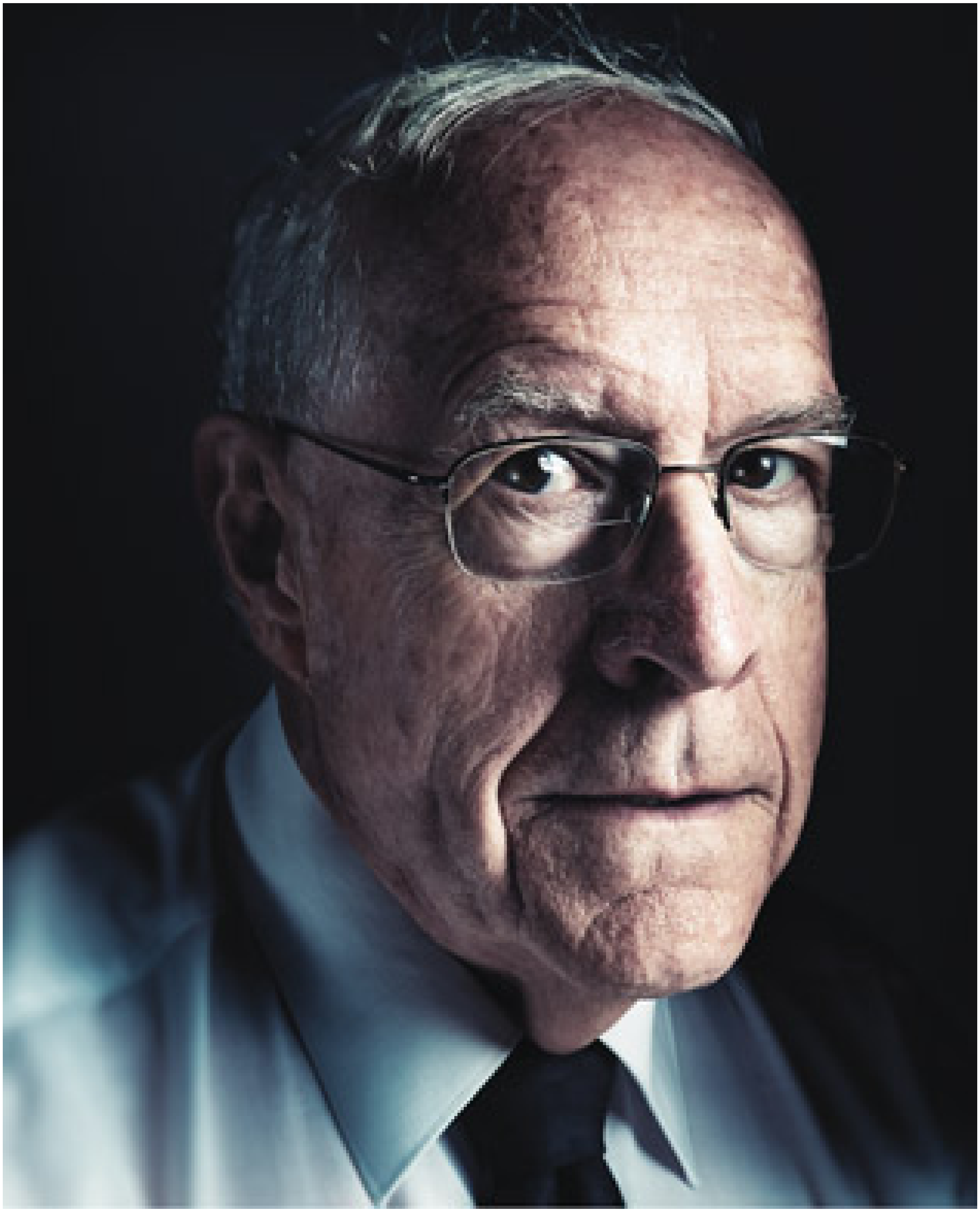
KEVIN KELLY MAGAZINE 07.28.10 2:00 PM

MASTER PLANNER: FRED BROOKS SHOWS HOW TO DESIGN ANYTHING



We might think that the limiting factor on many design projects is money, Brooks says, but that's not true.
Photo: Marco Grob

YOU CAN'T ACCELERATE a nine-month pregnancy by hiring nine pregnant women for a month. Likewise, says University of North Carolina computer scientist Fred Brooks, you can't always speed up an overdue software project by adding more programmers; beyond a certain point, doing so increases delays. Brooks codified that precept 35 years ago in a small technical book, *The Mythical Man-Month,* which he named after the flawed assumption that more manpower meant predictably faster progress. Today, his insight is known as Brooks' law. The book still sells 10,000 copies a year, and Brooks—who oversaw the creation of IBM's System/360, the company's most successful mainframe—is hailed as a legend. Now he's written a new book, *The Design of Design*. It's a collection of essays that extends his ideas into the fields of architecture, hardware systems, and leadership. *Wired*'s founding executive editor, Kevin Kelly, spoke with Brooks to discuss the upside of failure, lowercase letters, and what we can learn from Apple.

