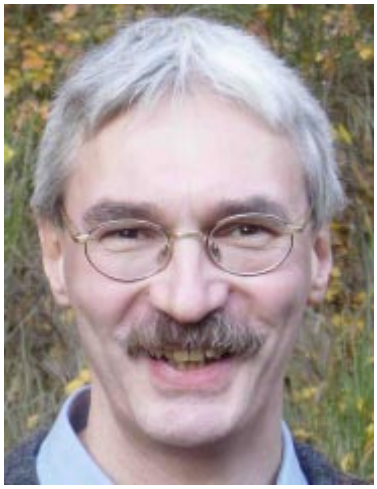
Gerhard Weikum Speaks Out

on Why We Should Go for the Grand Challenges, Why SQL Is Too Powerful, the Myth of Precision, How to Have a Big Research Group in Germany, and More

by Marianne Winslett



Gerhard Weikum
http://www.mpi-inf.mpg.de/~weikum/

Welcome to this installment of ACM SIGMOD Record's series of interviews with distinguished members of the database community. I'm Marianne Winslett, and today[1] *we are at the VLDB 2007 conference in Vienna. I have here with me Gerhard Weikum, who is a Research Director at the Max-Planck Institute for Informatics in Saarbrücken, Germany. Gerhard is an ACM Fellow and the president of the VLDB Endowment. He has a SIGMOD Best Paper Award, a VLDB 10-Year paper award, and a CIDR Timeless Idea Award. His PhD is from the University of Darmstadt. So, Gerhard, welcome!*

Most of our readers and viewers won't know that a German university has at most one professor in an area. So, by definition, you are the only *database professor at the Max Planck Institute for Informatics. From a US perspective, this guarantees that a CS department doesn't have critical mass in any one area. How have you built such a strong group under that system?*

The main thing to watch out for is having good people around you. Great students are a wonderful asset. If you have a strong team of great students, you can do good work with few people.

Let me also correct a little bit your perception of the German system. In Germany, you become a professor only at a relatively mature stage, and then you would traditionally have tenure right away. Only gradually are we adopting elements from the Anglo-American system, where you become a non-tenured assistant professor earlier. Obviously you cannot be a graduate student

[1] The print version of this interview has been revised slightly, to bring it up to date.

and then disappear into the world for ten years, and then come back as a professor. So during that intermediate period, people have titles such as *post-doc*, *research associate*, and so on. I have a few of these people as well.

I am at the Max-Planck Institute, but I am affiliated also with the University of Saarbrücken, which is on the same campus. So I have a colleague at the university as well. This used to be Christoph Koch; since fall 2008 it is Jens Dittrich, who came to Saarbrücken from ETH Zürich – the same path, by the way, that I took 15 years ago. Jens is a great colleague, and we have started collaborating.

You work in many different areas now, but you started out narrowly focused on transaction processing. What led you to switch from a narrow focus to such a broad one?

I am curiosity driven. I want to learn new things, understand new things, and the best way of doing this is by working on a new area. It forces you to understand the state of the art, and dig into the existing literature. Typically I start by teaching the subject or the field. For example, when I moved towards information retrieval seven years ago, I started by teaching an information retrieval class. By teaching the class, you also get research ideas.

When I look back, I think my career roughly consists of three parts. First was the transaction processing. (By the way, at that time we did not think of transaction processing as being *narrow*!) Then in the early nineties, I started working on the theme of automatic tuning, and that went on for about ten years. Then around 2000, I moved to the field of database and information retrieval integration.

You don't think you have gotten any broader over the years?

I have gotten broader, because there is some temporal overlap between these three areas. I do not completely stop working on all aspects of the previous theme, once I move on to a new theme. But some of the overlap is not exactly original research. For example, on the automated tuning theme, recently I have been doing tutorials together with Surajit Chaudhuri. That is not original research. Surajit is still doing original research; I am involved, essentially, in the educational part.

I have more students than I used to, because at the Max-Planck Institute, you are supposed to build up a big group, so that you have critical mass. At the university, before I joined the Max-Planck Institute, I worked with a handful of students. I think I was more focused. Now we have 20 people in the group, of which 10-15 are students and the rest are post-docs or people who, in the US, might be considered as young assistant professors. I tend to give some leeway to talented students. If they come with their own creative idea, and if they convince me that it is a good direction, and they have the talent to tackle the issues, then I let them go. So some of our work (such as what we did on machine learning) is prompted by a student having a good idea, and then I join as an advisor and get involved.

How do you keep yourself from being spread too thin?

