LEAN PUBLISHING

Peter Armstrong

http://leanpub.com/lean

Lean Publishing

Peter Armstrong

This book is for sale at http://leanpub.com/lean

This version was published on 2013-03-04

This is a Leanpub book. Leanpub empowers authors and publishers with the Lean Publishing process. Lean Publishing is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

©2010 - 2013 Peter Armstrong

Tweet This Book!

Please help Peter Armstrong by spreading the word about this book on Twitter!

The suggested hashtag for this book is #leanpublishing.

Find out what other people are saying about the book by clicking on this link to search for this hashtag on Twitter:

https://twitter.com/search/#leanpublishing

For Caroline and Evan

Contents

Introduction

Q. What do you call an outdated, unimportant book?
A. Who cares!

This happened to me.

My first book *Flexible Rails*, a computer programming book about how to use two relatively new technologies (Adobe Flex and Ruby on Rails) together, was fairly successful in its niche. So, after completing it in late 2007, I got extremely excited. I signed not one but two publishing contracts with my publisher, and I proceeded to start writing two books. Both books failed, but in different ways–it could be that every unhappy book is unhappy in its own way.

The first of my two new books, *Enterprise Flexible Rails*, was going to be a sequel to *Flexible Rails*. It was going to pick up where the first book left off, and go in the direction of using a framework instead of writing all the code manually. This framework was being written by my new startup, Ruboss (yes, I immodestly named it to be reminiscent of JBoss), so not only was the book going to be a profitable sequel, it was also going to launch my startup's product into the world.

The book lasted two chapters before my publisher decided that my plan was not a good one. Since I refused to change the plan, we agreed to cancel the book and refund everyone's money. This was a relatively quick, ignoble failure.

The second of my two new books actually had the distinction of failing *twice*–first as *Hello! Flex 3* and then as *Hello! Flex 4*. This book was going to be a fun introductory programming book about one of the two technologies (Flex) that I had covered in my first book *Flexible Rails*. My idea was for it to be *Flexible Rails* minus Rails plus cartoons: the publisher wanted to create a new illustrated series of introductory books, and I knew J. D. "Illiad" Frazer, the cartoonist who drew *User Friendly*. I made the introduction, and a book series was born–one which continues to this day.

After I got about six chapters into *Hello! Flex 3*, the publisher and I realized that the format of the book required an almost complete rewrite, as I needed to make the code examples more self-contained, and make other changes. (Hooray for being the first author of a new series.) Since I was rewriting the book anyway, I decided to also target the next version of Flex (Flex 4): during the writing of the first six chapters and experimenting with the format of the *Hello* series, Flex 4 had been announced. In the computer book business, books get obsolete quickly!

So, I took the 6 chapters of *Hello! Flex 3* and used it as the basis for only one chapter (the last, most advanced one) of the new *Hello! Flex 4* book. I then worked day and night and rewrote the rest of the book from scratch over a 3 month period. We launched it after two beta versions of Flex 4 had been released. I was excited to be the author of the first shipping introductory book about Flex 4. With the release of Flex 4 right around the corner, great things were ahead for my book.

The book was obsolete before the ink was dry.

Instead of Flex 4 being launched with essentially bug fixes, Adobe made a totally minor and almost pointless change, renaming something called an XML namespace. This change broke *every single code example* in my book, *2 weeks* after the book went to print. Sure, I immediately updated all the book code that was downloadable, wrote a blog post explaining the differences, etc. It didn't matter. The book was dead in 2 weeks.

I had gone from being too late (as essentially the last author of a Flex 3 book) to being too early (as the first author of a shipping Flex 4 book, just slightly before Flex 4 was stable enough to print things about). Venture capitalists often talk about how timing is so important for their startup companies. It turns out this is true for books as well, and that I had managed to be too late and too early–with the same book project!

Even worse, it turned out that Flex 4 didn't matter. Apple essentially killed the Flash platform by not allowing it on iPhone, and Flex was a developer-focused way of creating Flash apps. So not only was my book about Flex 4 dead on arrival, Flex 4 *itself* was dead on arrival.

I had written an outdated book on an unimportant topic, and nobody cared.

After having written a successful first book, I had started a startup about a framework, and had started two technical books. All three projects were miserable failures. I had gone from being someone who would be flown to Pisa to talk at a Rails conference to someone who, professionally, essentially had nothing but a failed startup and two failed books.

Worse still, was the irony: I had *already figured out* an approach which would have helped me avoid all of these issues–and then proceeded to not use it again. With my first book, *Flexible Rails*, I had come up with a fairly unique approach. I launched the book early, in self-published form, iterated in public with feedback from a small but growing and dedicated community of readers, and evolved my book into a finished, good quality book with traction, one that multiple publishers made offers for. I had stumbled into an excellent workflow for writing and publishing in-progress ebooks, and into an understanding of some of the deeper principles around publishing in-progress books.

After learning all those lessons, I then proceeded to ignore all of them. Looking back, I hadn't really realized the significance of what I'd done. It had just seemed like the right thing to do at the time.

I just hadn't suffered enough yet.

Only once I had spent about a year spectacularly failing at *Enterprise Flexible Rails*, *Hello! Flex 3*, *Hello! Flex 4* and pretty much everything else I touched, did I start to understand. I realized that this wasn't just suffering that was unique to me, but that the problems I had encountered were endemic to the entire computer programming book industry. As I learned more, I realized that the new Lean Startup techniques which had been developed and popularized by Steve Blank and Eric Ries applied to publishing, and actually dovetailed nicely with the experiences I had writing *Flexible Rails*. I also figured out that they didn't just apply to computer books, but to business books, other non-fiction books and, as I learned more about how serial fiction had functioned in the 1800s, even to fiction.

Lean Publishing was born.

I wrote the first version of the Lean Publishing Manifesto[1] in 2010. I proclaimed "A Book is a Startup"[2]. This idea resonated with many people, including Todd Sattersten–his book entitled Every Book is a Startup[3] was strongly influenced by my Lean Publishing Manifesto. We even gave a talk together at a Seattle unconference about the idea.

I also rebooted my startup, Ruboss. Instead of trying to sell a framework, we decided to focus on Lean Publishing. (This was not a "pivot"; this was a reboot.) My original cofounder (and the principal author of the Ruboss Framework) left, and the Ruboss Framework was renamed to RestfulX. My employee #1, Scott Patten, became my new cofounder and together we created Leanpub. Leanpub is our way of implementing the Lean Publishing ideas in this book.

Over the past few years, I have learned a lot of lessons about Lean Publishing from helping build Leanpub, from listening to our customers, and from reflecting on my previous successful and unsuccessful experiences. This short book is the result of these few years of thinking and seven years of experiences. Regardless of whether you are an author or a publisher, I hope that you find this book interesting. I also hope that the Lean Publishing approach saves you some suffering and helps you chart a course ahead in these interesting times.

[1]https://leanpub.com/manifesto2010
[2]https://leanpub.com/manifesto2010#a-book-is-a-startup-four-parallels
[3]http://shop.oreilly.com/product/0636920021261.do

Definition

Lean Publishing is an attempt to solve the following problems, which are shared by authors and publishers:

- How do we make the best book?
- Will anybody care about the book?
- How will people find out about the book?
- When will the book be done?
- Will the book be released while it's still relevant?
- How should we price the book?
- How do we produce ebooks that look good on all ebook readers?
- How do we do all this and still produce a great looking print book?
- Should we even make a print book?
- Do we really need to use Word?

Some of these questions are timeless, others are recent inventions. For example, "should we even make a print book" wasn't even something that could be asked twenty years ago. It wasn't that it was a stupid question, it was that the question could not even be formulated.

So, Lean Publishing is an approach which attempts to solve these problems. Sounds great – but what *is* Lean Publishing?

The definition of Lean Publishing is:

> **Lean Publishing is the act of publishing an in-progress book using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.**

This is an ambitious and inclusive definition. It's also a mouthful. In this chapter, we'll unpack this definition, and consider each of the ideas in turn.

Lean Publishing is the act of publishing an in-progress book…

Why would you want to publish a book while it is in-progress?

The fundamental reason is this:

A book is a startup.

There are four parallels to consider when comparing books and startups.

1. Risk: There are market risks, technical risks and a very low probability of success.
2. Creative: Both writing a book and creating a startup are highly creative processes undertaken by one or a few people working closely together.
3. Stealth: Historically it has taken about a year, often spent in isolation or "stealth mode", to develop and release the first version.
4. Funding: Historically, startups have been funded by VCs and authors have been funded by publishers, both of which are hit-driven businesses.

Let's look at each of these parallels…

Risk

Doing a startup is a highly risky endeavor. There are market risks (is there a market for the product? will some other company beat us to market?) and technical risks (is the product even possible? will it be any good?). Together with all the other ways a startup can fail (founder issues! lawsuits!), these risks ensure that there is a fairly low probability of success – the standard optimistic calculation is that 10% of venture-backed startups succeed, 20% limp along, and 70% catastrophically fail. The founders take all these risks because of the potential for a huge payday and to change the world – this has been referred to as "changing the world, one million dollars at a time."[4] If you've never done a startup, the best introduction to the lifestyle is to read *Founders at Work* and Paul Graham's essays[5].

Writing a book has a similar set of risks. There are market risks (will anyone want to buy the book? will someone else write a similar book first?) and technical risks (do I have what it takes to finish a book? can I write a book that is entertaining and engaging?). The author takes these risks for the possibility of fame and fortune – albeit a much smaller amount than can be acquired by building a successful startup.