***Wired*:** How does a guy who grew up in the 1940s among North Carolina tobacco farmers get into computers?

Fred Brooks: I collected maps as a kid. I had tried all kinds of ways to index my map collection, which got me interested in the notion of automatic data retrieval. In 1944, when I was 13, I read about the Harvard Mark 1 computer in a magazine, and I knew then that computers was what I wanted to do.

***Wired*:** When you finally got your hands on a computer in the 1950s, what did you do with it?

Brooks: In our first year of graduate school, a friend and I wrote a program to compose tunes. The only large sample of tunes we had access to was hymns, so we started generating common-meter hymns. They were good enough that we could have palmed them off to any choir.

***Wired*:** Did you imagine all the other things computers would do?

Brooks: Oh, no. Back then computers were mostly for scientific computation, like calculating the orbits of rockets.

***Wired*:** So were you surprised by the advent of personal computers?

Brooks: I was chiefly surprised that they could become so cheap. I didn't know a single person in the computer industry who believed, five years before it happened, that we could sell hundreds of millions of computers a year.

***Wired*:** What provoked you to write *The Mythical Man-Month*?

Brooks: As I was leaving IBM, Thomas Watson Jr. asked me, "You've run the hardware part of the IBM 360, and you've run the software part; what's the difference between running the two?" I told him that was too hard a question for an instant answer but that I would think about it. My answer was *The Mythical Man-Month*.

***Wired*:** Did you ever expect it to be read by nonprogrammers?

Brooks: No, and I've been surprised that people still find it relevant 35 years later. That means we still have the same problems.

***Wired*:** What do you consider your greatest technological achievement?

Brooks: The most important single decision I ever made was to change the IBM 360 series from a 6-bit byte to an 8-bit byte, thereby enabling the use of lowercase letters. That change propagated everywhere.

***Wired*:** You say that the Job Control Language you developed for the IBM 360 OS was "the worst computer programming language ever devised by anybody, anywhere." Have you always been so frank with yourself?

Brooks: You can learn more from failure than success. In failure you're forced to find out what part did not work. But in success you can believe everything you did was great, when in fact some parts may not have worked at all. Failure forces you to face reality.

***Wired*:** In your experience, what's the best process for design?

Brooks: Great design does not come from great processes; it comes from great designers.

***Wired*:** But surely *The Design of Design* is about creating better processes for great designers?

Brooks: The critical thing about the design process is to identify your scarcest resource. Despite what you may think, that very often is not money. For example, in a NASA moon shot, money is abundant but lightness is scarce; every ounce of weight requires tons of material below. On the design of a beach vacation home, the limitation may be your ocean-front footage. You have to make sure your whole team understands what scarce resource you're optimizing.

***Wired*:** How has your thinking about design changed over the past decades?

Brooks: When I first wrote *The Mythical Man-Month* in 1975, I counseled programmers to "throw the first version away," then build a second one. By the 20th-anniversary edition, I realized that constant incremental iteration is a far sounder approach. You build a quick prototype and get it in front of users to see what they do with it. You will always be surprised.

***Wired*:** You're a Mac user. What have you learned from the design of Apple products?

Brooks: Edwin Land, inventor of the Polaroid camera, once said that his method of design was to start with a vision of what you want and then, one by one, remove the technical obstacles until you have it. I think that's what Steve Jobs does. He starts with a vision rather than a list of features.

***Wired*:** In the past few decades, we've seen remarkable performance improvements in most technologies—but not in software. Why is software the exception?

Brooks: Software is *not* the exception; hardware is the exception. No technology in history has had the kind of rapid cost/performance gains that computer hardware has enjoyed. Progress in software is more like progress in automobiles or airplanes: We see steady gains, but they're incremental.

***Wired*:** You've been involved in software for over 50 years. Can you imagine what software will be like 50 years from now?

Brooks: Nope. All of my past predictions have been, shall we say, short-sighted. For instance, I once argued that every member of a team should be able to see the code of every other member, but it turns out that encapsulation works much better.

***Wired*:** Do you have any advice for young industrial designers and software architects?

Brooks: Design, design, and design; and seek knowledgeable criticism.

Kevin Kelly (kk@kk.org) *wrote about the socialistic aspects of digital culture in issue 17.06.*