Conversation with Kent Sievers Continued...

KS: After taking an operating systems class from Dr. Evan Ivie, I applied for a job as his TA. The job was already taken but he was kind enough to hire me to work on a special assignment writing a terminal emulator for some new monitors. Dr. Ivie was an inspiration to work with. He did a lot to encourage and help me. I owe him big time. What a wonderful man!

CSC: Where do you work now, what is your position and what are you working on?

KS: I spent 6 years at Unisys in Salt Lake working on their Distributed Communications Processor. I left Unisys to work with some old friends at WordPerfect in 1989. I survived the merger of WordPerfect into Novell. I have been working for Novell now for 17 years. I have passed up many opportunities to go into management because I would just rather program.

CSC: Debugging code can be difficult. Could you tell us about the most difficult debugging problem you have solved?

KS: I am actually highly respected for my debugging abilities. I get all of the toughest problems. So I have a lot of stories. The funniest fix was “a piece of scotch tape”. At Unisys I was flown to Olympia Washington to look at a problem in their communications processor. It was dropping connections several times a day and this meant that their highway patrol officers were having to walk up to car window not knowing whether it was a little old lady or an armed criminal. I was looking at the dumps from the crashes and found the problem. But the fix was a big rewrite of a large chunk of code. I couldn’t make them wait while I worked it out. So I looked for a work-around. The trouble all started because of a backed up queue. The queue was backing up because the master console was not processing because its printer was out of paper. The work-around that bought me the two days I needed to fix the code was putting a piece of tape over the paper censor on the console of the printer.

CSC: Working on a single product for a long time gives you a unique perspective on software evolution over time. What advice would you give an organization that is about to design the software for a new product?

KS: Almost all of the time—planning code structure for the future is a waste of time. You end up with including bells and whistles that never get used and you very seldom guess correctly on direction. I am convinced that you actually end up with less work if you create the best code for the job at hand and then re-factor it later. Don’t try to reuse code the first time. Don’t even try to reuse it the second time, just cut and paste it. You probably need 3 or 5 cases of how that code will be used, before you can cleanly identify how to structure it for sharing. Planning UI and database/storage layout is almost the complete opposite. Code can be re-done, but you end up entrenched in legacy and UI and DB issues that will haunt you forever. Take the time to design and dream with a long term end-point in mind. You have to do this to avoid painting yourself into a corner. Once you have the while, glorious long term picture laid out, you have at least a change of scaling it back from without limiting your future options. Design the sports car, shoot for the compact car and deliver the go cart. At least you will make room for an engine of some kind [...]

CSC: Finally, do you have any other advice for recent CS graduates who have just begun a programming career?

KS: What I see is very disappointing in our new graduates is the ability to work on and re-factor existing code. They have almost no skills for knowing how to change code without breaking it. They tend to want to rewrite anything that they do not understand. They tend to not consider all the paths through the code, the boundary conditions, the performance implications, etc. There should be courses on re-factoring and how to work on existing code. A second real shortcoming is that they have not worked on a large product. A lot of the work is fairly large scale with lots of developers working on the same large code base. Learning source control tools, proper unit testing, how to work out APIs and interfaces others will use [...] This ties in to the third shortcoming, making the code readable for others. This is usually more important than almost anything else. All homework assignments should get 10% off if they ever have a loop with i, j, or k as the iterative.

To read the full conversation, including war stories about debugging and more thoughts on software engineering visit http://csaa.byu.edu/sievers

Faculty Spotlights

Graphics Professor Nationally Recognized at SIGGRAPH

Dr. Tom Sederberg, a CS professor and an associate dean in the College of Physical and Mathematical Sciences, received the Computer Graphics Achievement Award, the highest annual honor given in the computer graphics field, during the 33rd Annual ACM SIGGRAPH Conference in Boston, held from July 30 to August 3, 2006.

He was nationally recognized as a pioneer and expert in computer graphics for his contributions to free-form deformation (FFD) and his use of algebraic geometry in geometric modeling. His work on a new technology, T-Splines, allows artists and designers to add detail and merge models in previously impossible ways. ACM, the Association for Computing Machinery, is an educational and scientific society uniting the world’s computing educators, researchers, and professionals to inspire dialogue, share resources, and address the field’s challenges. ACM SIGGRAPH is the foremost computer graphics association in the United States.

"I never imagined receiving the Computer Graphics Achievement Award, so it came as a very pleasant shock," said Dr. Sederberg.

Dr. Sederberg has taught at BYU for 25 years. He is a world-class researcher, said colleague

Faculty Advancements

Dr. Kent Seamons and Dr. Chuck Knutson were advanced from Assistant to Associate Professor in August 2006.