Should You Self-Publish a Textbook? Should Anybody?

How & Why We Self-Published *Engineering Software as a Service*

Armando Fox and David Patterson, University of California, Berkeley

In late 2011, we were in the process of reinventing UC Berkeley's undergraduate software engineering course[1] to better match industry needs and to be more rewarding for students and faculty. We had come to the conclusion that the existing textbooks targeted at such courses were unsatisfactory, and that we would therefore need to write one, mirroring Dave's experience over 20 years earlier in deciding to coauthor *Computer Architecture: A Quantitative Approach* (henceforth CA:AQA) with John Hennessy. Although the publisher of CA:AQA (Morgan Kaufmann, now Elsevier) expressed some interest, we ultimately decided to self-publish. In this article, we describe how we reached that decision and what the process has been like. Most of this information could be useful to our academic colleagues in other disciplines, although a few parts are specific to computer science.

Why Self-Publish?

We chose the self-publishing route for a variety of reasons.

High cost of textbooks. Textbook prices north of $100 are commonplace, with authors typically getting just 10 to 15 percent of the average selling price (ASP), even less on derivative editions such as translations and ebooks, while students are increasingly hard hit. Dave's experience with CA:AQA suggested that most of the really hard parts of the writing task—domain-specific proofreading, review by experts, and so on—were areas where most publishers don't add much value anyway.

Interest in ebooks. We were early adopters of ebooks and believers in their future, and we wanted to eventually create an ebook whose features could go beyond print books. For example, our book includes live links to Wikipedia entries in lieu of a glossary, live links to other interesting articles and online materials that supplement the text, and live links to online copy-and-paste code examples that match the code in the book. We spoke at length with two major textbook publishers about these ideas and discovered that they were no farther ahead than we were in thinking about fully-featured ebooks; the general reaction was "Yes, it would be great if you guys did that." We were aware of Amazon's self-publishing facility for the Kindle e-reader. Even though the Kindle is not only a hardware device but a software application that is freely available on any laptop, tablet computer, or smart phone, we were also interested in targeting specific tablets (iPads, Android) with a richer/more interactive book if possible.

Availability of affordable print-on-demand. Despite our enthusiasm for ebooks, we also believed many readers would also want a print book. Services such as Lulu and CreateSpace now support independent authors and publishers wishing to create and distribute print editions by offering print-on-demand (POD). Essentially, the authors provide PDF files of the book's interior and cover, and the POD house handles the rest. We chose CreateSpace because most authors who'd commented on indie publishing blogs rated their service and quality higher than Lulu's and because they had just been acquired by Amazon, presumably letting them best exploit Amazon's distribution and retail infrastructure.

---

[1] See http://www.armandofox.com/?p=117

[2] Covered in Section 10.5 of *Engineering Software as a Service,* in case you're interested.

Slow pace of revising print books for a software-intensive text. Dave's most recent edition of CA:AQA took nine months to appear after the time he finished his writing. Editions can therefore be revised only rarely, with extensive errata in between. Our book was going to be about the rapidly-changing field of software development, so we needed much quicker turnaround for revisions—indeed, we envisioned being able to make minor changes and have students receive those updates automatically within 24 hours. We knew that an ebook would be the only practical way to do this. We didn't realize at the time that with current print-on-demand technology, nearly the same quick turnaround is achievable for hardcopy books.

Ability to lower the price. Although we wanted the book to be low cost, we didn't intend to give it away for free. Dave's experience was that getting paid would give us a stronger motivation to keep the book up-to-date and make constant improvements (as has indeed been the case), and Armando's experience from the community theater world is that free goods tend to be dramatically undervalued ("if it was any good, they'd be charging for it") which could actually *hinder* adoption. As we learned, the economics of modern academic publishing are complex even when publishers are not part of the equation.