That is a good point. You have to continuously monitor yourself and see if you are focused. I try to have one or two main projects, typically centered around some kind of systems-building activity. But we don't build the system just to do systems-building work; the system is the focal point of the research activity, and an integration point for all the sub-projects. How to get students to talk to each other is a great worry, because they work from different angles on the

same system and need to understand how their components interact. When I was at the university working with 5-10 people, we typically had one system-oriented project plus something more half-baked and adventurous that could lead to a larger activity in the future. Now, at the Max-Planck Institute, we have two or three of these system-centric projects, just because of the size of the group. I have two roles to play: a research manager in all of the projects, and a researcher in a subset of them.

You have moved from an area where the work is very precise, to one where everything is very fuzzy. What led to that transformation?

I think that the notion of precision is a myth! In the good old days, when you were working on payroll data, maybe things really were precise. My salary, hopefully, is precise. But when you talk about scientific data such as measurements, when an experiment gives you a value, there is always an error bar, some degree of uncertainty. We just ignore that uncertainty. We pretend that yes, the temperature was exactly 35.5789 degrees Celsius, but that is not the case. In the text world, or in integration of text and data, there is inherent uncertainty in interpreting the text and sometimes also in the information need on the user's side. For IR people, a query is an approximation to the user's information demand. It is not a dogmatically given precise query.

It sounds like you are saying that you were ready to face reality.

Definitely one cannot be precise in a world of text data, such as when extracting entities and relationships from text. It is important to capture the confidence in what you are doing. As you move on to aggregating the data, composing the data, and querying the data, eventually you may get very far off from your original starting point, and that might lead to misinterpretation of what the query results mean. So it is important to capture the confidence, or lack of confidence, at each point along the way and understand how that confidence propagates to later processing steps. I am a deep believer in probabilistic data.

Working in many areas puts you in a good position to tell us where the database field is going, and what some of the interesting problems are for the future.

This is highly subjective, obviously, so my answer will match what my own group is working on. I see an enormous explosion of information everywhere. To me, text is one of the kinds of information that humans produce most efficiently. Precise data, that is, schematically structured data, just takes too long to produce. At its central core, a whole essay might not say much more than one tuple in a database, but most people can produce the essay more easily than the tuple describing the gist of the essay. So I expect to see a continuing explosion in text data, connected with fuzzy structures such as Web 2.0, blogs, and bulletin boards, with tagging, bookmarking, and all those kinds of things.

People can produce speech even more efficiently than text, and speech recognition is getting much better. I see potentially a big wave of speech processing that can extract interesting elements from the speech itself, such as the speaker getting very emotional, getting very upset, being ironic, and so on.

Yes, but you cannot tell me that speech recognition is database research.

I don't care which drawer you put me in: I am a computer scientist. I think it is good to look across the fence at other areas. I am not saying speech recognition is a database issue; it has a data management component, and lots of other components.

Someone suggested that in light of your move from things that are precise to things that are statistical, you may be being "seduced by the dark side" and end up doing AI. Do you think that you are heading in that direction?

That would be perfectly fine, I have nothing against this. When I was young, I also thought in terms of idioms like "being seduced by the dark side." Mike Stonebraker used to say, "This problem is AI-complete," which meant that the problem was beyond any hope of solution, was science fiction, was on the same level as "Scotty, beam me up!" Now, as I grow older, I think this attitude is wrong.

We should go for the grand challenges. Physicists also go for grand challenges, such as controlled fusion, and they have huge projects in areas such as plasma physics. And no, they don't keep their promises. They say that they will have a breakthrough in thirty years, and it doesn't happen. That tells us that their problem is hard. In the life sciences, in understanding not just the syntax of the human genome but also its semantics, its functionality – this is a century-long project. So, I think we computer scientists need to grow up and become more self-confident, and go for the really grand challenges. Then we can break the challenge down into steps; obviously it does not make sense if we spend 30 to 50 years with no visible progress. If we gave ourselves a goal whose validation or falsification is a hundred years in the future, we would not live long enough to see whether we are wrong or we achieved a breakthrough. So we need to break our grand challenge down into milestones, and into short- and intermediate-term goals as well. The grand challenges are out there, and we should pursue them.

Tell us about your 2002 VLDB 10-year paper award.

That refers to a paper we published in 1992 in the VLDB in Vancouver. Our paper was on load control for lock management. You can run into lock contention situations under extremely high load, like load surges with mixed workloads. Our paper was about how to do admission control and concurrency control, so it was very narrow and very technical. I don't think we got the 10 year award for the paper itself, but for the theme behind it: automatic tuning.

In 1989 I came back from my post-doc time at MCC in Austin, and took a position as an assistant professor at ETH Zurich. At MCC, we had gone after a grand challenge: a massively parallel database machine, which was a big thing back then. I was influenced by that project, so I wanted to work on something really challenging and long term, but at the same time something off the beaten path that not too many other people would work on. I was a young assistant professor, and I didn't have so many resources anyway.

