A Conversation with Pat Selinger

Leading the way

TO MANAGE THE
WORLD'S INFORMATION

Take Pat Selinger of IBM and James Hamilton of Microsoft and put them in a conversation together, and you may hear everything you wanted to know about database technology and weren't afraid to ask.

Selinger, IBM Fellow and vice president of area strategy, information, and interaction for IBM Research, drives the strategy for IBM's research work spanning the range from classic database systems through text, speech, and multimodal interactions. Since graduating from Harvard with a Ph.D. in applied mathematics, she has spent almost 30 years at IBM, hopscotching between research and development of IBM's database products.

After joining IBM Research in 1975, Selinger became a leading member of the team that built System R, the first proof that relational database technology was practical. Her innovative work on cost-based query optimization has been adopted by nearly all relational database

vendors. In 1986, she established the Database Technology Institute, a joint program between IBM research and software

development teams, aimed at accelerating transfer of research technology into products. In 1997, she moved from IBM research to development, serving as vice president of information management architecture and technology at the IBM Silicon Valley Lab in San Jose, California, and leading technology development for the next generation of data management systems. She assumed her current position in 2004.

Selinger was appointed an IBM Fellow in 1994. In 1999, she was elected into the National Academy of Engineering, among the highest professional distinctions an engineer can attain.

PHOTOGRAPH BY TOM UPTON

Leading her in this wide-ranging conversation about all things database is James Hamilton, who has spent most of his career working on the development side of the database business. For the past eight years he has been working with the SQL Server Team at Microsoft. Prior to joining the Microsoft team, he was with IBM for 11 years, where he was lead architect on DB2. Before that, he led the IBM C++ compiler project. Hamilton graduated from the University of Victoria with a B.Sc. in computer science in 1986 and has a master's degree from the University of Waterloo.

JAMES HAMILTON Let's start with the role of a query optimizer in a relational database management system and your invention of cost-based optimizers.

PAT SELINGER As you know, the fundamental tenet of a relational database is that data is stored in rows and columns. It's value-based in that the values themselves stand up for—represent—the data. No information is contained in pointers. All of the information is reflected in a series of tables, and the tables have a certain well-known shape and form: there's an orders table, a customers table, an employees table, and so forth. Each of those tables has a fixed set of columns: the first name, the last name, the address.

Relational systems have a higher-level language called SQL, which is a set-oriented query language. This is a unique concept and really what distinguishes relational database systems from anything that came before or after.

The set-oriented concept of the query language allows asking for all the programmers who work in department 50; or all of the orders over \$5,000; or all of the San Jose customers who have orders over \$5,000; and so forth. The information in relational tables can be combined in many different ways, based on their values only.

How do you take this very high-level set-oriented question that the user asks and turn it into an exact recipe for navigating the disk and getting the information from each of the different records within each of the different tables? This process is query optimization: the idea of mapping from the higher-level SQL down to the lower-level recipe or series of actions that you go through to access the data.

Query optimizers have evolved as an enabling technology to make this high-level programming language—this data-access language, SQL—work. Without that, you would end up using brute force: let's look at each row and see if it matches the description of what's asked for. Is it department 50? Is it an order that's over \$5,000? It would be very inefficient to scan all of the data all of the time.

So we have access techniques that allow you to look at only a subset of the data, and then you have to plan which of those access techniques makes sense for any given kind of query.

The trick to cost-based query optimization is estimating a cost for each of the different ways of accessing the data, each of the different ways of joining information from multiple tables, and estimating the sizes of results and the savings that you would get by having data in the buffer pools, estimating the number of rows you will actually touch if you use an index to access the data, and so forth.

The more deeply you can model the cost of accessing the data, the better the choice you'll make for an access path. What we did back in the late '70s was to invent this cost-based query optimization and provide a model that was good enough for searching a very large space of choices within a reasonable amount of time, then coming up with a very good cost estimate and, therefore, a very good access path.