These desiderata made clear that we would need a "tools pipeline" that could produce both a print edition and an ebook automatically from the same source materials. Given the idiosyncrasies of evolving ebook formats and the dearth of author-friendly tools that could handle documents of this complexity, we decided that only Knuth's venerable TEX (with Leslie Lamport's invaluable macro layer LATEX) was up to the job. The requirements include typesetting a full-length book with cross-references, sidebars, extensive bibliographic entries, images that would have to be produced in multiple formats (for print vs. ebook creation), an index, and so on.

General Writing Process

Besides the topical and chapter outline, we did some "structural floor planning" to decide what *elements* the book would contain besides text sections and figures. We used CA:AQA as an inspiration for types of book elements: elaborations (additional material of interest to advanced readers but safe to skip for beginning readers), fallacies and pitfalls (non-obvious gotchas even if you've read the material carefully), sidebars, and so on. At the beginning of each chapter, we also decided to introduce a Turing Award-winning computer scientist whose work was relevant to the material of that chapter, including a quotation from that person. (We were hoping to make our readers more aware of their professional roots.) The idea of element-based design would turn out to be important since the tool chain and writing process are based around the set of book elements.

Since the input to LATEX consists of ASCII text files, we used the Git version control system with the shared-repository model to coordinate edits while writing. As it was unusual for us to be editing the same material simultaneously, conflicts were rare. We are both good writers and finicky readers and proofreaders and we have mutual respect for each others' writing skills, so editing and proofreading each others' work extremely candidly and frequently was usually sufficient to achieve good form, argument flow, grammar, and spelling. Dave would also copy and paste into Microsoft Word to take advantage of its excellent grammar checking. We had an approximate page target per chapter since we wanted to avoid the bloat of competing textbooks we didn't like, so we had to be ruthless in suggesting changes (as a shorter book also takes less time to write!). Of course, after our own editing passes, we asked multiple colleagues who were content experts to review each chapter, as we would for a paper.

LESSON: Two people write more than twice as fast as one: deadlines make you ashamed of disappointing your co-author and keep you on schedule, and each writer gets another perspective on editing, which results in far better prose than self-editing only. Alas, a major caveat is that not all writers are good editors or proofreaders!

We used the free online software project-tracking tool Pivotal Tracker (pivotaltracker.com) to track and assign tasks—coincidentally the same tool we require our students to use to track software projects. In addition to writing tasks, there were various formatting tasks: we had to do all of the print layout of complex elements like sidebars and chapter heads ourselves (see below).

For "big" sets of changes, such as fixing a set of errata or adding a fixed quantum of new content, we used the "branch per feature" model of Git: each of us would create a branch to isolate our changes from the rest of the document, and when they were properly debugged, we would merge those branches back into master. Our goal was to maintain the invariant that the head of the master branch was always "deployable," that is, ready to upload for printing.

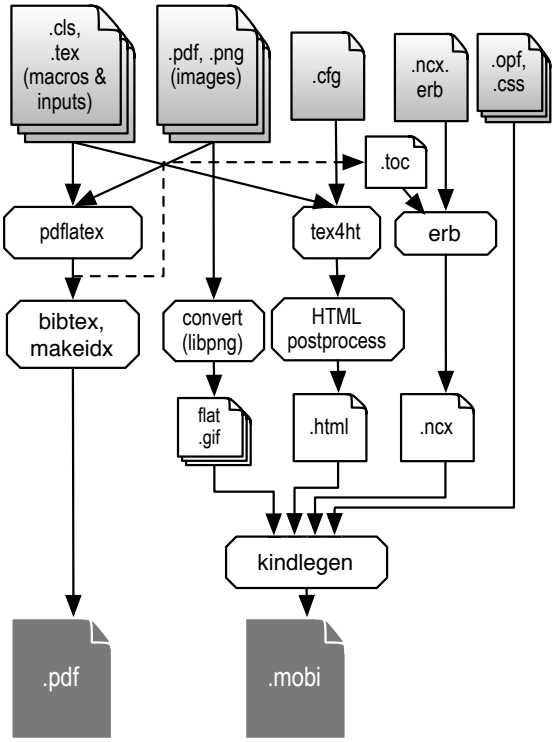
In short, we applied the same techniques to writing as we advise our students to apply to software:

