not something to look into if you have the least susceptibility to paranoia.

Now there are many myths about data mining. We're always looking for miracles, and data mining is one of the latest miracles. I think most companies would be better off if the CEOs who flew never read an airline magazine. Some of the senior executives suffer from technology envy.

The first myth is data mining tools need no guidance. I can't tell you how many people believe that if I bring a data mining tool in, I can put it on my database and when something interesting happens or some interesting pattern is observed, it's going to call me up on the phone and say, "Have I got news for you." That's not true. Trying to improve a direct mail campaign is a common usage of data mining. I may want to build a model that improves the response or the return that I get from my mailing. On the other hand, I may want to build a model that increases the average amount that is purchased on each response. These are two very different models. I don't set my data-mining tool loose any more than I would set a person loose and say, "Make it better." Another myth is that data-mining patterns explain behavior. This is a common and pernicious rumor. The fact that you subscribe to a certain set of magazines, that you're a certain age, or that you have a tattoo on an unmentionable part of your anatomy does not cause you to buy anything. What we're looking for are patterns because they may in fact exist. We may uncover a pattern that is positive, but we need to do some other experimentation or work to establish causation. In business we often don't care about causation. Remember, all models are wrong; some models are useful.

Another myth is that data mining eliminates the need to understand your business and your data and that data-mining tools require no data analysis skill. This assumption—"You all understand a lot about data analysis. We don't need you anymore, we went out and bought a data-mining tool. We now have artificial intelligence; we don't need real intelligence"—is utter nonsense. You cannot make effective use of these tools if you don't understand your data or your business. You

need to take advantage of every tool that is available to you in order to do the best possible job. You need to understand both your business and your data.

The last myth is the idea that data-mining tools are in some way different from statistics. They're not. They are the latest iteration of statistics. Back in 1936, R.A. Fisher developed an advocacy analysis that was a laptop computer. Today, the computing power in each one of these laptops is greater than the biggest and most powerful mainframe that existed 25 years ago. Of course, they didn't have to run Windows. It's not surprising that there are a bunch of new techniques that are evolving concurrently, some from the statistics community and some from the machine-learning community. We use the term Artificial Intelligence (AI) because AI is disreputable.

We use the data-mining tools to handle large volumes of data. What's a large volume of data? We hear people talking about terrabyte databases all the time now. Not as many people have them as talk about them. Some joke that the difference between a computer salesperson and a used-car salesperson is that the car salesperson knows when they're lying. Despite what you will hear from some vendors, data mining works well with sampling. Statistical tools can die when the record numbers exceed a few 100,000. Even if you are mining with databases of 75 million rows of customer data, if you're mining that customer data, you don't need to look at every customer. You can take a sample. If you do a sample with 250,000 or 500,000, that's still a lot of data because building a model and using these techniques inductively from the data on a parallel computer is still going to take you hours for a single model. Sampling still works despite the amount of data. We have the ability to handle large amounts of data. We reduce dependence on the model. When you build a logistic regression model, what do you have to do? For example, a company that has 75 million rows of customer data has 3,000 columns for every customer. Can you figure out which columns should be used to build a model out of 3,000 columns?

That's what people do when they do logistic regression. They pick 10 or 15 potential models. They do some transformations to make them win, and they do some other things to handle interactions. Eventually, they come up with a workable model. People who do logistic regression do a good job, but how do you know if the right models have been selected? How do you know if 10 better models exist? Maybe you can sacrifice some accuracy and have only four variables as predictors.

Thus, we reduce some of the dependency, but we're still not going to build models on 3,000 variables. We're still going to pay for our dimensionality, but we don't have to worry about linear assumptions or normal distributions. That was handled in the modeling tool. The emphasis is on exploratory analysis, not confirmatory

analysis. If you build a model, you may want to verify or validate it in a real-world situation. This is new statistical software that takes advantage of the advances in hardware and builds on the statistics and information systems.