JH It's amazing that this number of years later, this work remains the dominant approach to relational database system query optimization. Cost-based optimizers have been a real success in technology transfer from research to industry. Can you tell us a little bit about why that was so successful?

PS The quality of cost-based query optimization has really made it possible for people to have relatively hands-free application development. That is, the application developer doesn't have to know a huge amount about the layout of the data on disk and the exact places where the records are and the exact access paths to those records. There's a huge upside from the application productivity standpoint to being able to do really good cost-based query optimization. So, there's a compelling marketplace force for having good cost-based query optimization.

I participated in inventing the System R query optimizer, which was taken lock, stock, and barrel and put into IBM's DB2 relational database product where it has been continually enhanced. Many of the simplifying assumptions to make the problem tractable back in the late '70s have been eliminated, and the model is now deeper and richer and includes more techniques for accessing the data.

It's a growing science, and that's part of its success. It has been able to grow and adapt as new inventions come along for accessing the data or for joining the data in different ways. Relational database products have gone to a cost-based query optimization approach and have moved away from a rules-based approach, which was really too

inflexible to get you good performance all the time.

The technology still has room to grow—for example, when the data itself behaves differently from what the model assumes. Many optimizers do not model highly correlated data really well. For example, 90210 is a zip code that's only in California. Zip codes are not evenly distributed across states, and there isn't a 90210 in every state of the union. For a user request, nailing down the zip code to 90210 is sufficient and applying another predicate, such as state equals California, doesn't change the result. It won't reduce the number of rows because the only 90210 is in California.

JH One of the enemies of industrial query optimizers is complexity, and that can sometimes yield lack of query plan robustness. Small changes in the queries or in the data being queried can lead to substantially different plans. Customers often ask me for a good plan that is stable rather than a near-optimal plan that changes frequently in unpredictable ways. What direction should we be looking to make progress on the optimal-query-plan-versus-query-plan-robustness problem?

PS I think we have to approach it in two ways. One is that you have to be able to execute good plans, and during the execution of a plan you want to notice when the actual data is deviating dramatically from what you expected. If you expected five rows and you've got a million, chances are your plan is not going to do well because you chose it based on the assumption of five. Thus, being able to correct mid-course is an area of enhancement for query optimizers that IBM is pursuing.

Second, you have to continue to deepen the model because you've got to come up with reasonable plans before you can fine-tune them dynamically. Understanding the correlation between rows or between columns in different tables—noting the zip code example I gave before—is a very important part of continuing to understand the data more deeply and therefore being able to do even better query optimization.

The fact is that as customers use more and more shrink-wrapped packages or have ad hoc users who

haven't gone to SQL school for a year, there's a real need to be able to do good query optimization. You can't have database administrators running into the room, saying, "Don't hit Enter yet. I've got to look at your query to see if it's going to be OK." Outstanding cost-based query optimization is critical to speeding application productivity and lowering total cost of ownership.

JH Let's look for a moment at query optimization and where the technology can be taken beyond database management systems. IBM, and the industry as a whole, has been investing in recent years in auto-tuning and in autonomic computing. Do you see a role for cost-based optimization in this application area?

PS Absolutely. It's a rich new area for us to deal with. Companies have a lot of data that is quite well

structured—an order, a customer, an employee record—but that's maybe 15 percent of all of the data that a company has. The rest of it is in document files, it's in XML, it's in pictures, it's on Web pages—all of this information also needs to be managed.

XML provides a mechanism for being able to do that, but the data isn't quite as regularly structured. It's very dynamic. Every record could look different from the next record even in a related collection of things such as orders or documents.

So you have to have a query language such as XQuery that will be able to navigate and to ask set-oriented questions about this new kind of data. That opens up a different repertoire of data access techniques and requires enhancement of the query optimization process. But I think it's absolutely essential to continue on the path of automatic query optimization rather than put programmers back into the game of understanding exact data structures and doing the navigation in the application program manually. That's simply cost-prohibitive.