- "pair writing and editing" in which we frequently exchange work and make candid criticisms;
- short iterations, turning around new prose frequently;
- lightweight project tracking with Pivotal Tracker;
- managing different members' contributions to the team project using Git branches; and
- using branch-per-feature[2] to always keep the main trunk of the project in a ready-to-print state.

LESSON: Agile book writing works pretty well, and even uses most of the same tools we teach students to use to do nontrivial software projects. We'd encourage our colleagues to adopt this workflow even if working with a publisher.

Tool Flow

The figure shows our tool pipeline, much of which should be familiar to LaTeX users; most of the custom processing described here is available on for free on GitHub as `armandofox/latex2ebook`. The shaded document icons at the top are our versioned source files; the dark document icons at the bottom are the outputs; all other document icons represent intermediate files created by the tools, whose names appear in rounded-rectangles. The key is the use of the free tool `tex4ht`, described in *The* LaTeX *Web Companion,* which generates (ill-formed) HTML from LaTeX sources. Unlike the widely-used `latex2html`, this tool has a true front-end TeX parser that can handle arbitrary control sequences. HTML is the basic input to `kindlegen`, a multi-platform program freely downloadable from Amazon's Kindle Desktop Publishing self-service site that converts input consisting primarily of HTML and images into a `.mobi` file[3] suitable for uploading to the KDP site.



[2] Covered in Section 10.5 of *Engineering Software as a Service,* in case you're interested.

[3] Mobipocket is an HTML-based, binary-archive ebook format developed by the eponymous company and later acquired by Amazon. Kindle `.azw` files are Mobipocket files with a few extra features and DRM applied; plain `.mobi` files can be directly sideloaded to Kindles or any Mobipocket-compatible e-reader.

Because book elements are the organizing metaphor, our LaTeX markup must be free of specific visual formatting: *everything* is wrapped in a high-level macro, because many macros' behavior is entirely different for print than for the ebook. For example, sidebars in the print book use LaTeX `marginpar`s, but the Kindle cannot display true sidebars, so we use a shaded box with a highlighted border[4]. Similarly, figures have different formatting requirements for print than for the ebook, and so on. Formatting is complicated by the fact that the appearance of any given element in the ebook is controlled by up to three sources: the basic macros we define, the `.cfg` file used by `tex4ht`, and the `.css` file containing the (limited) styling information understood by the Kindle.

Generating the print version is largely automatic, but experienced academic writers know that LaTeX is an unforgiving taskmaster: a misplaced $, missing close-brace, or other trivial typo can generate seemingly incomprehensible error messages. The situation is exacerbated by our complex multilevel macros: since TeX is not a true programming language but a macro expander, unexpected interactions of macros would arise, such as certain escape sequences not being allowed inside certain elements without using `\protect`, or having to play games to place elements that can only be used in "outer par mode". (You are not expected to understand this.) We found the best remedy was to build very frequently while editing to localize the cause of the most recent bug. And TeX isn't perfect; experienced users know that it sometimes gets confused about sidebar placement because of the way page numbers are calculated, so sometimes a sidebar that should have been in the outer margin ended up in the inner margin. Essentially, we had to visually spot-check the entire PDF before uploading it for printing in order to catch such problems and fix them manually. Nonetheless, once the (very complicated) macros were coded, preparing a new edition required minimal manual labor.

For the ebook, the HTML output from `tex4ht` undergoes substantial post-processing. Some of it works around bugs in tex4ht, such as beginning-of-section anchors being in the wrong place in the markup. Other post-processing fixes character glyphs that tex4ht doesn't translate quite right, replaces standard HTML markup with .mobi-specific markup such as a "start of text" marker and forced page breaks, or adds appropriate CSS classes to certain elements so that they'll display correctly on both older and newer ("KF8" or Kindle Format 8) capable e-readers.