All things considered, it's completely unsurprising that the probability of success for both startups and books is so low.

Creative

There is a reason that there are no startups founded or books written by a hundred people: doing anything creative with that number of people produces a product in which any brilliance is diluted. (This is the reason why the phrase "design by committee" is not a compliment.)

A book or a startup is best created by 1 or 2 people, who are the authors or founders. You can create a book with 3 or 4 authors, but essentially all the great books have been written by one author. In fact, if you have more than 4 authors, you're not even really producing a book – you're really producing an anthology of individual essays. Similarly, a startup typically has 2 cofounders (Apple had Steve Jobs and Steve Wozniak, Microsoft had Bill Gates and Paul Allen, etc.). You can create a startup with 3 or more founders (BEA had three: B, E and A), but that's the exception – and it's extremely rare for any successful startups to have more than 4 founders.

There is actually an interesting, subtle difference between the ideal number of authors and founders: to achieve great results, it seems the ideal number of authors of a book is 1 and the ideal

[4]http://www.wired.com/wired/archive/4.05/gassee_pr.html
[5]http://paulgraham.com/articles.html

number of founders of a startup is 2. My personal take on the reason for the difference, as an author and a startup founder, is that writing a book is essentially the technical portion of creating a startup – a solitary endeavor which requires long periods of sustained thought. (Also, in writing, having one voice in the book is essential.) However, creating a startup is a more multi-dimensional activity: besides the solitary time spent at your computer creating the product, there are a vast number of activities (fundraising, reacting to the market, acquiring customers, getting feedback, the list is endless) which are too much for almost all individuals to sustain sole responsibility for, over the time it takes to develop a product.

Stealth

In ancient times (up to and including the dot com bubble), startups traditionally operated in secrecy from the moment of being founded until the product launch. In this time, the founders and the early employees rapidly and secretly build the product envisioned by the founders in order to get it to market before anyone else figured out what they were up to. This was called "first-mover advantage." There was typically very little outside feedback sought until the product was ready for beta testing, which was usually very shortly before its launch. The intent of beta testing was to catch any small wrinkles, not major deficiencies.

A similar situation exists for authors. Many authors will work on their manuscript in isolation until it is complete, before submitting it to publishers either themselves or via an agent. Alternatively, authors sometimes submit book proposals to publishers, seeking a contract for a project before beginning serious writing. In both cases, the manuscript has little contact with its true customers – readers – before it is largely completed. Traditionally in the technical book space, only a handful of readers are brought in as reviewers before a book goes to press.

Funding

Typically, startup founders spend much of their stealth-mode period not only in building the product, but also in raising money from angel investors (accredited wealthy individuals) or Venture Capitalists (VCs). While many startups are bootstrapped by the founders (either by their savings, by doing consulting on the side, or both), the goal for almost all startup founders has been to get funded.

Similarly, the goal for most authors has been to get a publisher. Historically, publishers have been needed by authors not only because of their control of printing presses and access to distribution channels, but also as a source of funding. In the 20th century, a publisher often paid an advance against royalties, which the author typically used to replace some of the lost income that s/he would have earned during the time spent writing the book. Since the prospect of repaying an already-spent advance is distasteful, this typically gives the publisher a lot of leverage over the author in terms of the content of the book. This can be a good thing, since it motivates authors to actually finish their books as well as to listen to the good suggestions from their development editors, but it also can lead to a book being finished prematurely or at a lower quality.

So, in both cases, the investors (angels / VC / publisher) end up having a significant effect on the startup or book that they are funding. This is entirely unsurprising, since people naturally want a sense of control over what they have bought. If you were a publisher or VC and you had an author or founder making what you felt were bad choices, chances are you'd have something to say too!

Because of the influence that investors in a startup or book have over the output, it helps to understand their goals.

The most important thing to realize is that both publishing and venture capital are hit-driven businesses. VCs encourage their portfolio companies to swing for the fences, as they need the huge wins to make up for the majority of their portfolio companies that fail. The prospect of doubling their money is never the reason a VC invests in a startup.

Similarly, publishers produce many books, but get the bulk of their profits from only a handful. Once these books are developed and the marketing starts, the publisher gets a better sense of which books (if any) have the potential to be breakout successes, and focuses their marketing efforts on those ones. Since publishers want to produce hits, they tend to force all books to seek as wide an audience as possible. When done badly, this can distort what made the book good in the first place, and bore and alienate some readers.

Consequences

When you consider these four factors together, the consequence is that it's very easy to create something that nobody wants.

The usual outcome of the above is as follows: with a startup or a book, what typically happens is that the founders or authors get some money, lock themselves in a room and slave away to produce something targeted at a very broad market (it has to be a hit, remember?), which the market may or may not want. If they get the product slightly (or completely!) wrong the first time, the money (the advance or seed funding) is mostly (or entirely!) gone and there's not enough money left to buy the time to iterate enough times to actually produce a product that people want.

For the past few decades of startups, and much longer for books, the methodology outlined above has actually been the state of the art. VCs and publishers have used the models described above since there hasn't been any credible alternative articulated until recently. VCs and publishers may be many things, but stupid is usually not one of them. The above approach seemed to be the best process available.

After all, startups are hard – surely this kind and rate of failure is inevitable, right?

Maybe not.

Customer Development and The Lean Startup

Recently, the startup community has been energized by new ideas about how a startup should be done. This thinking originated in Steve Blank's ideas about Customer Development, explained in his seminal self-published book *The Four Steps to the Epiphany*. Steve Blank's ideas were applied by Eric Ries to the problems of creating web 2.0 startups, resulting in Eric Ries' idea of The Lean Startup.

Briefly, the concepts of Customer Development and the Lean Startup involve launching the product extremely early (so early that you're embarrassed by it) and iterating rapidly in response to customer feedback. These iterations follow the OODA loop (Observe, Orient, Decide and Act) of USAF Colonel John Boyd, applied in close consultation with customers known as "earlyvangelists" who will be the early adopters of the product and who will evangelize it to others.

The Steve Blank and Eric Ries intellectual connection is more than the standard mentor relationship: Steve Blank actually funded IMVU, which Eric Ries was the CTO and cofounder of, on condition that Eric Ries take his class about Customer Development!

Since writing books and doing startups have so many parallels, understanding how the Lean Startup ideas apply to startups helps understand how the Lean Publishing process applies to writing and publishing books. So, Eric Ries's[6] and Steve Blank's[7] blogs and books are an excellent resource for all writers, not just for people doing startups. This is especially true since both Eric Ries and Steve Blank are excellent and entertaining writers – some of Steve Blank's war stories[8] are epic.

The Lean Startup approach of doing Customer Development and getting meaningful feedback from early customers is very similar to the process of releasing a book very early in the writing process and getting meaningful feedback from readers. In both

[6]http://www.startuplessonslearned.com/

[7]http://steveblank.com/

[8]http://steveblank.com/2009/05/13/gravity-will-be-turned-off/

cases, if the Lean approach is done well, the startup or book will evolve in ways that are not necessarily planned from the outset. (This is similar to the concept of Agile software development, but the scope is actually broader in that the entire direction of the startup or book can change. Agile software development assumes that the customer knows what the real requirements are.)

Now that we've looked at the parallels between books and startups, seen how the startup metaphor for writing a book is instructive, and considered Lean Startup theory, we can continue understanding the definition of Lean Publishing. Since we haven't seen it in a few pages, recall that the definition is:

> **Lean Publishing is the act of publishing an in-progress book using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do**.

Let's consider the reamining parts now.

…using lightweight tools…

Why are lightweight tools important? Simple: if the tools are "heavy" and cumbersome, there will be more friction in the process of releasing versions of a book. If you need to typeset every book version which is released, as well as getting approval from development and/or technical editors before releasing a version, it can take days or weeks between versions. The lighter and more automated the toolchain, the faster these releases are possible.

If the toolchain is fully automated, it is possible to release a new version of a book *5 minutes* after the author is happy with it. This isn't just possible in theory: this is what we do every day at Leanpub. And in the world of computer programming book publishing, leading publishers such as O'Reilly have built their own toolchains with similar goals.

The driving force behind this is the common recognition that a toolchain must be lightweight and automated for releasing books in-progress: if every version of an in-progress book needed to be manually typeset, the time and cost would be prohibitive – even with outsourcing.

…and many iterations…

This idea comes from the world of Open Source Software. Eric Raymond has explained the development process used by Linus Torvalds in developing the Linux operating system kernel as follows:

“Release early. Release often. And listen to your customers.”[9]

As this fantastic essay explains, the idea that rapid, sometimes daily, iterations on something as complex as a computer operating system kernel could occur was revolutionary. If it’s possible to do iterative development on something as difficult as rocket science, surely it is possible to do this with a book?

However, over the past hundred years, it has been considered to be extremely difficult to iterate on books. Typically, the only iterations that were produced were subsequent editions, and those editions only get produced for successful books. Primarily, this is an artifact of the fact that books were first completed and then printed, put on trucks and sold in stores. With that workflow, it has obviously been very difficult to iterate on an unfinished or unsuccessful book, in a way that distributed these iterations to its readers.

Lean Publishing is based on the premise that the inability to iterate on a book is a historical problem which has been overcome by ebooks, provided that publishers are willing and able to generate versions via a lightweight process and to solve the update distribution problem. (Also, as we’ll see in the next chapter, this problem was to some extent solved in the 1800s, and we need to learn the

[9]http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar/ar01s04.html

solution again.)

So, Lean Publishing repurposes Eric Raymond's idea for books as follows:

Publish early. Publish Often. And listen to your readers.