This software is also aimed at educating the users. When I started programming, we didn't have a GUI interface. All we had were zeros and ones and sometimes we didn't even have ones. I wrote an entire database program using only zeros. Sometimes, if we didn't have zeros, we had to use the letter "O." What you find is that the statistical guides are mathematically sophisticated and dedicated users, but we often want to extend this capability on problems that may be a little less critical to people who don't have as much training, but understand the business and the domain experts.

The data-mining tools can help us do this. They have well-made GUIs, although you shouldn't make the mistake that a GUI implies ease of use. Some products out there have well-designed drop-down menus. The neural net gives you 18 choices for summing things. They give you 12 choices for transformations, and I can't even define what half those choices are.

We've heard a lot about OLAP. One of the questions I most frequently hear from the information systems community is what's the difference between OLAP and data-mining? OLAP is not data-mining. Remember that the A in OLAP is analytical. In an OLAP environment, you, the user, formulate a hypothesis and do a series of queries. You typically have a database design that helps support this whether it's in a relational database or a multidimensional database while you go through these queries. You are trying to verify the hypothesis in the data. Can you support this hypothesis? You're not asking purely out of interest, although occasionally that happens. The government is working to stop that. The problem is, if you have 3,000 columns of data on every customer, can you formulate all the hypotheses that might possibly explain a behavior that might predict a risk? No way. I don't care how well you know your data and your business. You need a tool. In data mining, instead of formulating the hypothesis and verifying it in the data, it induces the model from the database, searches the data, and inductively comes up with models. You don't have to verify that data supports it. All of that, however, still has an important role to play. Let's say I'm a marketing manager for Harley-Davidson motorcycles, and I have discovered a group we are not selling to. It is a group of people 75 to 85 years old, who have had hip replacement surgery and wear a strange tattoo. All this information is available. If you think video stores are a problem, think about tattoo parlors. Let's say I'm a marketing manager for this. I want to embark on a $10 million ad campaign, but I have to stop and think how many people belong to this group. Let me analyze the size of the group using my OLAP tool. I discover only six people.

I cut my ad budget from $10 million to $2 million. You must understand that in marketing, your importance is directly proportional to your ad budget. We did a report on data-mining tools. The first chapter from the report is called "An Introduction to Data Mining and Knowledge Discovery." The chapter is a semitechnical introduction. Part of our survey asked how data-mining tools were used, and we found that they are overwhelmingly used in marketing applications. They are also used in manufacturing applications. They are not, at this point, widely used in actuarial modeling; insurance companies feel that actuaries are somewhat conservative. I have not had enough experience to verify that, but I'd love to hear from you after this session.

I'm not going to go through these applications in detail because we don't have much time. However, I'd like to point out one interesting issue. It is the fraud detection application. This is a big issue in the insurance industry. People have been known to commit insurance fraud. How do you detect that? What do you do about it? I heard in the past that insurance companies always pay towing claims because it was more expensive to verify that somebody had a towing endorsement than it was to pay the $50 claim. Once people know that, then the proportion of invalid claims goes up, so you have to start checking. How can I detect fraud? You need to understand how sampling works, but sampling becomes more difficult to do because you're looking for infrequent occurrences. The size of the sample needed is inversely proportional to the frequency of the effect you're looking for. You'll need a very large sample that can approach the size of the liability. The entire process starts with a problem statement and people need to learn how to make good problem statements. You collect data. You consolidate and clean the data. You select rows and columns. You prepare the data, and then you build the model.

Experience has shown that somewhere between 50% and 90% of the time is spent crunching the data. In the next presentation, we'll hear about some of the issues of building data warehouses. I wish I could tell you once you set up your data and move to model building it's all downhill, but that's not true. That's why I put some loops in there. In fact, if the first model comes out and you like it, you better get rid of it immediately because it's probably wrong. You're going to have to build a number of models and go through loops, validate, deploy, and put it out in a way so that people can use it. If you come up with ways of rating people, what good is it if you have to be called for every potential policy. You need to figure out an effective employment mechanism. Then, once you deploy a model, you need to monitor it. What if things change? I suspect over the next few years, the increasing efficacy of antibiotics is going to affect some actuarial models; life expectancy, for example.