We used OmniGraffle for drawings to create high-resolution vector images; the resulting PDF files are versioned and used directly in the print edition, and converted on-the-fly for the ebook, which requires images to be converted to `gif`, scaled, and layer-flattened.

Two additional XML files must be provided to `kindlegen`. The first is the Open Packaging Format (`.opf`) file, which contains book metadata (title, author, ISBN number, and so on). The second is the Navigation Control Format (`.ncx`) file, which contains the book's table of contents used for external navigation by e-readers, such as the "X-Ray" view on recent Kindles. These navigation anchors are filled in at build time by a Ruby script that reads the `.toc` intermediate files generated by LaTeX. The ubiquitous `make` tool ties everything together and manages the updating of intermediate files.

The book also contains numerous code examples, which we wanted to make available online in a copy-and-pastable format for reader convenience. To that end, another `make`-driven task checks which code examples have changed in each build (each code example gets its own file, even if it is only two lines long) and uses the external service API of the `pastebin.com` website to post each changed file and replace the link URL in the LaTeX file in which the code appears.

---

[4] We learned that technique and many other useful tricks from *Kindle Formatting: The Complete Guide to Formatting Books for the Amazon Kindle* and *Graphics on the Kindle* (a now-obsolete and out-of-print Kindle title). Amazon has since made most of this information available free.

LESSON: Writing a book with this tool flow is as much like programming as it is like writing. No GUI tools existed to do what we needed, but the Unix programming model allowed us to build them from simpler parts. Tools aside, "macro-based" writing is a minor adjustment for academic writers accustomed to LaTeX, a moderate adjustment for WYSIWYG users who are disciplined about using the Stylesheets feature of Word, and a ***world of pain*** for most other people who think in terms of applying visual styles like "larger font" and "italic" in WYSIWYG word processors.

Making it look good: Design, Typography, Indexing

Since publishers are increasingly outsourcing their traditional services such as design, translation, and indexing, we figured we could hire those freelancers just as well. Our professional connections in the publishing and design industries allowed us to retain a talented graphic designer who was also technically proficient enough to translate his visual designs into technical terms. He designed the overall look and feel of all the graphic elements of the book (icons, cover, Kindle look-and-feel, and so on) and has also designed the book's web site and publicity materials to match. When Amazon improved the Kindle format to support richer visual features, our designer used branch-per-feature to make extensive formatting and CSS changes to the Kindle Edition while work continued in parallel to index the book.

Indexing is an editorial skill we lack, so we hired a professional indexer to create a full index after the fact. Although LaTeX can typeset the index automatically if the terms to be indexed are marked up in the source files, our indexer wasn't familiar with this format; he used his own tools to create the index and delivered an XML document containing the index. Since we needed the index to be automatically created each time the book was modified or expanded, we sorted the indexer's XML file by chapter and parallelized the work of "backporting" the index annotations into the LaTeX sources across half a dozen additional freelancers familiar with LaTeX and Unix tools, again using Git branch-per-feature to avoid colliding edits. As a result, the index is now 100% automatically generated but its entries are based on the editorial judgment of a professional. We are exploring ways to improve the tool flow so that we can continue to work with him.

Our drawings were simple, so we did our figures ourselves in OmniGraffle, but it is certainly possible to outsource that work as well if your book needs it.

To date we have spent a total of about $10,000 of our own funds on graphic design (although this includes the ultimately failed attempt to produce an iBook, as described below) and about $3,000 on indexing, for a 500-page book.

