Alex Groce (agroce@gmail.com), Oregon State University

Tracy Kidder's *The Soul of a New Machine* is by far the best book "about" software engineering that was not written by a computer scientist of any kind, and does not really describe any serious *software* engineering at all. Never mind these qualifiers; no other book captures the excitement, the madness, and the *heart* of a big software engineering project as well as Kidder's story of the development of a 1970s minicomputer, Data General's Eagle. In fact, to consider *The Soul of a New Machine* only as a masterpiece of computer or software engineering is to sell it short: the book is likely the best general-audience account of the high-wire act of time-pressured modern engineering ever written.

The Soul of a New Machine has the great advantage of being written by one of the finest journalists of the last 50 years. Tracy Kidder lacks the voluminous output of a John McPhee, but shares McPhee's talent for using words with skill and charm to make any subject fascinating. Software engineers in particular should appreciate Kidder's talent (which is reminiscent of McPhee at his best) for bringing the human beings in his stories to life as powerfully as the best characters in fiction while never succumbing to the journalist's temptation to reduce every subject to a simple case of personalities. Too much "quality" journalism reduces the essence of a craft, a science, or a technology to a simple clash or harmony of characters, as if there were no objective and stubborn reality we face, but only the ambitions and emotions of men and women. Software engineering, as opposed to programming language theory, lives precisely at this intersection of human abilities, talents, and foibles and the actual "nature of the beast," the inhuman reality of the systems those human beings endeavor to transform from abstract concepts to usable artifacts. Kidder never dodges the hard task of trying to describe the *actual things* his engineers are building, and the result is readable for the average English-major reader of *The New Yorker* without boring any MIT-educated embedded software engineer in his audience.

Kidder manages to make quotations ranging from the Lord of the Rings to Macbeth smoothly integrate into a world where engineers are beginning to use the technological and process metaphors of computing and engineering to describe the real world -- a man giving his girlfriend an "Engineering Change Order" to alter their relationship or another saying "Give me a stack dump" as a way to ask for someone's thoughts. One confused engineer's limitations are aptly described: "See. He can push, but when it comes time to pop, he goes off in all directions." The connection between language, the metaphorical space of engineers, hardware, software, and engineering itself is always at play here; it is central to both the real-world story before Kidder ever touched it, and to the way Kidder tells that story. Much of the action of the book revolves around a testbed machine named Gollum. That's a detail Kidder might have invented were it not already true. *The Soul of a New Machine* deservedly won both the National Book Award and a Pulitzer Prize in 1982 for these literary qualities.

The story told in the book is driven in part by the rapidly growing centrality of software over hardware in computing: "... while the expense of building a computer's hardware was steadily

declining, the cost of creating both user and system software was rising." Software compatibility motivated the project that the book describes, at a business level, though the real motivation that animates the book is the glory of an (impossible?) challenge, something software engineers and hardware engineers have in common. One aspect of the story particularly relevant to current controversies in the software industry is the decision to run the Eagle project primarily using young engineers, who may be more likely to work "impossible" hours and can be paid less. These are engineers so happy they are given a chance to *really build something exciting* that they don't worry too much about jokes that divorce lawyers should be a company benefit, or see "You're gonna die, but you're gonna die in glory" as a threat rather than a promise. *The Soul of a New Machine* is never just a story of exploited labor, however, because both the technical beauties of a clean design and the strange pleasures of living with constraints that destroy that beauty are made real.

The best comparisons for *The Soul of a New Machine* are probably Steven Levy's *Hackers: Heroes of the Computer Revolution* and the more recent *Dreaming in Code: Two Dozen Programmers, Three Years, 4,732 Bugs, and One Quest for Transcendent Software* by Scott Rosenberg. Both are enjoyable reads, but suffer when compared to Kidder's book. Levy covers much more ground (at much greater length) but his book suffers some for these diffuse interests, never telling any single story as well as Kidder's tightly focused book. Levy's book is also written (only partly, fortunately) in a New Journalism would-be-Tom-Wolfe style that doesn't work as well for this history as it does when Wolfe turns the same methods to stock car racing or the early space program. Kidder's writing style fits this kind of technical history better. Rosenberg's book is, like *The Soul of a New Machine*, focused on a single, in principle emblematic, project, but the particular idiosyncrasies of that project, at least for me, make it less effective as a stand-in for software engineering projects in general than the Eagle hardware project. Kidder is also a better writer than either Levy or Rosenberg (no faint praise, especially in Levy's case), and this makes *The Soul of a New Machine* much more memorable than their also worthwhile books.

My particular interests in software engineering make it easy to suggest a single highlight of *The Soul of a New Machine*. Chapter 10, "The Case of the Missing NAND Gate" is the best account of debugging ever written, and that the bug is in hardware, not software, does not matter in the least. Certainly, the technical aspects of this particular bug-hunt are not as exciting as a story like Reeves and Neilson's description of "The Mars Rover Spirit FLASH Anomaly." The way the story is told, however, raises it from a single example of a process that many find to be both the most enjoyable and most frustrating part of software development to a chance for non-engineers to step partly into the trenches and *feel* that satisfaction and dismay.

Kidder, like many of the greatest journalists, knows when to give his subjects the last word. Chapter 10 ends with two paragraphs that ring deeply true to anyone who has been part of a large testing and debugging effort. Replace Gollum with one of the Mars Science Laboratory testbeds, and these words perfectly catch a feeling that came, again and again, to all of us

who tested the Curiosity rover, and likely to many readers of this column, with their own Gollums:

“They’ve reached a milestone, but one that they thought they had reached before. There’s no celebration, no sitting around Gollum with their feet up on analyzers, savoring the victory, rehearsing the battle. Plenty of diagnostics stand before them. Much trickier ones, in fact.

‘A feeling of accomplishment’ is what Veres says he has. ‘But then again, there’s lots more feeling of accomplishment to go.’”