Letter from the 2016 IEEE TCDE Impact Award Winner

In March I received the following news by e-mail: "...on behalf of the IEEE TCDE I am delighted to inform you that the TCDE Award Committee has awarded you the IEEE TCDE CSEE (Computer Science, Engineering, and Education) and Impact Award for this year 'for leadership and research excellence in building impactful data management systems, engineering tools, products, and practices'." I was very surprised, and I was extremely honored to be chosen! Along with that news came an opportunity to write a one-pager about "anything" for the DE Bulletin, and I've decided to use my DE space to offer some in-my-opinion (IMO) "stylistic suggestions" for researchers in our community – thoughts aimed at all of us, but especially at the younger set.

1. **Results, not papers, are the objective!** All too often, conversations with our fellow academics (both faculty and students alike) or research lab staffers include phrases like "I'm working on a paper about ..." or "I want to write a paper for SIG*XXX*...". The goal of our work should be to "do cool stuff" – to build cool systems or subsystems, or to come up with cool and useful algorithms or data structures or insights – not to write papers. Once we have something to report, it's paper time – but the paper should never be the end goal (IMO).

2. **It's possible to over-publish – but please don't!** In the "good old days", a fresh Ph.D. student would have a small handful of papers on their CV – maybe one per year spent in graduate school – and that was sufficient as long as the results were cool (see point 1). It's really not possible for one person to come up with a new publication-worthy result in under 6-12 months per result. Look at some of the most impactful systems researchers in CS – e.g., let's look at two of my heroes, Barbara Liskov (MIT) and John Ousterhout (currently at Stanford). Both are "repeat offenders" at doing terrific and impactful work on topics like operating systems, file systems, programming languages, CAD tools, storage systems, ... Each has built systems that have had truly lasting impact on our field. But check out their publication records in DBLP...! During a number of their years, each published just a small handful of papers with their students. If we just counted papers, neither would have gotten hired or been tenured – it's their cool stuff and their impact that have mattered over the years. We should become suspicious and skeptical about quality and motivation when we see a CV with more than a few papers per year – it's pretty unlikely that those are all papers that really deserved to be written (IMO).

3. **You won't really know unless you try!** Somewhat sadly, our systems conferences often include some papers that – if you really tried to use the ideas – just wouldn't work. If you work on a problem in isolation, it's dangerously easy to develop something that sounds good on paper, simulates or works well in isolation, but would fall on its face in a real system due to incompleteness or aspects that were overlooked (possibly on purpose, more likely by accident). Great, so you have a really cool new search structure! But can you build a big instance of it fast enough? Can it be updated? Made multiuser and recoverable? Is the information needed to build and maintain it available in an actual system setting? Are its APIs compatible with how real systems are structured? Is the portion of the end-to-end path that it improves on really where the key bottlenecks are? (Being 10x faster on 2% of a query's execution time sounds impressive if you don't say the 2% part too loudly.) The ideal approach (IMO) to contributing a cool new result in systems is what I like to call the "BMW" approach: Build, Measure, Write – in that order. Come up with your idea(s), or examine ideas from others, and then start by building them completely – preferably in some actual system setting. Then measure their behavior and understand what you see – figure out why you're seeing what you're seeing. When doing experiments, first use workloads used by others – yes, their same workloads – before you show how your ideas do on your own favorite use cases. (That's how it's done in the real sciences – and CS should do this too.) Finally, once you're done, you'll have a well-understood and cool new result (see point 1) – so *now* you should write it up.

Please give some consideration to the points above. Our field doesn't suffer from a lack of papers, so let's see if we can reset our research culture a bit and move it back in the direction of the "good old days". Thanks for reading – and thanks to the TCDE community for somehow picking me as its 2016 CSEE Awardee! While it wasn't actually deserved (IMO), I'm certainly honored and appreciative.

Michael Carey
U.C. Irvine