The default formatting provided by LaTeX, especially the default body font Computer Modern, is distressingly reminiscent of many academic papers. We settled on the stalwart Times for body text and Futura for display fonts, based on Web articles containing typesetters' advice on which display fonts go well with which body fonts, and did the necessary low-level black magic to make nonstandard fonts work with LaTeX. If you're wondering why you should even care about such matters, it's because they can make the difference between a book that looks dowdy and one that is visually attractive for the reader to spend time with.

LESSON: Design, typographic choices, indexing, and other "collateral" tasks should really be handled by professionals, or at least strong aficionados (e.g. as Armando is a font-and-typography geek).

Marketing

Traditional publishers maintain publicity mailing lists, set up booths at conferences and trade shows, and send salespeople to university campuses, all of which efforts can usually amortized across multiple books.

We quickly learned the hard way that setting up tables at conferences doesn't amortize well if you have only one book to sell, as it's hard to attract people. But as academics, we were already being invited to give talks at other universities on how we created the course and the MOOC, which gave us opportunities to expose the book too: we'd usually travel with a few "comp" copies. Of course, "comp" copies for self-publishers means we pay for these out of pocket, but we gave away a great many, since our faculty colleagues are the decision makers and having a physical copy of a book makes it harder to ignore on your desk, our enthusiasm for ebooks notwithstanding.

Although we didn't know it when we started the book (mid 2011), we were about to be offered the chance to adapt the first half of the campus course to a MOOC (Massive Open Online Course), which turned out to play a major role in the textbook's development. We accelerated our writing in order to have an "alpha edition" consisting of half of the content, that is, the chapters that would be most useful to the MOOC students. Indeed, based on advice from colleagues who had offered MOOCs, we were already structuring the MOOC as short video segments interspersed with self-check questions and assignments; we decided to mirror that structure in the book, with each section in a chapter mapping to a topical segment in the MOOC. While the book was only recommended (not required) for the MOOC, the MOOC was instrumental in increasing the book's visibility, as we describe in our technical report *Software Engineering Curriculum Technology Transfer: Lessons Learned from MOOCs and SPOCs*[5].

We also spent about $2,000 to purchase mailing lists from contacts in the publishing industry; Armando had learned from his experience on the board of a community theater that a combination of email plus postcards works better than either one alone, so we had our designer create postcards to match the book's look and feel and we did a combined postcard-plus-email campaign.

We've learned two things through our "marketing" efforts. First, textbooks require domain-specific marketing: you have to identify and reach the very small set of readers who might be interested, so "mass" marketing techniques don't necessarily apply. Second, in the past, publishers were the main source of information about new books, so your competition was similar titles from your publisher or other publishers; today, the competition has expanded to include the wealth of free information available online, an enormous used-book market, and so on, so the build-up may be slower, and indeed the "800 pound gorilla" model in which one book dominates a field may not be an option for new arrivals.

One final note concerns the importance of Amazon reviews. When we released the Alpha Edition, it was priced even lower because over a third of the content had not yet been written (we wanted to release something in time for our MOOC, to get feedback from those students as well as our on-campus students). Despite repeated and prominent warnings in the book and in the Amazon blurb about this fact, many readers gave us low reviews because content was missing. We later decided based on reader feedback to change the book's title, which also required changing the ISBN number and opening a new Amazon product listing, which "broke the chain" of reviews of previous editions and wiped the slate clean. This turned out well, since the vastly improved Beta edition received higher than 4.5 stars out of 5 from Amazon readers, a high review level that has continued into the First Edition, while our main established competitors have far lower Amazon reviews. For better or worse, even people who purchase in brick-and-mortar stores rely on Amazon readers' reviews, so it was good to start from a clean slate after extensive changes. The one downside was that this also "broke the chain" of free updates for Kindle buyers; very few buyers complained directly, and we arranged to gift new copies to those who did.

LESSON: We've found marketing to be the biggest challenge for self-publishing, and the largest potential remaining advantage of traditional publishers today. While it's obvious that we can produce a book far more inexpensively and quickly ourselves, it's less obvious whether the book will be as popular as if we had gone with a traditional publisher. We may know in a few years if we're successful as self-publishers; if the book never becomes popular, we may never know.