I like to divide data mining into a hierarchy or taxonomy of elements. The first is the problem statement. What is the problem we're trying to solve, and what are the kinds of models I'm going to build to solve it? For example, churning is a problem. Good customers leave; bad customers stay. What can I do about this? My business problem is keeping good customers. I want to encourage my good customers to stay, so I do a regression to predict profitability. What's the lifetime value of a customer? I may need to do both. A lifetime value not only comprises the revenues that you're going to produce, but also the associated costs. What are my payouts going to be? In this case, I'm going to build it with a neural net and predict with the technology. The algorithm is a backdrop neural net in a particular product. I also want to know who's going to stay and who's going to leave. In that classification I'm going to build a decision tree, using the cart algorithm and a particular implementation of cart.

There are many technologies. The traditional tools are still important, but we have many other tools. Two of the most commonly used are decision trees and neural nets. There are two kinds of decision trees: classification trees and regression trees. A regression tree is used to predict continuous values, and a classification tree is used to specify which category something falls in. This is a method of showing a set of rules, and it has been around for a long time. What's relatively recent is the ability to inductively find the underlying rules that circumscribe what happens to a decision tree. Take a decision tree that's going to predict who's a good and who's a bad credit risk. If your income is greater than $40,000, then that would imply that you are a good credit risk, so we take the yes path. If you have high debt, then we travel down the yes path, that would be a bad risk. When you build a decision tree, you have to look at whether the data are equal to the number of levels in the tree. You can't go below that. You can build trees that are even worse than that.

So how do you decide if income is at the root? Look to see which of the possible predictors is the best. Then, at the next level, determine, of all the possible predictors, what's the best predictor. By the way, when you find a predictor, you then have to look at all the values and judge where the best place is to split. What's the best discriminator? Do this integral until you finally find what you like. Trees offer the advantage of rules that are more or less interpretable. Decision trees are very good discriminators.

Neural nets behave differently. Neurons in neural nets resemble some aspects of the brain. Neurons are arranged in layers. What we're going to do is create a layered representation. We're going to take our input variables, and we have what's called a hidden layer with hidden nodes. At each one of these nodes, we're going to take a weighted sum, transform it, and then go to the next layer, which in this case, is just an output layer. We're going to take a weighted sum again, and

then we're going to see, how good the answer is. In both decision trees and neural nets, when you build a model, you train it. That's the term that comes out of machine learning. If you're a statistician, you estimate it.

I have a set of data in which I know the results. I build this model, and then I validate it against a set of data in which I know the results, which I didn't use in building the model. There are many validation schemes. There's cross validation, boot strapping, and jack knifing. You can use these different validation techniques, depending on the characteristics of the problem.

I'm going to describe a back propagation neural net. I check the difference between what I'm trying to predict and my actual prediction. Then I go back and change the weights for my weighted sum, and I propagate that change back into my hidden layer. I take a look at my error, adjust my weights, and then I feed forward. I check what happens again with the next row, and I keep looping through my data. If I have 50 input variables, 20 variables in my hidden layer, and 20 nodes in my hidden layer, how many weights do I have? I have more than you think because I have an output layer; it would be 20 times 50 or 1,000 weights. It is not surprising that neural nets can be excellent predictors because you have 1,000 parameters that you're using to fit a curve. In fact, you can take a table of random numbers and fit it with a neural net. It has been done.

You put the numbers on a chart and run a neural net. It will draw a curve that hits every one of them. Of course, it doesn't hit the next one. That's what we call overfit. Unlike people, neural nets, decision trees, and models can be overfit.

There are different ways of deploying the solution. You can apply the model to a database, but you're going to get scores. You're going to score the people in the database. No insurance company that has ever rejected every application has ever had a payout.

The point is, who do you want to accept, or can you do scores? There are some pertinent issues in data mining. One of the things about data mines is that actuaries build models and predict things and use variables for predicting those models. I would assume most of the data you use is related to what you're predicting. How many of you use magazines somebody subscribes to for predictions? Not many. You do? It may be that someone who subscribes to a biker magazine is a worse risk than someone who subscribes to *Modern Bride*.