I came up with the theme of trying to automate tuning by putting more smartness in the system. We pursued adaptive methods, we pursued stochastic models, and we pursued feedback control, which was the major element behind this VLDB 1992 paper. I think that when the committee picked us for the 10-year award in 2002, they were acknowledging the importance of the theme that we had started working on: automatic tuning, self-managing systems, or autonomic computing, as it is called now.

Tell us about your 2005 Timeless Idea award from the Conference for Innovative Database Research (CIDR).

That is kind of a fun award; it is very different from the other awards I have received. I have listed it as well as the more serious awards on my web page, because I think people should not

overrate the serious awards. There is some luck involved in receiving one of them, and so there are many good people who deserve one and have not yet gotten one.

The CIDR award refers to a late-night event called the Gong Show that takes place at the conference. Twenty to 25 people each get five minutes to talk about whatever they want: challenges, unsolvable problems, their pet problems, jokes, whatever. People are in a very loose mood at 10 in the evening, after some drinks, and my presentation was semi-serious. My theme was the experimental methodology in our field. Obviously this is a serious theme, but the presentation itself was more on the funny side. I showed some tricks about how you can massage your performance charts and make them look better than they are. My real point was that we should carefully reflect on the experimental methodology in our field. We are really kids in this regard, compared to other fields. The slides for my presentation are available on the web, by the way, and the last one or two slides have some serious thoughts on what to do about this problem. This was a very timely topic, considering the recent change at SIGMOD to include experiment repeatability for accepted papers.

What do you think of SQL?

As a language? I have never worked on designing a language or on other language issues, but I have considerable teaching experience. When I teach SQL, I want to give formal semantics, like translations to relational algebra or to relational calculus. You can do this easily for a narrow subset of SQL, like select, project, join. But once you have complex SQL queries with five levels of nested subqueries, a handful of left outer joins, correlated variables, and aggregations and groupings, hardly any students get such queries right. We teach these features by example: the teacher gives one example of how to use a new feature and hopes that the students will magically know how to use the feature correctly. But the students don't.

I think that the result of this teaching methodology is that our students will produce lousy SQL code later on, with bugs that must be found and removed. It is not easy to debug SQL, because it is so declarative. The more declarative a language is, the harder it is to debug. Debugging SQL typically involves breaking a complex statement down into simpler SQL building blocks. But then, why do you want such a super powerful language?

So I think there is virtue in having a language that does less. My role model for this is people like Niklaus Wirth, the inventor of Pascal. His languages have a *less is more* philosophy. They have the right, easy to use building blocks. The building blocks are composable, and the language doesn't have too many features. Feature richness is a curse. I know this is debatable, and many people would disagree. This is just my opinion, my two cents, nothing else.

If the more declarative a language is, the harder it is to debug, then assembly code should be the easiest to debug.

That is a good comment! In terms of state-driven debugging, where you look at the state of a program, that is actually true, or close to true. But ease of debugging is not the only important aspect of a language. The amount of time required to design, code, and reason about a program is also important.

I am not saying declarative programming is wrong. I am just saying that if you go all the way to being declarative, then you also have to be extremely precise in terms of the language's semantics, and you have teach this semantics. Hoping that the students understand the super-complex semantics from a few examples cannot work.

There is a TODS paper that gives a formal semantics for SQL.

It is tough to teach from that paper. And by the way, this is not specific to SQL, this is about all declarative languages. I think making them too rich is an issue. There is no silver bullet here.

You are a director at a Max-Planck Institute, and you are the president of the VLDB Endowment. Do you have further organizational ambitions?

Oh, definitely not! I am taking on these kinds of service positions because I think someone has to do them, and I feel indebted to the community. When I was young, I attended conferences and enjoyed them. Someone has to organize them. And someone has to organize the organizations behind the conferences. So I think I am just doing my duty.

I have been told that you have great technical, administrative, and meeting skills. Since people like that are in high demand, how do you balance your time between service to the community, managerial roles, and research?

I don't think I am particularly better at these things than other people. When I accept a job, I try to do it right. That is my attitude; it is probably in my genes. But I am certainly not a natural born manager or chair of anything. When you say I have great skills in these kinds of things, this is just because I put time and work into this.

I am trying to limit the time I put into these kinds of service activities. I am still at a phase where I feel indebted to do these things. At some point, I will think that I have paid back my debt, and then I will refuse them again. At that point maybe I will write a book again, or something of that kind.

What killed that nice Parallel and Distributed Information Systems (PDIS) conference?

Oh, that was Jeff Naughton! Jeff and I were the PC co-chairs of the last PDIS conference, held in Miami in 1994. Actually, I am just kidding. Jeff and I killed PDIS together! The truth is that PDIS killed itself automatically, and there is nothing wrong with that.