---

[5] UC Berkeley Technical Report EECS-2014-17,
`http://www.eecs.berkeley.edu/Pubs/TechRpts/2014/EECS-2014-17.html`

Piracy

Even though we never expected to make very much money from the book, it's extremely demoralizing when others steal your content. Dave had had plenty of experience with this with CA:AQA, with the most common mode of piracy being someone scanning every page of the print book and posting the scans online. (Indeed, recently the publisher of CA:AQA had to send a takedown notice to a *professor* who had done this for his/her students.)

Our initial naïve hope was that a low price would discourage piracy—$10 for the Kindle Edition, $20 for the Beta version of the print edition (we raised the price once it was out of Beta), and (as of early 2014) when buying the print edition you get the Kindle edition free, with Amazon's new MatchBook program. In many but not all countries where they do business, Amazon gives Kindle authors a 70% royalty on books priced below a threshold value, which is US$10 or its equivalent everywhere except India, where it's substantially lower (399 INR=about US$6.60).

However, when we informally surveyed thousands of MOOC students in mid-2012, nearly 30% anonymously self-reported having "found" a copy of the book online. There are sites that charge a membership fee for unlimited access to pirated PDF and ebook files, located in Russia and other interesting places. Amazon KDP was shocked to hear this, which surprised us given Dave's experience.

We conclude that if piracy occurs at a price of $10 including free updates for life, it's unlikely that *any* price will stop piracy. As one of our readers from India commented, the book costs the same as going to a movie, which we assume is not very onerous for students who have sufficient access to PC's and the Internet to do the coursework at all. We briefly considered and rejected the idea of planting "broken" copies on other sites to make it harder to find a good copy, as music publishers allegedly tried with MP3 files in the early days of Napster; we preferred to concentrate our efforts on improving the material.

Publishers and distributors have long struggled with both outright piracy and the "gray market" of used books. Amazon now rents Kindle Editions of expensive ebooks; publishers have long used the revision of new editions as a way to dry up the used-book market for older editions; brick-and-mortar campus bookstores now rent textbooks or buy them back at reasonably generous rates. Since independent publishers can do none of these things, we decided to look for other ways to add value to the book.

The book already "came with" a free MOOC delivered through edX, but for many reasons, we had no desire to "monetize" MOOC students directly, and not just because it would have contravened the institutional agreement between UC Berkeley and edX. Instead, we viewed the MOOC (which remains free, and has been taken by over 100,000 students from over 120 countries) as a way to get exposure for the book, which is *recommended but not mandatory* for the course.

Most recently, Amazon, which has been extremely progressive in helping us explore new marketing and distribution models, agreed to try a pilot in which purchasing the Kindle book (whether standalone or as part of a bundle with the print book) also gives the buyer a $10 credit on Amazon Web Services, which coincidentally can be used to do the exercises in the book. Amazon had developed all the necessary fulfilllment infrastructure to do this; they get product placement with a very targeted audience, and we get a value-add commodity bundled with the book that essentially means that if you're willing to try using AWS, you get the book for free, including free updates for life. We are working with other companies on similar arrangements: if the value of the bundled "freebies" equals or exceeds the nominal cover price of the book, it may reduce the desire to pirate the bits.

Finally, we are sympathetic to readers in developing countries for whom US$10 is a lot of money, or where Amazon doesn't do business at all and the Kindle version is unavailable. During the course of the MOOC, Armando created watermarked versions of the book individually for over 200 students based on their email requests and sent them out for free after extracting a promise from the recipients that they wouldn't post their copy online. (We haven't followed up to see if anyone has done this; Armando doesn't really want to know.) This approach isn't scalable, but we also had readers offering to pay higher

than list price in order to subsidize the purchase for someone who couldn't afford it. We would love if there were "book scholarship" matchmaking service that could solve this problem.