There are many issues like that regarding which variables you include and which variables you exclude. Can you afford to build models on large numbers of variables? There's an interesting court case in California in which a company found

that your driving record is a good predictor of your credit risk. I guess if you're a bad driver, you die before you can pay your debts. The company was sued by somebody who was denied credit. He said that just because he had 682 points didn't mean he didn't pay his debts. He claimed, "I paid all these fines and I still pay my debts." They made an inquiry.

When you are building a system, it is important to remember that you not only have knowledge that you have discovered through data mining, but you also have a lot of domain knowledge. You have history. You don't need data mining to determine that someone who is a pilot for the Blue Angels is not necessarily the best risk for life insurance. Maybe they are, I don't know. You don't need to know that somebody who is skydiving without parachutes is a bad risk (even if that person has a theory about how he holds his body). So you want to incorporate that knowledge and then apply the rules.

When you bring data mining into an organization, you're going to run into some difficulties. I find more problems on the organizational side than on the technological side because many groups are involved. You've got the providers; you've got the folks who build models; you've got the folks who build applications for deploying the models; and you've got the folks who use the models. I could add other groups, such as the people who are paying the bills (and there might even be different bill payers in more groups). So there are many interesting issues. Then you have to consider the skill levels. Who are the people who are going to be using this? Who are the people who are going to be developing the models and building the deployed applications? When we talk about products, it is a mistake to simply say, this is a high-end product or a low-end product. That's too easy. You need to look at what the product does. There are products that have been packaged for particular industries and particular applications within industries. There are also products that build models across a broad spectrum.

There are good products today. This is not a mature market. The products have not reached the level of maturity that database management systems or traditional statistical packages have, but they are good products. I would suggest that you investigate how data-mining products can help you do a better job in meeting your goals and missions.

In conclusion, data are strategic assets and you can use data properly to transform your business. Knowledge discovery is a process, not a product. There is no product that you can buy that will get the job done for you. Similarly, buying a shovel does not dig holes. Intelligent data mining requires a mix of tools and techniques and you need to know both the tools and the business to succeed. If

you do not avail yourself of every available tool, you are at a disadvantage. Conversely, a good tool is worthless if you don't know the business and data.

Mr. Davlin: Brad Murray is going to talk about some of the work he has done with what we'd probably call DOLAP products. Two-and-a-half years ago, I gave a presentation in New Orleans on OLAP and I tried to bring this idea to people's attention, but there wasn't anybody who seemed to be using the technology. I was very excited about six months ago when Brad called and told me he had actually spoken to clients about these things and conducted some pilot experimentation. He's here to discuss his experiences with us.

Mr. J. Bradley Murray: I'm not a technological expert. I come from the plain old actuarial perspective, and I've seen some applications of this kind of technology and work that we do. The emphasis of my talk is going to be from the user side, or more specifically, from the actuary's perspective. Why do we even care about this? I hope you'll gain a greater appreciation of that question as I run through my presentation.

Let's start off with something that we have traditionally looked at—experience studies. What's the big deal about experience studies? We all know that experience studies are used for pricing. We also use them for product development. It all boils down to being a major component in our modeling activity. We all know this, but there's not as much attention given to this, primarily because of all the data issues that we run into when putting together an experience study. This gets down to where technology has come into play, and that is what we've been able to use technology for. We put together slick modeling systems, like TAS or PTS, or one of many projection models.

These models come from the user perspective. We started out by using spreadsheets, and now we have these great modeling systems. Herb touched upon a point that I think addresses a problem with our process: for what we do, these modeling systems are really more of a descriptive model. I think he described that a little further, so I might be misusing the term somewhat. Our modeling systems enable us to understand the relationships between the various products we're trying to look at or understand. But it really doesn't have any predictive value.