By publishing early you get your ideas out there. If no one cares, you find this out as quickly as possible, sparing yourself as much wasted effort as possible. In the startup community, this idea is called "fail fast."

Furthermore, by publishing often, you build an authentic community, as your readers feel connected to the development of the book. This is the same effect as in Open Source software: Linux had an engaged community of developers since Linus released it early and often, whereas commercial software that is released as Open Source long after it is done often fails to gain momentum and build a community. For an example of the comparison, contrast the fate of OpenSolaris[10] to that of Linux[11].

Building a community of readers is essential, and not just for marketing purposes. Importantly, you can listen to them.

From the perspective of writing books, however, there's another related reason about why publishing early and often is essential: it helps you overcome the debilitating self-criticism often referred to as "writer's block."

On January 4, 2013, the editor of the New York Times magazine, Hugo Lindgren, published a fantastic essay entitled "Be Wrong as Fast as You Can"[12]. The essay shows a profound, penetrating understanding of how fragile ideas are in the face of self-criticism.

Go read it. Now.

The essay starts off like this:

> Here's a partial, redacted-for-the-sake-of-my-dignity list of stuff I once aspired to write but never did...

[10]http://en.wikipedia.org/wiki/OpenSolaris

[11]http://en.wikipedia.org/wiki/Linux

[12]http://www.nytimes.com/2013/01/06/magazine/be-wrong-as-fast-as-you-can.html?pagewanted=2&_r=1&pagewanted=print

Now, reading this line about preserving dignity, you know the writing adventure isn't going to end well. To quote F. Scott Fitzgerald, "I'm afraid the price for doing professional work is a good deal higher than you are prepared to pay at present."[13]

After listing a bunch of (frankly, pretty decent) ideas, the essay continues:

> If you had the time, believe me, I could flesh out these ideas for you, explain their origins, describe in fine detail my vision of the characters and plots and how it would all coalesce into something awesome.
>
> Or not. For at least 25 years, I've been serial day-dreaming like this, recording hundreds of ideas in a sequence of little notebooks that I have carried around and then stacked in a shoe box in my closet, a personal encyclopedia of undone to-do's.

Reading the essay, I can't help but feel that if Hugo Lindgren had followed the Lean Publishing approach and published some of these works in-progress, the world might have pulled the good ones out of him until he had completed works, the way that the "market *pulls* product out of the startup"[14] in Marc Andreessen's formulation of product-market fit.

In his essay, Hugo describes his career as follows as something that would just pay the bills and help him prepare himself for the "Masterwork of Spectacular Brilliance that would eventually define me." This is a very high bar for a piece of writing.

Hugo contrasts his creative process to that of John Lasseter (a Pixar founder), as described in an interview with Charlie Rose:

> Pixar's in-house theory is: Be wrong as fast as you can. Mistakes are an inevitable part of the creative process, so get right down to it and start making them. Even

[13]http://www.lettersofnote.com/2012/07/youve-got-to-sell-your-heart.html

[14]http://pmarca-archive.posterous.com/the-pmarca-guide-to-startups-part-4-the-only

> great ideas are wrecked on the road to fruition and then have to be painstakingly reconstructed.

Hugo describes how superior this is to his own approach:

> My confidence always collapsed under the weight of my withering self-criticism. I couldn't bear the awfulness and keep going. Even as I'm writing this essay, I have to stop myself from scrolling back to previous parts and banging my forehead against the keyboard as I see how short I've fallen of my expectations.

Most writers don't have this clear an understanding of how crippling self-criticism can be. But understanding the problem and successfully fighting it are different things. Forcing yourself to publish is the most effective way to fight this.

The best way of expressing this is the Steve Jobs slogan:

> "Real Artists Ship."

This is a very simple, cutting statement – a double-edged sword of a slogan:

1. You not a real artist unless you can ship your creations.
2. The act of shipping your creations is liberating enough that it empowers you to become a real artist.[15]

It is absolutely not a coincidence that John Lasseter is a founder of Pixar, and that Steve Jobs was the CEO of Pixar. The notion that Real Artists Ship is true whether you're making movies at Pixar or computers and iPhones at Apple.

So, how to overcome self-criticism and writer's block?

[15]Empower is one of those horrible business words now, but there is no better way to describe the result of actually getting your work in front of people: it fills you with creative power.

Ship. Publish early, publish often, and listen to your readers. Lean Publishing.

I understand this at a personal level. One of my issues, both in writing and in other areas, is that I'm a horrible perfectionist. For example, when I was in grade 12 I had problems finishing math tests: I'd do every question 2 or 3 times mentally before moving on. Heaven forbid I would make a "stupid mistake." I considered my score out of 10 (90% = 0, 95% = 5, 100% = 10), and I would honestly be devastated by a 94%. (This is absolutely not a humblebrag[16]: I was on an all-consuming quest to get a big enough scholarship to escape Saskatchewan, and I felt that my entire adult life was at stake with every calculus exam.)

So, coming from a perfectionist: You want to know a great thing about publishing in-progress?

The work cannot be perfect.

It isn't even done! Something unfinished cannot be perfect. *Q.E.D.* No perfectionism. Forcing yourself to publish unfinished, imperfect work is deeply therapeutic. It's the beginning of getting over yourself and trying to become a real artist.

Write. Publish. Repeat.

Of course, for this process to work, it has to be as easy as possible. That's why lightweight tools are so important – the writing is hard enough without the publishing aspect making things worse.

Publishers owe it to their authors – and thus to themselves – to make publishing this easy. For example, at Leanpub, the publishing process is one click of one button on one web page. (On Leanpub, this is true for both self-published authors and for authors working with a publisher.) The endorphin rush you get from clicking that button and seeing your book is enough to overcome a lot of writer's block.

[16]http://www.nytimes.com/2012/12/02/fashion/bah-humblebrag-the-unfortunate-rise-of-false-humility.html?pagewanted=all

...to get reader feedback...

Why is reader feedback important?

Well, how else will you know if what you are writing is understandable and enjoyable? If you are oversimplifying your readers will tell you. If your writing is incomprehensible, your readers will tell you. With *Flexible Rails*, I had readers submitting everything from broken links and grammar corrections to compilation errors – and even one security bug!

Publishers know that reader feedback is essential, so they pay development editors to provide good quality reader feedback. In essence, the development editor is a professional proxy, or surrogate, for readers.

You know who else is good at providing reader feedback?

Readers.

Authors have always known this: this is why great writers often enlisted their peers to review their partially finished or finished manuscripts. Now, while these types of interactions were essentially mutual favors between authors, today normal readers often want to read in-progress books.

Why?

For non-fiction books such as computer programming books, readers will benefit from earlier access to the information. If the information is new and in demand, it can help readers' careers immensely. *Common knowledge is cheap knowledge.* Once something is understood well enough that the relevant material is contained in print books on store shelves, understanding it is not worth as much as when it was new, confusing and obscure.

For other types of books, readers benefit by enjoying a more authentic connection to the author and to fellow readers. For fiction, the author can publish the work in serial. Writing in serial has merits and drawbacks, but the one thing it will help authors to do is to finish their books. We discuss serial fiction in more detail in the next chapter.

…pivot until you have the right book…

The concept of the pivot goes back to Eric Ries and Lean Startup theory. While the word "pivot" has now become a cliché, it does have meaning which should be examined for books.

> "I want to introduce the concept of the pivot, the idea that successful startups change directions but stay grounded in what they've learned. They keep one foot in the past and place one foot in a new possible future. Over time, this pivoting may lead them far afield from their original vision, but if you look carefully, you'll be able to detect common threads that link each iteration. By contrast, many unsuccessful startups simply jump outright from one vision to something completely different." –Eric Ries, "Pivot, don't jump to a new vision"[17]

Similarly for books, if you just start a new thing every time the writing bogs down you'll end up with a shoebox full of unfinished work. But if you incorporate reader feedback and evolve your book accordingly, you may actually finish your book, and produce a much better book.

But how do you know when to stop pivoting? How do you know when you have the right book? Well, if a book is a startup, let's look at the parallels for inspiration.

For a startup, the equivalent question is: How do you know if you have the right product? Marc Andreessen answers this with the concept of "product/market fit":

> The only thing that matters is getting to product/market fit.
>
> Product/market fit means being in a good market with a product that can satisfy that market.

[17]http://www.startuplessonslearned.com/2009/06/pivot-dont-jump-to-new-vision.html

> You can always feel when product/market fit isn't happening. The customers aren't quite getting value out of the product, word of mouth isn't spreading, usage isn't growing that fast, press reviews are kind of "blah", the sales cycle takes too long, and lots of deals never close.
>
> And you can always feel product/market fit when it's happening. The customers are buying the product just as fast as you can make it – or usage is growing just as fast as you can add more servers. Money from customers is piling up in your company checking account. You're hiring sales and customer support staff as fast as you can. …
>
> Lots of startups fail before product/market fit ever happens.
>
> My contention, in fact, is that they fail because they never get to product/market fit.
>
> …
>
> When you get right down to it, you can ignore almost everything else.
>
> –Marc Andreessen, "The Pmarca Guide to Startups, part 4: The only thing that matters"[18]

Getting to product/market fit is the main benefit of publishing in-progress. The advance sales are just a nice bonus. Once a book has product-market fit, it is a lot easier to build traction.

Building traction is the final component of the definition of Lean Publishing.

[18]http://pmarca-archive.posterous.com/the-pmarca-guide-to-startups-part-4-the-only

...and build traction once you do.