LESSON: As the music industry and others have learned, if most of the value is in the bits themselves, your economic model is precarious. Try to provide value in the ecosystem around it—in our case, the MOOC, AWS credits, and so on. The bits *will* get copied, and even with these measures, the residual piracy rate will probably never get below 10%.

Beyond the 1st Edition: Translations & Maintenance

As the MOOC grew in popularity alongside our campus course, a few leaders emerged as "early adopters" of the material in both their campus classrooms and in offering to improve and steward the online materials. One of them was so valuable that we eventually offered him the formal position of Editor of the book, to help us manage not only the MOOC but the community of instructors using the book in their classrooms and the developers contributing to the online course materials. His compensation is formal recognition in all book- and course-related materials and a share of the royalty from book sales.

We are now exploring a similar model for translations. In the special case of China, we had a professional colleague who facilitated a relationship with a highly-regarded Chinese publisher who is managing the entire process as a traditional publisher would (and paying us the same low royalty most publishers would), but we are exploring models for other translations in which the translator gets a small advance (perhaps to hire some student helpers) and gets a much larger royalty share on sales of the translated edition, motivating them to help us market and minimizing our exposure if the translated version doesn't sell.

For that matter, the concept of "edition" is fuzzy. Our book has version numbers, which started from 0.8.0 and are now at 1.0.1. The first digit will mark "major" editions with huge content changes, the middle digit will mark significant revisions or new content within the existing content framework, and the final digit will mark minor revisions, errata, code example changes to keep up with the changing software ecosystem, and so on. We don't expect a Second Edition (2.x.x) for three or four years, but we expect many "steppings" of the First Edition (1.x.x) over that same period.

Retrospective: Did we meet our goals? (✔, ✖, ?)

✔ **Low price and wide reach.** Our Kindle version costs $9.99 in an era where $100 printed textbooks and even $100 e-textbooks are common. In addition, ebook distribution lets us easily reach readers in countries where distribution of a print book might be impractical or too expensive. As our book ended up being used in a MOOC (Massive Open Online Course) that has had over 100,000 students from 120 countries, this expanded reach was important to us. Amazon is now selling the bundle of the print book *plus* an ebook for a total of $28, which in our view is quite a bargain.

✔ **Fast turnaround**: We've revised the book about 12 times since January 2012. Kindle edition owners get the updates free within 24 hours or so. For the print book, within 24 hours of our pushing changes to the print version, the next copy purchased will reflect those changes.

✔ **Free updates**: Originally we worked with Amazon's "white gloves" author team to manually push changes to Kindle Edition purchasers, but as of mid-2013, Kindle AutoUpdate automates this process with no Amazon intervention.

✔ **Reasonable author income without reader hardship:** Amazon incentivizes Kindle authors to keep prices under $10 by offering a 70% royalty in some countries, so we get $7 on a Kindle book sale in those countries, $3.50 in other countries, and about 50% of the ASP on the print edition in Europe and the US, and 25-30% in other countries (see "International Distribution" below). As instructors sensitive to student

needs, a "scholarship" program that allows interested buyers who can afford to do so to "subsidize" the purchase of a book for a deserving student who can't afford it would be most welcome; perhaps one of the MOOC providers such as edX or Coursera could take the lead on this.

✔ **Bundling print + ebook conveniently:** Surveys of MOOC students confirmed our belief that most people would want to purchase both the print book and ebook, which became easy when Amazon started its MatchBook program in late 2013.

✖ **International distribution of print edition.** For Europe and the US, we steer readers to purchase on Amazon: since Amazon owns CreateSpace, which has facilities in those areas, readers get delivery times of a few days. For other parts of the world, CreateSpace contracts manufacturing out to LightningSource, a POD network that works with the worldwide book distributors Ingram and Baker & Taylor. Those customers can only order through retail channels that have a relationship with one of these distributors (virtually all bookstores do), and we get a much lower royalty.