There are specialized conferences such as PDIS, Deductive and Object Oriented Database Systems (DOODS), and some of the workshops that repeat year after year. These specialized conferences and workshops make sense as long as their topics are not sufficiently well covered by mainstream conferences like VLDB, SIGMOD, etc. But when they run out of steam, either in terms of submissions or attendance, because the mainstream conferences have absorbed these directions, then there is nothing wrong with letting them phase out. So PDIS just phased out.

I see what you are saying, although since there is such a submissions crunch at the main conferences, it seems like we should be able to have some good specialized conferences.

Things are different now. For example, the web space is exploding, so there might be a case for a Web 2.0 conference. But of course there are already new conferences in that space. Unlike the typical DB guy, I am also observing what's going on in our neighbor communities, IR and web research. For example, there is a new conference called WSDM, pronounced "Wisdom", which is for all search- and mining-related issues on the web, including Web 2.0.

I hear that you like to take solo vacations that involve crawling on your belly in the desert. Is that how you would describe them?

I like the desert. I have spent significant time in the US, and I like the Southwest. Yes, I have done some backpacking solo trips. The crawling is not by design, but sometimes you end up in a space where you cannot backtrack. I did some canyoneering, for example, in the Southwest, in southern Utah. Sometimes you are at an obstacle, and you can get forward, but you cannot get backward, because you already went down some obstacles, like steep drops over boulders and small waterfalls and that kind of thing. And sometimes you have to crawl underneath boulders.

If it is that dangerous, is it wise to be out by yourself?

It is no more dangerous than other things. I have had in my life a few injuries; one of them was a torn ligament in the ankle. That happened while I was waiting for someone on the sidewalk! So, hiking by myself is no more dangerous than other things.

True, although when you tore your ligament on the sidewalk, there were people all around you. If you are off in the desert in an obscure spot, no one will be around you. And your cell phone might not work either.

That is true. In fact, I don't carry a cell phone on these trips, because it is very clear that it won't work. Typically you register with some agency, the Bureau of Land Management or a national park. A few days after you were supposed to return, they would start searching for you. And I am a careful guy during these trips. I don't take unnecessary risks.

What do you think of the European funding agencies' emphasis on very large projects?

"Large" can mean different things; you were probably referring to the number of partners in the project. There are different kinds of large projects. Those with 20 partners, or maybe even more, are really *networks*, so they break down into subprojects with perhaps five partners. There is some loose coupling between the subprojects, in the sense that people talk to each other, but there are no joint development goals across subprojects. So these kinds of large projects are fine. The many-partner projects I have been involved with have been networks.

Some many-partner projects are designed to have hard deliverables in the form of systems at the end of the project. If you strictly evaluated such projects against their officially declared goals, then they would be bound to fail. I do not believe that 20 partners can do joint development, especially if half of them are academic partners, and no partner has a big budget.

There are also what are called STREPs – Specifically Targeted REsearch Projects, perhaps – I am not sure what the acronym stands for. These typically have 5-10 partners in different roles. There might be 3-4 technology partners and research partners, and the other partners would be technology transfer or requirements analysis partners. Sometimes, when companies are involved in these projects, they can play different roles. They could play a major development role and a technology role; but they could also play a role of supplying use cases, requirements, or potential business models a few years down the road. The latter type of partner is involved in the early and late phases of these projects. That reduces the remaining partners to just a handful, who are the core technology and development and research partners. That model works.

Do you have any words of advice for fledgling or midcareer researchers or practitioners?

I know quite a few people in computing theory, where there are so many young superstars, and then there are the 40 year olds. In Europe, if you are a decent theoretician, but you are not among the stars and you are approaching 40 and you have not gotten a tenured professor position, you might be in trouble, because there are always these 25 year old new superstars. These 40 year olds are not bad researchers. They have good skills, so they might consider moving to another area where they can leverage their expertise and their skills and methodology. The irony is that I think the demand for researchers on the practical side is much higher, but there is a disproportional fraction of talent going into theory because it looks more challenging. So there are people who are just not good enough to be a superstar in theory, but could do very well by moving to a more practical field.

From among your past papers, do you have a favorite piece of work?

There are a few I like very much and I am very proud of. One that is lesser known is a workshop paper in WebDB 2000. We started working on ranked retrieval for XML, and we had this little paper, just 6 pages, for which we coined the title "Adding relevance to XML." At that time, the XML wave was about to take off, but it was all schema driven. I thought there was no way of unifying the world into the "right" schemas. If you have heterogeneity, diversity, and ambiguity, then you need ranking. This was our first paper on DB and IR integration.

If you magically had enough time to do one additional thing at work that you are not doing now, what would it be?

Write a book with Surajit Chaudhuri.

On automatic tuning?

You got it!

If you could change one thing about yourself as a computer scientist, what would it be?

I should have learned more theory when I was young – more math and more theory. I see that this can be leveraged. You can be much more effective and efficient at certain things when you have the right base ingredients.

Thank you very much for talking with me today.

Thank you, Marianne.