Liskov’s Creative Joy

Barbara Liskov muses about the creative process of problem solving, finding the perfect design point, and pursuing a research path.

The greatest joy Barbara Liskov has experienced in her distinguished career has not been the results of her influential work but the creative process itself. “It’s incredibly exciting,” she says, “to be thinking about a problem and suddenly see a way to solve it that you hadn’t thought of before, and that makes a lot of other problems go away.” Creative activity is what makes research so interesting, she says, and “is not dissimilar” to what artists of all types experience during their work process. “It just happened to show up for me while thinking through solutions to problems,” she says.

Among the contributions for which Liskov received the ACM A.M. Turing Award is using data abstraction to organize software systems. This new way of thinking resulted in a paradigm shift that had immense practical consequences; it made systems much easier to build and more likely to operate correctly. It involved creating modules with an interface consisting of many operations that provided more flexibility for users than previous techniques and also allowed more details of the implementation to be hidden.

The work that led to this insight began in 1971 when Liskov was at Mitre Corporation building VENUS, a small, interactive timesharing system. She left to join Massachusetts Institute of Technology, and during the transition began reflecting on what she had accomplished with VENUS. “I stood back and thought about programming methodology and what I did in organizing the system. I saw there was this different technique being used,” she says.

This major discovery was the basis of a sequence of advances that refined and extended these ideas, Liskov says. First she saw that multi-operation modules could be naturally linked to programming languages as a way to define new data types. Working with her research team, she built on this insight to design the programming language CLU. She decided to design a programming language because she wanted everything to be well-defined and such precision is necessary for programming languages because they are mathematical artifacts. Liskov also thought presenting the idea of data abstraction in the context of a programming language would make it easier to communicate to programmers. Additionally, Liskov firmed up the separation between how a data abstraction was implemented in a programming language and how it was described in a specification. Later, she developed the Liskov substitution principle, which explains how hierarchies of data types should be organized.

During the early 1980s, Liskov became interested in the ARPANET, the precursor to the Internet. Only a few major universities and a small group of people were using it for email and file transfers, but computer scientists dreamed of building programs that worked on a collection of ARPANET-connected machines. No one knew how to do that, so Liskov decided to tackle the problem. While working on CLU, she consciously had limited her work to sequential programs as opposed to concurrent ones with many parts running in parallel. “We had enough problems without thinking about concurrency,” she explains, “but I had always planned as a next step to return to concurrency.” The result was the language Argus, which enables coders to write programs with components on different computers that communicated remotely through the fledgling Internet.

A stream of related work followed. Liskov delved into other aspects of distributed computing, particularly how to store files online instead of on an individual’s machines. That, in turn raised questions about crashes and losing information. Liskov worked on highly reliable storage on remote machines, which piqued her interest in replication algorithms. “To solve the problem, you must have more than one machine to store data,” says Liskov, “Then you need a protocol that enables machines to keep data in sync so you always get the most recent copy. That was the precursor to my fault tolerance work. Not yet Byzantine failures—just plain old crashes.”

Asked what it was like to develop a computer language, Liskov says, “You’re trying to create something simple and yet expressive. You’re looking for the perfect design point where the mechanisms that you put into the language are powerful enough to allow people to do the things they need to do in a fairly straightforward way, and yet syntax and semantics remain simple enough that the complexity of language isn’t overwhelming. It’s very hard work to find that design point, but it’s very satisfying. It’s a lot like mathematics because you’re looking for the elegant solution.”

Reflecting on the progress of computer science in general during her career, Liskov says people were naïve...
“The question of how you know what’s worth working on and what’s not separates someone who’s going to be really good at research and someone’s who’s not. There’s no prescription.”

Women in computing have also made progress, although they continue to encounter unconscious gender bias, Liskov says. As associate provost for faculty equity, she educates colleagues, including members of search committees, about unconscious bias. She references studies of sexist hiring processes, including one that involved evaluations of résumés. When Swedish researchers changed men’s and women’s names on resumes, the resumes with a woman’s name were ranked lower than those with a man’s. In another example, a symphony orchestra held auditions behind a curtain and, with the gender of the musicians being unknown, more women were offered jobs. “Hopefully telling them makes them more sensitive and sophisticated,” says Liskov, “so that they notice when a letter of recommendation compares a woman only to other women, for example.” However, she says the issue involves not only the biased material that hiring committee members see but also their bias in how they interpret it.

Liskov’s advice for those wishing to pursue a career in research is to avoid taking a certain direction because it is likely to yield many published papers. Instead, she encourages following one’s own star. “It’s much better to go for the thing that’s exciting,” Liskov says. “But the question of how you know what’s worth working on and what’s not separates someone who’s going to be really good at research and someone who’s not. There’s no prescription. It comes from your own intuition and judgment.”

Based in Manhattan, Karen A. Frenkel is a freelance writer and editor specializing in science and technology.

© 2009 ACM 0001-0782/09/0700 $10.00