The modeling systems give us a process. As we back down into this process, we see that the heart of that predictive value comes more from the standpoint of the data and how we're feeding into that model. Experience studies are a very key component. Experience studies enable us to capture some of the key elements of the data, so that we're able to express that in these models and, in turn, project forward what we think will happen.

I'm going to talk mostly about experience studies in modeling because, as you saw from the previous diagram, data are at the heart of both of these. We know that experience studies are more or less a study of the population, movement characteristics, or behavior of a population, or lapses, mortality, and so forth. Actually, I think the key part of what experience studies do for us is provide a bridge to the reality of the situation we're trying to understand. That's very important. We have these great models, but if we don't have confidence and tie back to the situation that we're trying to understand, then it really makes the models useless. We're in that category that Herb describes as useless.

Let's focus on some of the issues that we faced in doing experience studies or models. I'm going to start with experience studies. It's all based on data. In fact, I think that's one of the reasons that experience studies have moved to the background. They can be quite boring.

Getting clean, consistent data is a big issue when you do an experience study. It's a big process. There must be consistency of that data with other sources, such as reports, annual statements, and financial statements. The data behind the experience studies is the same data that feeds into the model. Many times that's not known.

The next issue is the black-box process. Think of what typically occurs when an experience study system has been set up, especially on the big mainframe systems that Herb described. Those that could do Windows would fill this room. In those mainframe systems we go to those IT mystical people. We tell them what we want, and they set up a nice system with reports for us. More or less, it's a black-box process. Similarly when we start going down the path to build those models, we go to those same IT people and tell them we need similar data to be able to understand what to model. After talking to them, we, on faith, believe that it's all coming from the same source. In fact, we know those data are not all the same. They are overlooked, and that's where I think serious issues arise.

Obtaining clean data is also a problem in the modeling process. In fact, the problem is not only retrieving the data, but getting the data in a usable format. This is a major detail. We must have a good format that we can use. I'm sure some of you have struggled with data where there are some funny-looking zeros and you don't know what to do with them. You spend the next three months banging your head against the wall.

The second issue is consistency of the data. Consistency is very important. This boils down to validation. Are we able to validate the data to show something is modeled on a static basis? Is it representative of what we think we want to look at:

the products we want to look at, the population, the company? Second, are the data somewhat tied to dynamic interactions in the company and are those represented by things such as income statements? Can we relate the experience study data back to that?

The last issue has to do with how you could go through this process and all this work, and then after you set up this thing, suddenly somebody else in another area wants to build a different model. This person either does a different kind of process or does a business planning model versus an embedded model. He may rehash all the same problems all over again. He goes to the black box, but it is not very efficient. Sure enough, technology comes to the rescue in our situation.

I'm going to focus now on the main point of our presentation, which is data warehousing. Let me get at the basic concepts behind data warehousing. I hope that you can appreciate, as Herb discussed, things such as a cluster rate, seeing various relationships, and the distances between groups with similar characteristics. Well, that rings true with what insurance studies are or what we try to do. What we have historically done with data is look at the historical relationships and determine what's happening there. Perhaps you can start to appreciate the connection that we have with this data warehouse approach.

What is a data warehouse? Again, I'm going to start at the base level. Maybe all of you understood OLAP, DOLAP, and MOLAP, so I'll start at the base level. A data warehouse is like a large container where the data resides. It is a data repository. This is integrated stored data. The key word is *integrated*; it's continuously fed. Timing of that data, as far as when you're looking at it, is consistent. You don't have to worry about going to the system people to do one report and then sending somebody else to those same people to do another report. You think they are the same data, but they're not. They don't seem to tie together. The myriad of timing problems are dealt with here.

The field name is a major issue. You may go to your systems people and want one report that says, "This is gross premiums," and then you'll get another report that says, "Gross premiums." You start looking at the two for the same policyholder at that level, but they're completely different. You realize that one includes policy fees and another includes modal loading, and you, stuck in this tangled web, put it in a folder in hopes that the person who wants to do the project forgets about it. You want a single set of naming conventions so that you can ensure that "gross premium" means one thing across the board for whatever data are fed into this container.