JH Looking at new optimization techniques, feedback-directed systems, and dynamic execution time decisions—all significant areas of continuing research—what do you see as the most important next steps looking out, say, five years or so?

PS I think the cost of ownership is on every customer's mind, not just because of the economic downturn that some of them are still in or have just experienced, but because the cost of processors, disk space, and memory are all going down—and the cost of labor is going up.

Furthermore, you have to look at the ratio of how many administrators you need to take care of a terabyte worth of data. Unless you can dramatically improve that ratio, as you accumulate more and more terabytes of data, pretty soon you're looking at employing half the planet to administer it. So we are inventing ways to make administrators capable of handling 20, 100, 1,000 times more data than they do today.

At the same time, we're under pressure to incorporate, search, understand, and take advantage of information that's in this more unstructured form—e-mail, for example. Companies want to be able to look at e-mails or customer service files to give their customers better service, and as we do that, we have to manage and understand and analyze more kinds of information.

As we look at what it's going to take to do that, we have to change the game in terms of the cost of organizing, administering, and searching this data. The autonomic computing initiative that we at IBM have been proposing, that the industry has now adopted as well, is the saving grace that's going to make this possible.

JH You are very focused on driving down the cost of administration. What about the cost of developing applications?

PS There are two aspects to autonomic computing: one is application development; the second is administration. In the application development arena, customers want to have a common set of tools. That's why IBM has been so involved in helping open-source the Eclipse platform and to encourage people to contribute to that platform—so that there can be one set of tools and one set of skills that allow people to range across a variety of platforms using them. The high level of programming—things like XQuery, SQL standards, Enterprise JavaBeans, and COM objects—all contribute to being able to put together building blocks for rapid application development rather than just using coding for navigational data access.

JH I find it strange to be working in an industry that is almost 30 years old, yet it feels in our conversation as if the vast majority of the problems are yet to be solved.

PS That's one of the wonderful things about data management. It includes all of the problems of programming languages, all of the problems of operating systems. It's the same with data storage. Because of the changes in computer hardware architecture—things like large

memories and so forth—there are new opportunities for adapting the data engines to leverage hardware.

Huge things are going on in information management. Information is really the lifeblood of how the government runs, how our customers run their businesses. It's critical to the economy.

JH You have recently returned to the IBM research division after a successful stint on the development team. Tell us about your new role as VP of area strategy, information, and interaction.

PS The role that I agreed to take on is a broadening of what we call information. I've spent 27 years out of the 30 that I've been in data on the structured part: relational database engines, query optimization, making them perform faster, inventing new access techniques for them, and so forth. As I mentioned earlier, that's about 15 percent of the information out there, and the other 85 percent is in this more unstructured form.

What I'm doing in this new role is going after the whole picture: pulling together all of the forms of information and managing all of that information, which can demand a very different design point than structured information. The system's understanding of what's in that data is not very deep, so researchers get more involved in semantics and speech understanding and ontologies and categorizations and various other kinds of analytics to be able to understand what's in that data and derive information from it.

JH Can you describe some projects that you're currently involved with?

PS One of the things that we have spent a lot of time on is the UIMA (unstructured information management architecture). This is an architecture that we think is a framework for being able to represent unstructured information and to analyze and manage it. It's basically a platform where you can plug in things like text analytics and ontology searches, where you could take information and annotate it—this is the name of a president, this is the name of a university—and then be able to do generalizations based on categories and semantics and ontologies, and be able to answer questions about that. For example, a car manufacturer could ask: What's the sentiment of people who call in complaining about brake problems on such-and-such an automobile? How are we handling those? Can we improve our customer service based on these results?

Things like regulatory compliance are also really good examples, where companies are being asked, for example,

to analyze e-mails and make sure on a sampling basis that nothing untoward is happening. How can you provide companies the tools to do that automatically?