Successful startups know that traction is the most important thing. What is traction? Early and growing success in a market, a foothold, progress, momentum.

How do you get traction? Well, as discussed, it helps to have the right product, for the right market, at the right time. So, somewhat circularly, having product-market fit helps you build traction. Equally circularly, in terms of books, some books easier to build traction for than others. Again, this comes down to product-market fit.

Using the Lean Publishing approach helps you gauge reaction to your book, improve it, and judge whether you have product-market fit before pouring lots of effort into marketing the book.

If you don't have product-market fit, you can continue to pivot the book, or you can just kill it. Both are good outcomes, since you will minimize wasted effort.

If you do have product-market fit, then the Lean Publishing approach can also help you build traction. Since the book is published in-progress, authentic, word-of-mouth (or word-of-tweet) buzz can build before it is even done. Your readers have Twitter followers and Facebook friends. Some of them even have blogs. It doesn't take a "Social Media Expert" to tell you that's a good thing–provided the book is good.

If you are a publisher, this is also the point where you do all the normal things you do to help a book build traction. The difference, is that using the Lean Publishing process means you aren't starting at square one.

At this point, we have a pretty good understanding of the definition of Lean Publishing. In the next chapter, we consider its origins. Surprisingly, these date back to Victorian England and Charles Dickens.

Origins

In the first chapter of this book, we considered the definition of Lean Publishing at length. In this second chapter, we consider its origins.

The present reality of authors writing finished books which publishers then publish is neither natural nor inevitable. Despite how inescapable it seems today, the notion of selling a finished book is very much a recent, and almost reactionary, phenomenon. The natural way that stories are told is in serial. This has been true since humans first told stories around a campfire.

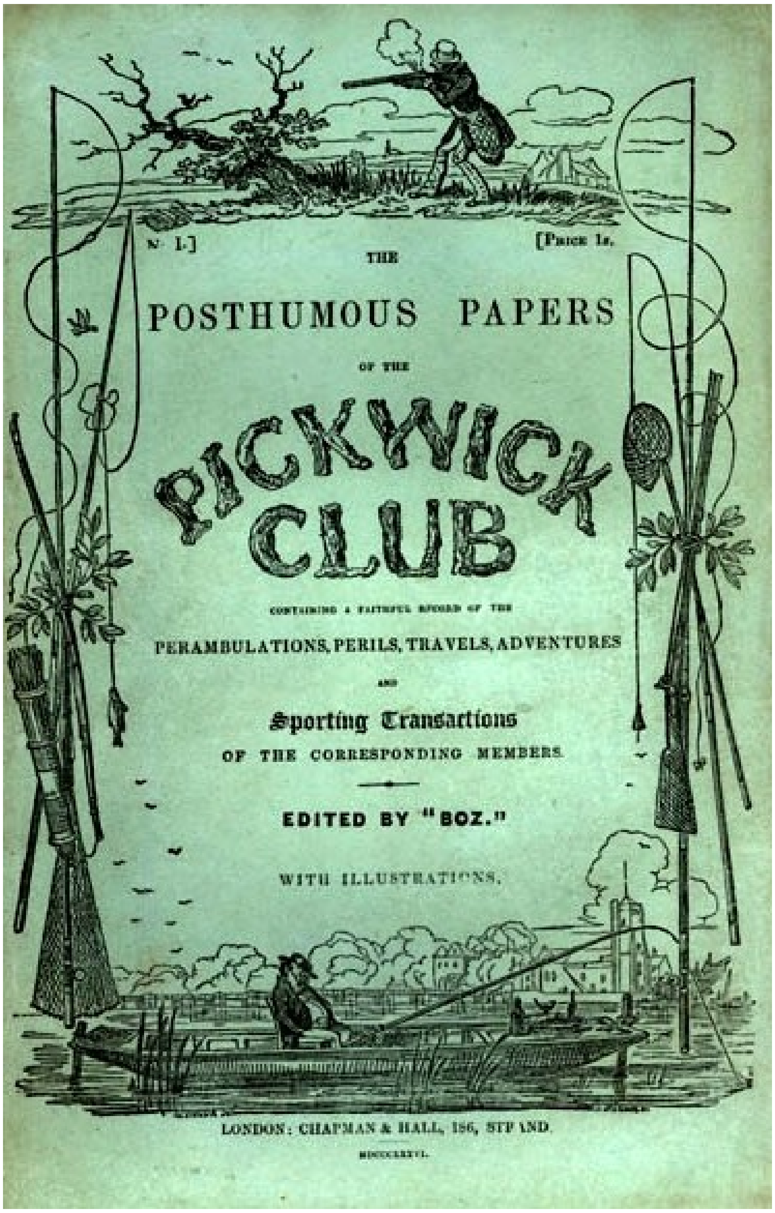
But in terms of Lean Publishing, the place to look first for origins is the 1800s.

Charles Dickens, *The Pickwick Papers* and the Rise of Serial Fiction

Serial fiction was first popularized in Victorian England by Charles Dickens.

Dickens wrote everything but his Christmas stories in serial. His firt work was *Sketches by "Boz"*, which he started writing in 1833 when he was only 21. Dickens published them in various newspapers and periodicals while he wrote them. Completed in 1836, these sketches were works of fiction and non-fiction describing London life.

The primary importance of *Sketches by "Boz"* was that it led to Dickens being involved in *The Pickwick Papers*. *The Pickwick Papers* was a monthly serial which was started when Dickens was 24. It was published in serial from March 1836 - October 1837.



Nº 1.] [Price 1s.

THE

POSTHUMOUS PAPERS

OF THE

PICKWICK CLUB

CONTAINING A FAITHFUL RECORD OF THE

PERAMBULATIONS, PERILS, TRAVELS, ADVENTURES

AND

Sporting Transactions

OF THE CORRESPONDING MEMBERS.

EDITED BY "BOZ."

WITH ILLUSTRATIONS.

LONDON: CHAPMAN & HALL, 186, STRAND.

MDCCCXXXVI.

The Pickwick Papers

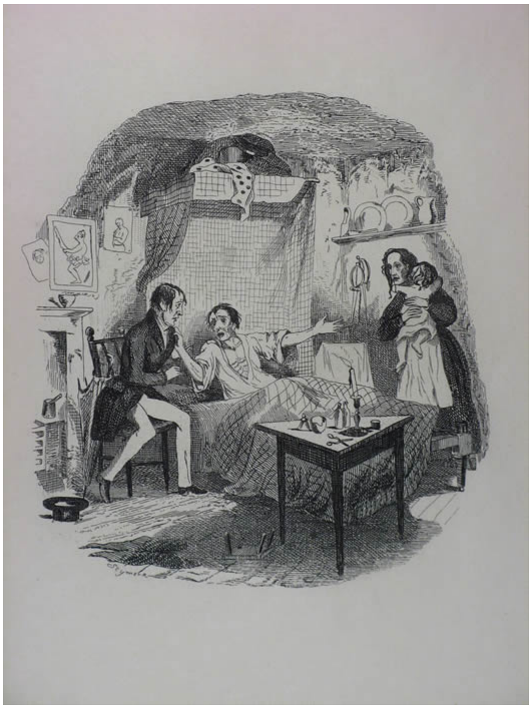
The Pickwick Papers began as an idea by famous illustrator Robert Seymour for a series of sketches about a club of cockney hunters and fishermen. Seymour proposed the idea to his publisher. All that was needed was a writer to fill in some text around the pictures.

Enter Dickens.

However, Dickens didn't know much about hunting or fishing, so he helpfully proposed that he write his own story and the drawings be adapted to match his story. (Talk about a pivot!) Thus, *The Pickwick Papers* began. (Well, actually, the title was: *The Posthumous Papers of the Pickwick Club, Containing a Faithful Record of the Perambulations, Perils, Travels, Adventures and Sporting Transactions of the Corresponding Members*. But we'll call it *The Pickwick Papers*, for obvious reasons.)

Needless to say, Robert Seymour was unhappy. Shortly after the project began, he had an argument over drinks with Charles Dickens. The next night, Robert Seymour completed this illustration[19] for The Pickwick Papers, entitled "The Dying Clown".

[19] http://en.wikipedia.org/wiki/File:The_Writings_of_Charles_Dickens_v1_p38_(engraving).jpg



The Dying Clown

He then killed himself with a shotgun.

The Pickwick Papers was only on chapter 2 at that point, and it was doing badly. However, like all great writers and anyone who has done anything worth remembering, Dickens persisted–and in chapter 10 invented the humourous cockney character of Sam Weller[20], who was to Mr. Pickwick what Sancho Panza was to Don Quixote.

[20]http://en.wikipedia.org/wiki/File:Sam-weller-kyd.jpeg

Sam Weller

The Pickwick Papers became a publishing phenomenon. Whereas the first monthly instalment had only sold about 400 copies, the last monthly instalment sold about 40,000 copies.

Dickens' combination of popularity and critical acclaim (well, from most critics) established serial fiction as the way that books should be first published in the 1800s. Books, split into volumes, were where successful serials were reprinted once completed. The serial was how you built traction for the book. The finished book was a predictable way of earning extra profit afterward.

