In the 80s, Tracy Kidders The Soul of a New Machine attempted to define the story of the development of a minicomputer: from the new science to the business and nascent culture of electronic hardware and software that was characteristic of that time. Scott Rosenbergs Dreaming in Code draws on Kidders model as it attempts to document the state of software, the Internet, and everything circa 2006 through the lens of Chandler, an as-yet-unfinished software application for the management of personal information. The Chandler project--driven by Mitch Kapor, the founder of Lotus Development and main designer of its 1-2-3 spreadsheet, and later co-founder of the Electronic Frontier Foundation--isn't the primary point of Dreaming in Code, though reading about software people and their social behavior is at least as interesting as reading about that of meerkats or monkeys. Rather, Chandler is a rhetorical device with which Rosenberg takes on the big questions: How do software development teams work (or not)? Why does the reuse of software modules rarely work altogether correctly? Does open-source development by volunteers on the Internet lead to innovation or just insanely bifurcated chaos? Chandler helps his readers think more clearly about all of these issues; however, answers to these questions are, of course, not to be had, which is one of his points.

The problem with books about technical subjects that aspire to appeal to a general audience, particularly computers and software, is that such subjects are so far outside the realm of familiarity of most people that the prose bogs down in analogy and metaphor. Rosenberg manages to avoid too much of that and deliver a readable account of software development and culture. --David Wall
Santa came through this year with a slightly advance copy of Dreaming in Code, which tries to do for software engineering what The Soul of a New Machine did for computer engineering, following a single project through to its attempted conclusion. Software development is a story thats very rarely told, considering how dramatically software has changed all of our lives in the last 30 years. Author Scott Rosenberg does a good job of conveying the difficulties in software engineering, and the inevitable headaches and drama that come with incomplete plans and shifting specs (and theyre always incomplete and shifting).

Where Rosenberg went wrong, unfortunately, is his choice of project to follow. Mitch Kapors Chandler is quite atypical of software projects: its driven entirely by one mans quixotic vision, and never has to encounter the usual give-and-take with VCs or upper management that help to clarify a plan. Kapor comes off as an untethered idealist (Al Gore makes the obligatory cameo at the office), and his project is afflicted by the same we-are-the-world unseriousness as his politics. Most notably, Kapor decides there should be no central repository for data (because, hey, down with authority and all that): instead, every item will just be represented, Napster-style, across users personal computers. Its a costly decision that I dont think would have been made if it were more than just Kapor running the show.

Actually, I think the strongest part of the book is when Rosenberg abandons the project entirely in the middle section to delve into the history of the programming discipline, noting everyone from Donald Knuth to 37signals Jason Fried. Its a useful, lucid introduction to the field that contains stories I hadnt seen before.

To pick some nits, there are errors that betray Rosenberg as an outsider. Foo and bar, for instance, usually arent stand-ins for variable names, theyre stand-ins for *values*; variable names are decided on almost immediately once the need for one becomes known. [UPDATE: this caused some controversy in the comments. Its true that foo and bar can also represent variables, but I still contend that its only in theoretical discussions, when nothing is known about those variables. The book (p. 196) calls them placeholders during real-life coding, which I dont think is often true. FURTHER UPDATE, AFTER MORE COMMENTS: Okay, okay, I guess I was wrong. You guys win!] Rosenberg also, I think, makes too big a deal of softwares need for precise language: many other engineering fields, and the legal profession, require precise writing, with small errors potentially leading to catastrophe. Rosenberg also overreaches when trying his hand at software philosophy, declaring The only software worth making is software that does something new (tell that to the OpenOffice people).
All that aside, this is an entertaining book with some interesting insights, and it would be a great read for anyone who's thinking of going into programming - hopefully it won't scare them off.

For More 5 Star Customer Reviews and Lowest Price:
Dreaming in Code: Two Dozen Programmers, Three Years, 4,732 Bugs, and One Quest for Transcendent Software by Scott Rosenberg - 5 Star Customer Reviews and Lowest Price!