JH It's a good time to branch out beyond focusing purely on structured storage since, as you mentioned, relational systems currently manage less than 15 percent of enterprise data. I would argue that, if we were to look beyond enterprise data, we would find that much less than 5 percent of the total world's data is stored in RDBMSs. The funny thing is, in the relational world, we manage close to all structured data; yet, less structured data management is still very close to a green field.

PS And it's spread across so many different formats and so many different management systems. In the content management area alone, there are thousands of companies, and nobody has a big share of the marketplace. You can name the key relational vendors on the fingers of one hand, but if you start naming content management companies, you quickly get into hands, toes, and across your neighborhood. It's a huge number of companies doing very specialized kinds of management. Now is the time to start pulling together those pieces and taking a broader look.

Metadata is going to be a key piece of being able to find out where all that data is and tell us how to access it and provide us an introduction on how to find out more.

JH What are some of the challenges in the broader world that you're working in right now—specifically with respect to nonconventional data types such as photo-

graphs, videos, and voice? Where do you see the big research challenges for the next decade?

PS I think with all of the individual technologies, there's always room for making those better. In addition to all of that, finding ways to bring together the combination of those technologies can be extremely valuable. If speech recognition technology could benefit from the knowledge that text analytics is providing to the things that I've already transcribed, we can use that information to narrow down the focus of what a conversation is likely to be about.

A lot of cross-relationships are possible that have not been fully exploited. Exploiting these synergies is a very exciting area.

JH Are there any other areas that you view as under-researched, where you wish that we would see more interest?

PS I would love to see more people address the problem of metadata. Metadata helps integrate information, particularly information of different kinds and formats, and provides information on what kinds of features can be extracted from that information. Those things have great potential to add to the value of the information that we're so good at storing. I would love to see more work going on there in terms of mapping and discovery—relationship discovery, correlation discovery, and so forth.

JH Are there areas where you feel that we're over-researched?

PS Yet another locking mechanism for data management

systems simply doesn't interest me, but thankfully the research work on such areas is dying out.

JH I've seen two pictures painted of the future of unstructured data. One of them has file systems augmented with search appliances, and another is based upon an expanding role of structured stores that are much more flexible and much more capable of dealing with dynamic schemas and content. Is there a role for file systems and search appliances? Where do you see this playing out?

PS I don't think that any current or future data storage mechanism will replace all the others. For example, there are many cases where file systems are just fine and that's all you need, and people are perfectly happy with that. We have to be able to reach out to those data sources with a meta-engine that knows how to reach and access all those different data repositories and understands all the different formats—.jpg, .mpg, .doc, etc.—and knows how to interpret that data.

The notion of an intergalactic-size, centralized repository is neither reasonable nor practical. You can't just say to a customer, "Put all your data in my repository and I'll solve all your problems." The right answer from my perspective is that customers will have their data in a variety of places under a variety of applications in file systems and database engines. They're not going to centralize it in one kind of data store. That's just not practical. It's not economically feasible. My job at IBM is to drive a strategy

that accesses the data in place and integrates it virtually without necessarily having to integrate it physically.

So, file systems will still be around. They may get enhanced with special search techniques as we have more capability and processing power in RAID systems and disk servers, file servers, and so forth, and relational systems will get richer in what they can handle, but we're not going to replace all of the technologies with any one single answer.

JH Do you see content management systems of the future mostly layered on relational database systems, or do you see them as independent stores built using some of what we've learned over the past 30 years of working on relational technologies?

PS I like the architecture that we have in the DB2 content manager, where DB2 is the library server—the card catalogue, so to speak. It uses some extra semantics in a system-level application surrounding DB2 with some new user-defined types and functions, and stored procedures implementing those applications. It has separate resource managers, which are capable of handling a certain class of data types and styles with this kind of document, these kinds of images. They could be physically stored in either the DB2 as the library server or some separate place or file system out on a number of different engines.