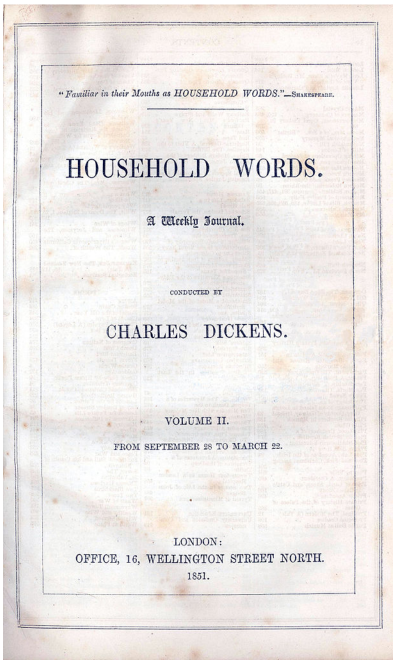
Dickens published all his novels in serial, except the Christmas books. You may have heard of some of them:

- *The Adventures of Oliver Twist*, *Bentley's Miscellany*, Monthly serial, February 1837 - April 1839
- *The Life and Adventures of Nicholas Nickleby*, Monthly Serial, February 1838 - April 1839
- *The Old Curiosity Shop*, Monthly Serial, *Master Humphrey's Clock*, February 1840 - April 1841
- *Barnaby Rudge: A Tale of the Riots of 'Eighty*, Weekly serial, February - November 1841

- *The Life and Adventures of Martin Chuzzlewit*, Monthly serial, January 1843 - July 1844
- *Dombey and Son*, Monthly serial, October 1846 - April 1848
- *David Copperfield*, Monthly serial, May 1849 - November 1850
- *Bleak House*, Monthly serial, March 1852 - September 1853
- *Hard Times: For These Times*, Weekly serial, April - August 1854
- *Little Dorrit*, Monthly serial, December 1855 - June 1857
- *A Tale of Two Cities*, Weekly serial, April - November 1859
- *Great Expectations*, Weekly serial, December 1860 - August 1861
- *Our Mutual Friend*, Monthly serial, May 1864 - November 1865
- *The Mystery of Edwin Drood*, Monthly serial, April - September 1870

One reason that Dickens preferred to publish in serial was that he could use cliffhangers to build buzz. For example, there were scenes such as crowds at docks shouting "Is Little Nell dead?" at passing ships from England during the serialization of *The Old Curiosity Shop*. The release of the last volume of *The Old Curiosity Shop* in 1841 was comparable to the excitement surrounding the 2007 release of the final Harry Potter instalment, *Harry Potter and the Deathly Hallows*. (Imagine what could have been, however, if the final Harry Potter instalment had been serialized in the New York Times Magazine, one chapter per issue.)

Dickens didn't just write serial fiction, he also published it. From March 1850 to May 1859, Dickens was the half-owner and full editor of *Household Words*, which proudly proclaimed itself to be "Conducted By Charles Dickens".



"*Familiar in their Mouths as HOUSEHOLD WORDS.*"—Shakespeare.

HOUSEHOLD WORDS.

A Weekly Journal.

CONDUCTED BY

CHARLES DICKENS.

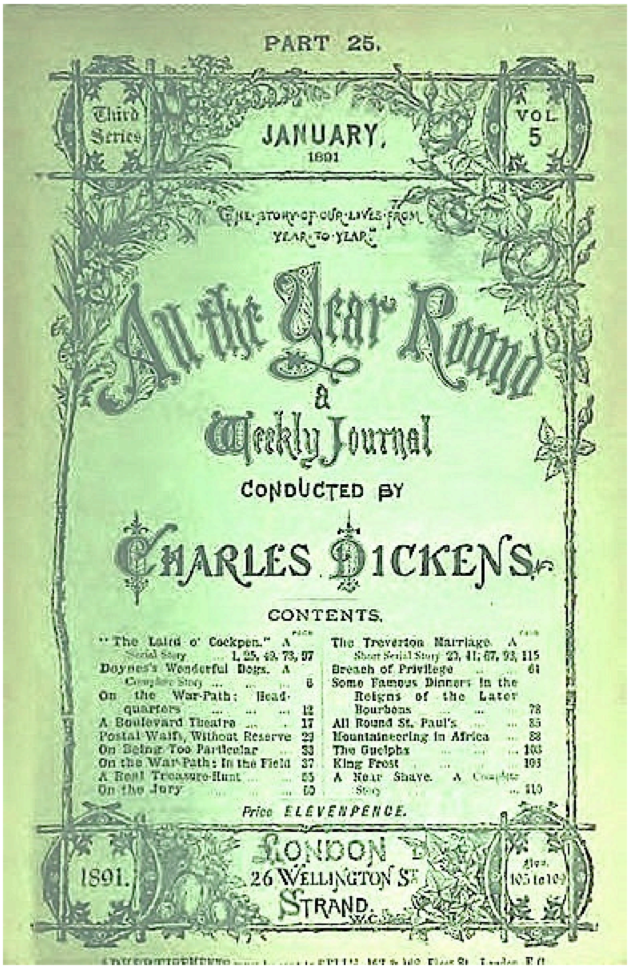
VOLUME II.

FROM SEPTEMBER 28 TO MARCH 22.

LONDON:
OFFICE, 16, WELLINGTON STREET NORTH.
1851.

Household Words

Household Words published serials including Dickens' own *Hard Times.* Dickens wanted complete control, however, and moved on to create his own serial publication, *All the Year Round*, which was founded, owned and edited by him.



PART 25.

Third Series

JANUARY, 1891

VOL. 5

"THE STORY OF OUR LIVES FROM YEAR TO YEAR."

All the Year Round

a Weekly Journal

CONDUCTED BY

CHARLES DICKENS.

CONTENTS.

	PAGE		PAGE
"The Laird o' Cockpen." A Serial Story	1, 25, 49, 73, 97	The Treverton Marriage. A Short Serial Story	23, 41; 67, 93, 115
Doyne's Wonderful Dogs. A Complete Story	6	Breach of Privilege	61
On the War-Path: Head-quarters	12	Some Famous Dinners in the Reigns of the Later Bourbons	78
A Boulevard Theatre	17	All Round St. Paul's	85
Postal Waifs, Without Reserve	29	Mountaineering in Africa	88
On Being Too Particular	33	The Guelphs	103
On the War-Path: In the Field	37	King Frost	105
A Real Treasure-Hunt	55	A Near Shave. A Complete Story	110
On the Jury	59		

Price ELEVENPENCE.

1891.

LONDON 26 WELLINGTON ST STRAND. W.C.

Also 105 to 109

All the Year Round

All the Year Round was published from 1859 to 1895. Among the works it published were Dickens' own *A Tale of Two Cities* in 1859 and *Great Expectations* from 1860 to 1861. *All the Year Round* also published seminal works by Wilkie Collins, as we'll see next.

Wilkie Collins, Mary Elizabeth Braddon and the Rise of Sensation Fiction in 1860s Victorian England

Wilkie Collins

Wilkie Collins

Two of Wilkie Collins' most important works were published in *All the Year Round*: *The Woman in White* from 1859 to 1860 and *The Moonstone* in 1868. *The Woman in White* was the first "Sensation Novel". In the 1860s in Victorian England, there was a craze for serialized, exciting, melodramatic fiction which was critically labeled as "sensation fiction". This was started by *The Woman in White.* Wilkie Collins also wrote and published *The Moonstone* in serial in *All The Year Round.* *The Moonstone* was the first–and according to T. S. Eliot, the best–English detective novel.

The notion of "sensation fiction" is extremely hard to understand today, since we live in a time in which fiction is split into literature and the type of fiction people buy at the airport–and in which very few books occupy both camps, and are mistrusted if they do. *Literary fiction is hipster fiction.* However, as T. S. Eliot wrote in his essay "Wilkie Collins and Dickens", before the 1860s there was no distinction:

> But there are many living who are not too young to remember the melodramatic stage before the cinema replaced it ... and who are not too old to have observed with curious interest the replacement of dramatic melodrama by cinematographic melodrama, and the dissociation of the elements of the old three-volume melodramatic novel into the various types of the modern 300-page novel. Those who have lived before such terms as "highbrow fiction," "thrillers" and "detective fiction" were invented realize that melodrama is perennial and that the craving for it is perennial and must be satisfied. If we cannot get this satisfaction out of what the publishers present as "literature," then we will read–with less and less pretence of concealment–what we call "thrillers." But in the golden age of melodramatic fiction there was no such distinction. The best novels *were* thrilling...

Later, Eliot discusses the distinction between melodrama and drama:

> But the frontier of drama and melodrama is vague; the difference is largely a matter of emphasis; perhaps no drama has ever been greatly and permanently successful without a large melodramatic element.

Eliot concludes his essay as follows:

> We cannot afford to forget that the first–and not one of the least difficult–requirements of either prose or verse is that it should be interesting.

Wilkie Collins' writing and life was certainly interesting. He was a close friend of Dickens, and through their friendship Dickens got better at writing plots, while Collins got better at writing characters. Collins also led a tumultuous personal life, living two separate lives with two partners (Caroline Graves and Martha Rudd). And if this wasn't enough, he also predicted the idea of Mutually Assured Destruction, in *1870*:

> "I begin to believe in only one civilising influence—the discovery one of these days of a destructive agent so terrible that War shall mean annihilation and men's fears will force them to keep the peace."