The last point is very important in this large process. I think this was addressed earlier. Again, what I'm going to highlight here I hope will illustrate a smaller approach. This data warehouse concept is powerful from a very basic standpoint, which is, you're able to visualize much better how all this data mining and warehousing operates. If you think data warehouse, that's because that's what you're really doing. You're taking your data and you're warehousing it up front. You're taking all the activities that you would do with the data—cleaning, integrating, and understanding the data—and doing it first. Once you have a clear understanding of the limitations and what that data means, and as you start having everything feed into the corporate-wide data warehouse, you can then make predictions from that data.

You want the same container and the same definitions for the field. Whether it's for the financial people, modeling, or experience studies, you're able to look at the same source, and I think that's extremely beneficial over the current process. It's not a black-box process. Everything feeds into this system.

Another major advantage of this process is this consistency in validation I spoke of earlier, which is a critical backbone to everything we do. We do a modeling system or put together a model, but we can't convince others that the model indeed tied back to the rest of the operation of the company. You can say, "This is my model and here's my data. Let me show you how those data tie to the blue book and let me show you how this also ties to your report," and walk confidentially into a given person's office and pull out your model. The data that are supporting it are coming from the same source and have the same definitions.

Another major point of this is that it's not going to solve your problems. If you have bad data, you have bad data. Many people say, "We have bad data. We've spent years on it." What this does for you is create a process around which you can organize your data. You know your limitations. If you have three certain fields that are completely meaningless where they've been organized, you understand it, and you're able to summarize it; you go back to the IT people and the systems and start to make progress on it, as opposed to doing a one-year big project. All these people dedicate a year to the project. They leave it and never want to see it again. Who knows what happens to it?

I'm going to discuss some of the components of this data warehouse or data mart. There is what I call the corporate-wide data warehouse, which might be a store of data across the board for all or many corporate activities. What we're focusing on here is more of the specific application of some of the activities that we do as actuaries. We can benefit from this process.

A database is just a table of data—rows of information and columns of information. The rows, similar to policyholder information, contains descending policy numbers. Columns contain information such as premiums paid to date, volume, coverage amount, and riders; things of that nature would run across the top. That's all a database is.

Relational databases make this a powerful approach. With a relational database, you have two different tables with a common field existing between both of them, which may be a policyholder number. You have common information points from which you're able to pull together the data from all those different systems to do your analysis. It enables you to avoid having to pull together everything into one big database, which is an inefficient process. As I said earlier, data warehousing is one big step ahead of this in this relational database structure, but this I hope will help you to start to picture and understand how these relationships operate and the power behind the data warehouse.

What are some of the advantages of this whole process? One leverage comes with current data sources. There is no need to create more reports or sub bases. First, you're able to take the raw data from the systems in this repository. It's helpful to be able to understand what is out there. Second, you're able to drive your multiple processes from the same source or from the same repository, such as modeling, or experience studies. Third, there is wholesaling of the data functions. As I said earlier, when you do this, you spend six months doing it in one area, while another person wants to do something similar. That's wasted effort replicated many times. Here you put forth all the effort with the data up front. You have easy-to-integrate multiple data sources. It's critical to be able to start small.

I had experience working with companies where you're able to set up an experience study system with two or three data sources. From there you can add data sources as they become available or as time permits because of the relational structure; you don't need to have everything up front. You have your administration system or maybe a claim system that becomes available a little later. You also have agency information. You're able to continue to improve upon that without ruining your current process.

Obtaining quick answers to complex questions determines the power of a database. A query or database structure enables you to ask questions, such as "give me all the policies sold in this state or these three geographical regions that have a volume that's greater than this with a certain underwriting class." You can add to that list and make a complex question. You can hit a button and, within seconds, get back a summary of the data that addresses that question and its major benefits. I believe that when we start talking about OLAP, ROLAP, and other processing tools,

that similarly gets at that kind of power as well, but in a much greater way. Finally, there is the ability to integrate multiple sources, which means that you can start small and easily expand.