? **Marketing/adoption**. The book is profitable but penetration into academia is slower than it was for CA:AQA in 1990. On the other hand, much has changed in the textbook ecosystem since then, so even if we achieve comparable popularity, we may never know all the reasons for the slower build. While there is fierce competition from open content, we believe a textbook as a narrative created by an informed tour guide still has considerable value.

✖ **Rich "tablet edition."** Our original ebook vision included interactive questions, links to Wikipedia, "live" coding windows embedded in the textbook, video screencasts embedded in the book (they're currently hosted separately on Vimeo), and so on. Since the Kindle platform was very limited (and still is, compared to full-blown tablet applications), we sank thousands of contractor dollars into an HTML5+JavaScript edition that used PhoneGap to deliver richer features, but it was slow and unstable. Then Apple released iBooks, and mandated that all iPad titles be delivered that way. Besides meaning that we would now have to use different authoring frameworks for Apple vs. non-Apple devices, iBooks had zero automation: even though we had complete control over exporting our book content in any imaginable data format, iBooks could not import anything except Microsoft Word documents. Thus, we spent several thousand dollars more to do the manual-labor-intensive port of our text into iBooks Author, and then several thousand dollars on top of that when we added substantial new content. The results looked fine, but we weren't selling nearly enough iBooks to justify the expense, so we killed the iBooks version when we went to the Beta Edition. (To our surprise, another disappointment was that iBook sales were limited to the US.)

✖ **EPUB edition.** We also originally targeted the Barnes & Noble Nook e-reader to hedge our bets. That reader consumes the EPUB format, a close relative of MOBI (the format that's the basis of Kindle books) that was easy for us to output. But Nook Stores sales were nonexistent—perhaps 50 copies over several months, a tiny fraction of Kindle sales—and despite our best efforts, we were unable to decipher how and whether we could use Google Play to distribute EPUB books for other readers. We ultimately decided that Amazon had decisively won the e-reader format war because of the Kindle's superior buying and reading experience, so we killed all ebook versions other than Kindle. The revised Kindle Format 8 adds a lot to what can be rendered, and doubtless Amazon will continue to improve the format to exploit more capable tablets (such as the Android-based Kindle Fire).

Conclusions: Pros and Cons of Self Publishing

Should you self-publish? Should anyone? Certainly, you can do some great things by self-publishing:

+ Keep the price much lower than a publisher would, yet get a reasonable return (~50% of ASP in most cases).

+ Rapidly deploy new editions and errata corrections.

+ With the right help, get the same quality of design, indexing, and proof checking as a publisher would provide. As independent authors, we'd love better tools to collaborate with professionals such as indexers.

+ To their credit, Amazon has generally been very progressive and responsive in exploring new business and publicity models with us. Perhaps they believe this is the future of academic publishing. We looked at boutique publishers such as Inkling, but they appeared interested only in dealing with major publishers.

Reasons not to do it:

– Even more work than writing a book with a publisher, and you have to be self-motivated (or ideally, work with a partner so you keep each other on schedule).

– Marketing is an unsolved problem, but there are several hopeful paths to explore. MOOCs and SPOCs—Small Private Online Courses, in which individual instructors customize private copies of MOOC materials for use in their own classrooms—are certainly are one of them.

– Piracy is an unsolved problem for everyone, but bundling services with ebooks may prove to be a successful deterrent.

– The current tools aren't terribly author-friendly if you're not familliar with LaTeX. But maybe you can hire a freelancer to help with that as well!

In various other articles about our experience with our MOOC, we have written that the future of curriculum technology transfer may be a combination of MOOCs/SPOCs and ebooks. Perhaps the future of textbook publishing is similar, or perhaps textbook publishing and curriculum technology transfer are finally converging in a way that aligns with the goals of authors and instructors rather than relying on scarcity as part of the economic model.