But before we veer too off-topic, the dangers of which have been chronicled in xkcd 214[21], we return to Wilkie Collins and his experiences with writing and publishing *The Moonstone*. Collins pubilshed it in serial, and his recounting of what happened shows how publishing in-progress enables authors to complete works they may otherwise have abandoned. While Collins was writing *The Moonstone*, he was afflicted by a severe case of gout. But having already started writing and publishing instalments of the work, he persisted, not wanting to disappoint his readers. In the "Preface to a New Edition" which followed the completion of *The Moonstone*, Collins described his experience as follows:

> While this work was still in course of periodical publication in England and in the United States, and when not more than one-third of it was completed, the

[21] http://xkcd.com/214/

bitterest affliction of my life and the severest illness from which I have ever suffered fell on me together. At the time when my mother lay dying in her little cottage in the country, I was struck prostrate, in London–crippled in every limb by the torture of rheumatic gout. Under the weight of this double calamity, I had my duty to the public still to bear in mind. My good readers in England and in America, whom I had never yet disappointed, were expecting their regular weekly instalments of the new story. I held to the story–for my own sake as well as for theirs. In the intervals of grief, in the occasional remissions of pain, I dictated from my bed that portion of The Moonstone which has since proved most successful in amusing the public–the "Narrative of Miss Clack." Of the physical sacrifice which the effort cost me I shall say nothing. I only look back now at the blessed relief which my occupation (forced as it was) brought to my mind. The Art which had been always the pride and the pleasure of my life became now more than ever 'its own exceeding great reward.' I doubt if I should have lived to write another book, if the responsibility of the weekly publication of this story had not forced me to rally my sinking energies of body and mind–to dry my useless tears, and to conquer my merciless pains.

The novel completed, I awaited its reception by the public with an eagerness of anxiety which I have never felt before or since for the fate of any other writings of mine. If *The Moonstone* had failed, my mortification would have been bitter indeed. As it was, the welcome accorded to the story in England, in America, and on the Continent of Europe was instantly and universally favourable. Never have I had better reason than this work has given me to feel gratefully to novel-readers of

> all nations. Everywhere my characters made friends, and my story roused interest. Everywhere the public favour looked over my faults–and repaid me a hundredfold for the hard toil which these pages cost me in the dark time of sickness and grief.
>
> I have only to add that the present edition has had the benefit of my careful revision. All that I can do towards making the book worthy of the reader's continued approval has now been done.
>
> W.C.
> *May*, 1871.

The benefits of writing and publishing in-progress books have never been explained better.

While Wilkie Collins wrote the first Sensation Novel, the star author of the genre (featured on the cover of this book) was Mary Elizabeth Braddon.

Mary Elizabeth Braddon

Mary Elizabeth Braddon became famous for writing *Lady Audley's Secret*, published in serial from 1861-1862. *Lady Audley's Secret* was the most popular Sensation Novel of Victorian England. It's a fun read, featuring themes of manipulation, murder, bigamy, arson and insanity[22]. (Who said Victorian England was boring?)

Lady Audley's Secret was very much in the style of *The Woman in White*, according to Braddon herself, years later:

> "I always say I owe 'Lady Audley's Secret' to the 'Woman in White'. Wilkie Collins is assuredly my literary father."

[22]I wrote that sentence in this way so that I don't ruin the book for you. Seriously, read it–it's more fun than most novels these days. You can find it in print or on Project Gutenberg.

On the other hand, this could just be the polite modesty of successful people–Henry James claimed that Braddon invented the sensation novel in *Lady Audley's Secret.*

Lady Audley's Secret was Mary Elizabeth Braddon's second published novel, and her first real hit. But it almost didn't happen. She was living with John Maxwell, a publisher with 5 kids and a wife in a lunatic asylum, to use the technical 1860s term. John Maxwell was launching a new magazine entitled *Robin Goodfellow.* Just before launch, the lead serial failed to be delivered. (If you're a publisher, I'm sure you've never had an author miss a deadline!)

Naturally, Mary Elizabeth Braddon volunteered to John Maxwell that she could write the lead serial.

His reply was as follows:

> "But even if you were strong enough to fill the position, there is no time."

Yes, that was a direct quote. Here's the rest of the conversation, as described by Mary Elizabeth Braddon:

> "How long could you give me?"
>
> "Until tomorrow morning."
>
> "At what time tomorrow morning?"
>
> "If the first instalment were on my breakfast table tomorrow morning, it would be in time."

The next morning John Maxwell found the opening chapters of Lady Audley's Secret on his breakfast table.

So, *Lady Audley's Secret* launched the *Robin Goodfellow* magazine in June 1861. *Robin Goodfellow* then folded in September 1861, after a few issues. (Back then, even the magazines themselves led melodramatic lives!) This could have killed *Lady Audley's Secret.* But it had fans hooked, including a famous actor who wrote and urged her to keep writing. The serialization resumed in *Sixpenny Magazine* (whose strapline was 'QUALITY, QUANTITY

AND CHEAPNESS'–who knew Victorian England was awesome?) and the novel was completed. As Braddon described it:

> "It was written from hand to mouth, as a serial, wherever I happened to be when the time of publication drew near..."

Lady Audley's Secret was published *extremely* early and often. It was a massive success, the *Fifty Shades of Grey* of the 1860s.

Mary Elizabeth Braddon was extremely prolific, completing **80** novels in her lifetime. In the 1860s, she wrote 2 or 3 novels *per year*, published in serial, of course. If that's not enough, from 1861 to 1870 she also managed to produce 6 children (with John Maxwell providing occasional assistance in the matter), edit *Belgravia* magazine (created by John Maxwell for her) and help raise the 5 children he already had. Some people are more productive than others, but Mary Elizabeth Braddon was on another level.

Serial Fiction Worldwide in the 1800s

In the 1800s, serial fiction wasn't just popular in Victorian England. Around the world, important and popular literature was originally published in serial form. Besides being a worldwide phenomenon, it was the defacto standard for quality: in 1878, a piece in Scribner's Monthly stated that only the "second and third rate novelist who could not get published in a magazine and is obliged to publish in a volume, and it is in a magazine that the best novelists always appear first."[23]

In Russia, Dostoyevsky's *The Brothers Karamazov* was published as a serial in *The Russian Messenger*. So was Tolstoy's *Anna Karenina*. Oh yeah, *War and Peace*–you guessed it, *The Russian Messenger*. (Well, sort of. An earlier version of parts of it were published as a serial in *The Russian Messenger* as *The Year 1805*. It's a big book.)

[23]http://en.wikipedia.org/wiki/Serialized_novel

In France, *Madame Bovary* was published in serial in *La Revue de Paris* (no, not *The Russian Messenger*).

In the United States, *Uncle Tom's Cabin* is a particularly interesting case. It was published in the US as a 40 week serial in *National Era*, starting in June 1851. Afterward, its author Harriet Beecher Stowe was approached by a publisher to make it a book. However, she didn't think people would read it in book form.

300,000 copies of Uncle Tom's Cabin were sold.

In the first year.

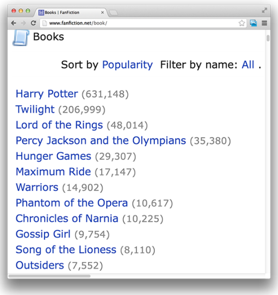
(So much for the idea that publishing in serial first is something that can harm a book launch!)

Within a few years, millions of copies of *Uncle Tom's Cabin* were sold and pirated, in the US and Britain. It was the bestselling novel of the 1800s, and the number two bestselling book in the US in the 1800s, second only to the Bible. It was also a prominent abolitionist event; one of many contributing factors leading to the US civil war.

Master of the Universe and *Fifty Shades of Grey*: Serial Fiction and Fan Fiction

The need for recurring fiction is, like melodrama, an essential human need. Book series such as *Harry Potter*, *Twilight* and *The Hunger Games* have all benefited, as have movie series such as *Star Wars*. However, while modern attention spans are typically held to be shorter than in days before Twitter and Facebook, the way that most people consume serial fiction is typically in novel-size chunks, unlike in the 1800s. However, this is just the result of historical accident, because the way that people consume print books is affected by how print books are produced.

When freed from the constraints of print books, instalments get naturally shorter. Nowhere is this more obvious than in one of the more interesting recent origins of Lean Publishing: fan fiction. Fan fiction involves authors, typically amateur authors, writing stories set in a universe created by another author, such as *Harry Potter* or *Twilight*. To say that fan fiction is popular for writers is a spectacular understatement: the following screenshot of FanFiction.net shows the most popular works, ranked by works of fan fiction:



Books

Sort by Popularity Filter by name: All .

Harry Potter (631,148)
Twilight (206,999)
Lord of the Rings (48,014)
Percy Jackson and the Olympians (35,380)
Hunger Games (29,307)
Maximum Ride (17,147)
Warriors (14,902)
Phantom of the Opera (10,617)
Chronicles of Narnia (10,225)
Gossip Girl (9,754)
Song of the Lioness (8,110)
Outsiders (7,552)

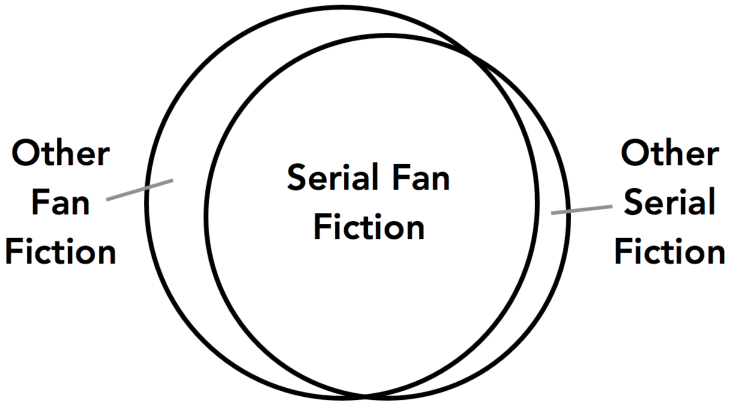
FanFiction.net Rankings

To be clear: this is not some metric like “page views of fan fiction”: this is the *number of unique works* of fan fiction set in that universe. Yes, there are *hundreds of thousands* of works of fan fiction about *Harry Potter* and *Twilight*. There are so many works of fan fiction that there are elaborate sub genres, just within Twilight fan fiction (or “Twific”).