It gives you a flexible configuration. You can exploit as much as you like of the functions of DB2—XML, for example—or you can choose to use some of these repository managers. They may be less feature-rich but are

expert in a particular kind of information and could be stored locally to where you need that data—particularly if it's massive amounts of data, such as mass spectrometry results. Those are huge files and you want them close to where you're doing the analysis.

JH Do you see XML having a more fundamental role in content management systems?

PS I see customers very interested in XML, and many of them today are already using it for interchange. I see customers wanting to have the richness of XML within the database engine itself. That's why IBM is doing a full native implementation of XML inside of DB2 on Linux, Unix, Windows, and z/OS.

If a customer submits a transaction or is sending you an order form in XML, it may be a good idea to keep it in XML and use it that way inside of the database. We could also translate XML into relational if most of the processing from then on is going to be relational. This is a have-your-cake-and-eat-it-too strategy. DB2 can save the data in XML and analyze it, search it, and access it that way, or it can turn it into relational and drive all the transactions that are already written for relational systems.

JH In the relational data management world, we've seen a marked reduction in the number of commercial systems providers. You mentioned earlier that in the content management world there are a large variety of stores currently available. Do you see the same consolidation happening in content management where, over time, there will be fewer store producers, or do you see it continuing with a wide variety of specialized systems?

PS There is some industry consolidation around wanting to adopt a common set of standards, and that will provide a base. Take the JSR170 standard, for example. If you can code to that application interface, then customers will be more able to move between content systems, or an application will not require you to have a certain content management system underneath it. That will offer the freedom for the industry to consolidate if that makes economic sense, and it will allow people to have a common set of applications, even if they choose to have a multiplicity of vendors.

JH Given that relational stores now support XML and full-text search, what's missing? Why haven't extended relational systems had a bigger impact in the unstructured world?

PS The semantics of content management go beyond what we offer in just the data storage parts, the data storage engines, the DB2s of the world. There's a significant set of other abstractions and management techniques that either have to go on top or have to come from a

content management system that uses and exploits an extended relational engine but doesn't solely rely on it.

For example, content management systems have the ability to allow Pat access to Chapter 1 of a document, and James access to Chapter 2, and Ed access to Chapters 1 and 2, at the sub-sub-document level. This is something that relational systems don't do today.

Similarly, foldering, the idea of document collections that really aren't related to similar structure but are tied to some higher-level semantic content, is beyond what relational systems are undertaking at this point.

JH Are there other areas where you see research needed for content managers and relational stores to improve and help customers manage a wider variety of data?

PS If I were choosing today to do research or advanced development, there are a number of areas that are very, very interesting to me. There's continued invention needed in the autonomies. What do you have to do to have a truly hands-free data system that could be embedded in anything? What do you have to do to have truly mass parallelism at the millions-of-systems (e.g., Internet) level? As commodity hardware becomes smaller and smaller, can we link and talk to systems and compute things on a scale of millions, where today we're at a technology level of thousands? How do you deal with data streams where the queries are fixed and the data is rushing by, and it could be unstructured data?

How do you find the Osama bin Laden telephone call as the data streams by, with techniques such as semantic analysis, voice recognition, and automatic speech-to-text transcription and language translation?

How do you accumulate metadata and keep it up to date? How do you manage it, learn from it, derive information from it?

Searching is still in its first generation. There are lots of opportunities to make search better. If it knew you were angry when you typed in your three keywords to a search engine, would that help it understand what you were searching for? If it knew what e-mails you had just seen before you typed those search keywords, would that help it understand what you were looking for? How can a search engine find what you *intended* as opposed to what you *typed*?

How reliable is derived information? There are many sources of unreliability. What if I have a source of information that's right only half the time? How do I rate that information compared with another source who's right all of the time? How do I join together that information, and what's the level of confidence I have in the resulting joined information?

All of those things, as we start dealing with unstructured data and incomplete answers and inexact answers and so forth, are great opportunities for research and advanced development.

JH We've started to see open source have a larger role in server-side computing. Specifically in the database world, we've now got a couple of open source competitors. Is open source a good thing for the database world?