Since the fan fiction genre is so massively popular, sexually focused and varied in quality, most people overlook one of the most interesting things about fan fiction:

It is almost entirely written and published in serial.

Most fan fiction is serial fiction, and there’s so much fan fiction written, that fan fiction is the dominant form of serial fiction.



Fan Fiction and Serial Fiction

Not only is fan fiction published in serial, it's typically published in-progress, using a Lean Publishing approach.

Recall the definition of Lean Publishing:

> Lean Publishing is the act of publishing an in-progress book using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

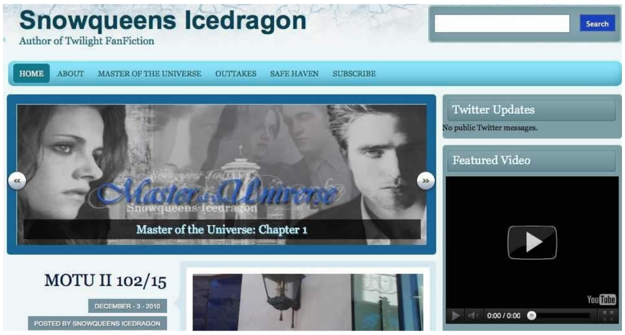
This pretty much defines fan fiction. Sites such as FanFiction.net and Wattpad are certainly lightweight tools focused on getting reader feedback!

To understand the impact of fan fiction and serial fiction, let's consider one high-profile example: *Fifty Shades of Grey*.

Fifty Shades of Grey started on FanFiction.net as Twilight Fan Fiction called *Master of the Universe* by "Snowqueens Icedragon", the pen name that author Erika Leonard chose.

Since FanFiction.net is really good for getting reader feedback and a community around your book, it's obviously the best place to first publish Twilight fan fiction. So, *Master of the Universe* was published on the fan fiction website FanFiction.net. It was massively popular, garnering over 37,000 reviews[24] and countless more readers: if you use the typical 90/10 ratio of reader to reviewer, you can assume over 300,000 readers.

After gaining a massive fan base, *Master of the Universe* was then removed from FanFiction.net website for Terms of Service issues regarding the sexual content of the book. So, Erika Leonard moved the work to 50Shades.com. The following screenshot[25] of 50Shades.com in December 2010 from a Galleycat article[26] is fascinating:



50Shades.com in December 2010

Here are some of the features of the website from the screenshot:

- an indication that “Snowqueens Icedragon” is an “Author of Twilight FanFiction”
- images of Edward and Bella from *Twilight* with the *Master of the Universe* title
- a spot for a featured YouTube video
- a spot for Twitter updates
- some sort of mailing list or RSS feed (the subscribe link)

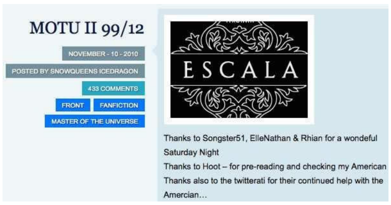
[24]http://fiftyshadesofpopculturetheory.blogspot.ca/2012/03/full-exchange-with-jason-boog-for.html

[25]http://www.mediabistro.com/galleycat/files/2012/03/2December2010.jpg

[26]http://www.mediabistro.com/galleycat/fifty-shades-of-grey-wayback-machine_b49124

- an indication of the most recent update

There was an active reader community, as you can see from the following screenshots of two different installments.



MOTU II 99/12

NOVEMBER - 10 - 2010

POSTED BY SNOWQUEENS ICEDRAGON

433 COMMENTS

FRONT FANFICTION

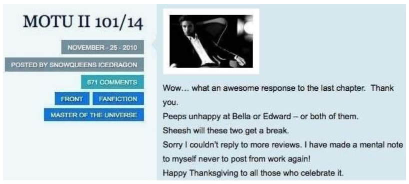
MASTER OF THE UNIVERSE

ESCALA

Thanks to Songster51, ElleNathan & Rhian for a wondeful Saturday Night
Thanks to Hoot – for pre-reading and checking my American
Thanks also to the twitterati for their continued help with the Amercian…

***Master of the Universe* evolving with reader feedback**

This instalment had 433 comments.



MOTU II 101/14

NOVEMBER - 25 - 2010

POSTED BY SNOWQUEENS ICEDRAGON

671 COMMENTS

FRONT FANFICTION

MASTER OF THE UNIVERSE

Wow… what an awesome response to the last chapter. Thank you.
Peeps unhappy at Bella or Edward – or both of them.
Sheesh will these two get a break.
Sorry I couldn't reply to more reviews. I have made a mental note to myself never to post from work again!
Happy Thanksgiving to all those who celebrate it.

More *Master of the Universe* reader buzz and feedback

This instalment, posted two weeks later, had 671 comments.

You don't get feedback and buzz like this with an astroturfed social media campaign once a work is done; you only get it if the work is built using a Lean Publishing approach, with real, authentic interaction between the author (whatever her pseudonym) and her readers.

After building this authentic community of readers and establishing product/market fit and traction, it was time to cash in.

Erika Leonard rewrote *Master of the Universe* as *Fifty Shades of Grey* by "E L James" and published it with The Writer's Coffee Shop, a small press based in Australia. Even more success followed, along with the obligatory bidding war by major publishers.

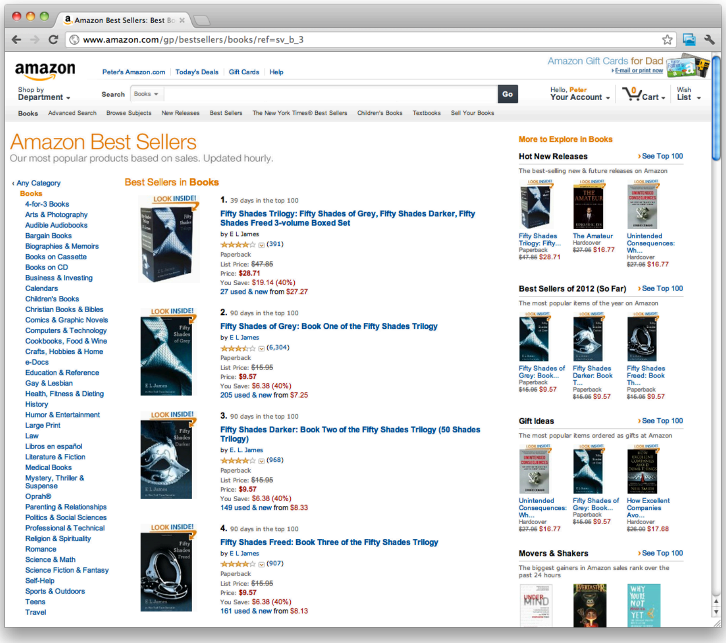
As an aside, by “rewrote” I mean “rewrote at least 11% of it.” The website Dear Author[27] did a fantastic textual comparison[28] of *Master of the Universe* and *Fifty Shades of Grey* to determine how similar they were. They also submitted the works to the student paper plagiarism detector Turnitin[29] and got the following similarity score: **89%**. This is interesting because since *Master of the Universe* is a derivative work of *Twilight*, and if being derivative is transitive, than *Twilight* author Stephenie Meyer is being very magnanimous here given the huge financial success of *Fifty Shades of Grey*. I didn’t start this book expecting to admire Stephenie Meyer, but I do.

Anyway, regardless of how much of *Fifty Shades of Grey* was essentially unchanged from *Master of the Universe*, the important fact is less subtle: *The Fifty Shades Trilogy occupied #1, #2, #3 and #4 on the Amazon.com bestseller list last year.*

[27]http://dearauthor.com

[28]http://dearauthor.com/features/industry-news/master-of-the-universe-versus-fifty-shades-by-e-l-james-comparison/

[29]http://turnitin.com/



Amazon Best Sellers
Our most popular products based on sales. Updated hourly.

Best Sellers in Books

1. 39 days in the top 100
Fifty Shades Trilogy: Fifty Shades of Grey, Fifty Shades Darker, Fifty Shades Freed 3-volume Boxed Set
by E L James
(391)
Paperback
List Price: ~~$47.85~~
Price: $28.71
You Save: $19.14 (40%)
27 used & new from $27.27

2. 90 days in the top 100
Fifty Shades of Grey: Book One of the Fifty Shades Trilogy
by E L James
(6,304)
Paperback
List Price: ~~$15.95~~
Price: $9.57
You Save: $6.38 (40%)
205 used & new from $7.25

3. 90 days in the top 100
Fifty Shades Darker: Book Two of the Fifty Shades Trilogy (50 Shades Trilogy)
by E. L. James
(968)
Paperback
List Price: ~~$15.95~~
Price: $9.57
You Save: $6.38 (40%)
149 used & new from $8.33

4. 90 days in the top 100
Fifty Shades Freed: Book Three of the Fifty Shades Trilogy
by E L James
(907)
Paperback
List Price: ~~$15.95~~
Price: $9.57
You Save: $6.38 (40%)
161 used & new from $8.13

More to Explore in Books

Hot New Releases › See Top 100
The best-selling new & future releases on Amazon
Fifty Shades Trilogy: Fifty... Paperback ~~$47.85~~ $28.71
The Amateur Hardcover ~~$27.95~~ $16.77
Unintended Consequences: Wh... Hardcover ~~$27.95~~ $16.77

Best Sellers of 2012 (So Far) › See Top 100
The most popular items of the year on Amazon
Fifty Shades of Grey: Book... Paperback ~~$15.95~~ $9.57
Fifty Shades Darker: Book T... Paperback ~~$15.95~~ $9.57
Fifty Shades Freed: Book Th... Paperback ~~$15.95~~ $9.57

Gift Ideas › See Top 100
The most popular items ordered as gifts at Amazon
Unintended Consequences: Wh... Hardcover ~~$27.95~~ $16.77
Fifty Shades of Grey: Book... Paperback ~~$15.95~~ $9.57
How Excellent Companies Avo... Hardcover ~~$26.00~~ $17.68

Movers & Shakers › See Top 100
The biggest gainers in Amazon sales rank over the past 24 hours

50 Shades of Amazon.com

From a Lean Publishing perspective, however, the most interesting thing is that not only did *Fifty Shades of Grey* begin life as fan fiction, but, like most of fan fiction, that *it was published in-progress and in serial.* The success of *Fifty Shades of Grey* shows what product/market fit and massive advance buzz from serial fiction can do for you. To be clear:

The top 3 bestselling books published in 2012 were first published as serial fiction.