PS I love the idea of open source. I was the manager of the IBM Cloudscape team at the time that we contributed it to Apache, where it has become an incubator project under the name Derby. My dream here is that this allows many more opportunities for using databases in places where people wouldn't ordinarily go out and buy a database engine.

So open source can bring the benefits of the reliability, the recoverability, the set-oriented query capabilities to another class of applications—small businesses—and the ability to exploit the wonderful characteristics of database systems across a much richer set of applications. I think it's good for the industry.

JH A responsibility that I know you've always taken very seriously is mentoring—helping the next generation to

grow and succeed. It's obvious how important this is, but it is very difficult to do well. What's your secret?

PS Some people laugh at me and say that I eat my way through my mentoring. I think people talk much more easily when they're sitting over a beer or having lunch or a cup of coffee, so I do all my mentoring basically in the cafeteria over food, in a bar, or over dinner. It puts people in a more comfortable setting for talking about what's really on their minds. I have about 30 men and women whom I mentor, probably two-thirds women. Any lunch that I have free, I spend it mentoring.

What I think mentoring does for people is to bring a third view, an experienced practical view, and it gives people a sense that they do have choices and that they have much more control over the directions of their careers than they think they have.

JH What are your thoughts on getting more women involved in computer science?

PS The National Academy of Engineering conducted a study several years ago looking at what we can do to get more women interested in engineering. What was particularly surprising to me is that other countries are more successful than we are in this area and that women make up half the work force in careers such as biology, medicine, and law.

So it's not the science that discourages women. I think there's something about the way North American culture treats the engineering profession that dissuades women from getting into it.

There's a perception issue as well in that when you talk about an engineer, the popular media still pictures the guy in the white shirt with the pocket protector and the calculator hanging from his belt, and wearing big thick glasses. To the extent that we perpetuate that kind of an image, it's very defeating to a girl who doesn't want to grow up to be like that. Some of the recent popular television forensic shows actually have quite positive images of women being capable and scientifically knowledgeable. I think some things like that can really be helpful. I've also written articles, for example, on the importance of making sure your teenage daughter takes enough math and science so that she doesn't rule out that type of career before she even knows what it is.

I spend a lot of time in outreach. Next week I'm going to a women's university in Mississippi and talking about my career choices and that, yes, you can have kids and survive—succeed, in fact—as an engineer. Just providing people with the comfort of knowing that it's possible may help them consider engineering as a career opportunity for themselves.

JH You certainly have been very successful yourself. Are there people or events that you can point to that have really helped you personally and professionally?

PS I had some wonderful mentors early in my career. The original manager of the System R project, Frank King, was just tremendous in encouraging me and spending the time talking over business issues with me that got my feet under me at a very early stage in my career. Janet Perna and Don Haderle, who are the senior-most people in IBM data management as executives and technologists, have just been wonderful.

Every opportunity I've wanted to have, they've made it possible for me to pursue. It's been a much easier path for me because Janet Perna appreciates the value that diversity brings.

I love being in a development organization where 30 to 40 percent of the people on the team are women. That's not typical across the industry. I think being at IBM has been very, very valuable to me, and I've been treated very well. ☺

LOVE IT, HATE IT? LET US KNOW

feedback@acmqueue.com or www.acmqueue.com/forums

© 2005 ACM 1542-7730/05/0400 \$5.00

The image shows the cover of the ACM Queue magazine. The left side features the 'acm queue' logo in white on an orange background, with the tagline 'architecting tomorrow's computing' below it. The right side has a dark background with a textured, wavy pattern. The main title 'Mobile Applications' is in large white font, followed by the subtitle 'Enterprise-grade wireless Delivering mobile media'. Below this, it says 'Also Next Issue: You don't know jack about networking' and 'Queue interviews Sun Distinguished Engineer Tim Marsland'. At the bottom right, it says 'Coming in May'.