Serial fiction is not something that is coming soon. It's here now. And if you expand the definition to include book series such as *Harry Potter* and *Twilight* where the size of the instalments is a novel, you'll realize that it has been dominant for a while. All that is happening now is that the instalments are getting shorter, and reaching their natural size of one sitting.

Erika Leonard is the modern equivalent of Mary Elizabeth Braddon. Both are/were bestelling authors, with millions of readers each. Both women first published their novels in serial and in-progress. Both wrote their novels very quickly, both were hated by critics and both profited handsomely from their efforts.

Let's look back at the definition of Lean Publishing again to see how close to it *Fifty Shades of Grey* stayed:

> Lean Publishing is the act of publishing an in-progress book using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

In-progress? Check. *Ebook?* Check. *Lightweight tools?* FanFiction.net is as lightweight a tool as it gets. Check. *Many iterations?* The screenshots above imply over a hundred iterations, just for part 2. Check. *Reader feedback?* Check. *Pivot?* Well, she pivoted as she wrote *Master of the Universe*, and then pivoted again, changing at least 11% of the book to create *Fifty Shades of Grey*. Check. *Build traction?* You saw the screenshot of Amazon.com, right? Check, please!

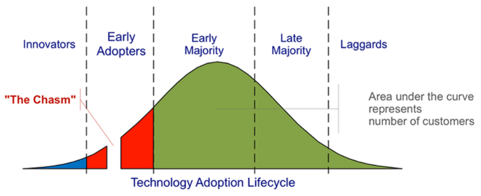
In considering the origins of Lean Publishing, next we turn our attention from the serious world of *Fifty Shades of Grey* to the salacious world of computer programming books.

The Technology Adoption Lifecycle

The Technology Adoption Lifecycle[30] is a very famous bell curve describing how new technology spreads. For disruptive innovation, such as a revolutionary new technology product, author Geoffrey Moore has argued[31] in *Crossing the Chasm* that there is a gap

[30]http://en.wikipedia.org/wiki/Technology_adoption_lifecycle
[31]http://en.wikipedia.org/wiki/Crossing_the_Chasm

between the early adopters and the early majority. This curve[32] looks like this:



The Technology Adoption Lifecycle with Geoffrey Moore's Chasm

The Technology Adoption Lifecycle has profound consequences for publishing. Currently, ebook readers such as iPad and Kindle have recently crossed the chasm from early adopters into the early majority. PDF has already reached the late majority in terms of distribution of programs that can read and write the format, but the suitability of PDFs for books still has an image problem for everything but technical books.

For the past five years, even before the Kindle and iPad were created, PDF sales have been steadily growing for technical books. Why technical books and not, say, novels?

The key is something I call the *Technology Adoption and Information Distribution Lifecycle*. The name is a mouthful, but what the concept refers to is the ways that information about a given technology can be distributed as that technology moves through the Technology Adoption Lifecycle.

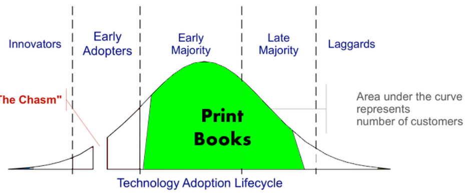
One of the main drivers of this is that the speed of technological change is increasing, so the length of time that the x axis of the Technology Adoption Lifecycle exists for a given technology is being compressed. Whereas in decades past, a book about a given technology could be relevant for 5 (or 10!) years, now you're lucky if that's true for 2 years. Since it takes time for a book to get written, this often means that by the time the book is written, copy edited, typeset, printed, put on trucks and shipped to stores, parts of it are

[32]http://en.wikipedia.org/wiki/File:Technology-Adoption-Lifecycle.png

obsolete. It is possible for entire books to become obsolete before they are even finished!

One consequence of this is that there is a pressure to start early, building and shipping print books based on beta versions of software, and hoping that the information will still be correct when the final version is released. Sometimes this works, other times it does not. As I discussed in the introduction to this book, my book *Hello! Flex 4* had the latter outcome. Even in circumstances that aren't as extreme as that, different methods of information distribution apply to different parts of the Technology Adoption Lifecycle. We'll consider how this applies to various methods of information distribution, including blogs, in-progress ebooks, completed ebooks and print books.

First, we'll see what this looks like for print books. Since print books take a while to get written, edited, copy-edited, printed, put on trucks and shipped, the curve looks like this:



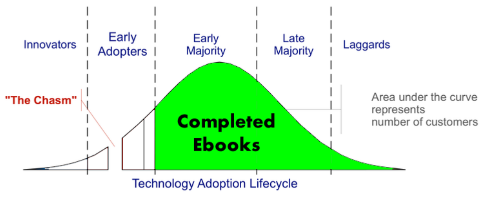
Technology Adoption and Information Distribution Lifecycle: Print Books

There are two very interesting consequences here.

First, due to the time required to produce a print book, print books completely miss the innovators and early adopters! These people represent a minority of the total addressable market, but by their nature as enthusiasts they are the most passionate customers. They are the evangelists of a technology, and the exact people that you want to promote your book. Choosing print means you will miss them *completely*. By the time your book is on a shelf, these customers will already know what you are saying, and won't be in the market for your book.

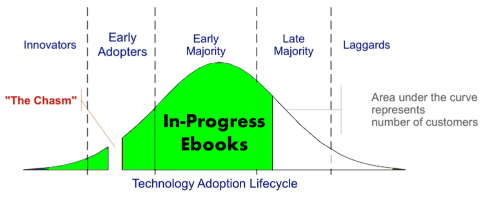
Second, since shelf space is worth money, print books currently don't go all the way to the end of the curve. Over time this will be changed by print-on-demand and Amazon for distribution, but today print books still do go out of print.

Next, completed ebooks. This curve resembles the curve for print books, since it takes time to write, edit, copy-edit and typeset an ebook. The curve starts a bit sooner, however, since the bits in an ebook don't need to be shipped in trucks and sit on shelves, the way that print books do.



Technology Adoption and Information Distribution Lifecycle: Completed Ebooks

Three notable publishers, The Pragmatic Programmers, Manning and O'Reilly have realized this. So, they have long offered in-progress ebook versions of their books, in order to publish at least some of the content in their books in some format before it becomes obsolete. The Pragmatic Programmers call their program "Beta Books"; O'Reilly calls theirs "Rough Cuts"; Manning calls theirs "MEAPs", for "Manning Early Access Program." In all cases the point is the same: publishing in-progress ebooks.

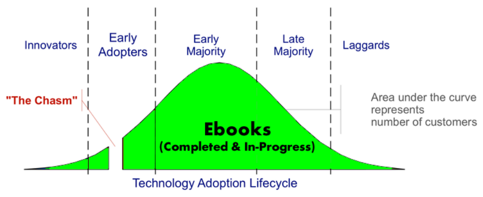


Technology Adoption and Information Distribution Lifecycle: In-Progress Ebooks

It is no surprise that all three of these publishers sell computer programming books: computer technology changes make books obsolete quickly. So, it was inevitable that three of the best computer book publishers would publish in-progress books: the market demanded it.

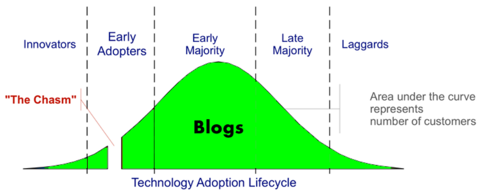
Note that as a technology matures and starts to wane, no new books are started about it. This is why the In-Progress Ebooks curve stops before the technology itself dies: for essentially every technology, there is a point when no new books about it are being started.

Taken together, when you combine in-progress ebooks and completed ebooks, they cover the whole Technology Adoption Lifecycle curve:



Technology Adoption and Information Distribution Lifecycle: All Ebooks

Finally, blogs. People start blogging about something new and shiny right away. So, blogs reach the innovators (or are written *by* the innovators) and the early adopters. Once a technology reaches the mainstream, however, it's less blog-worthy. So blogs also cover the entire area under the curve, even the laggards who are served by obsolete blog archives. Dead technology gets dead blogs.


Innovators
Early Adopters
Early Majority
Late Majority
Laggards
"The Chasm"
Blogs
Area under the curve represents number of customers
Technology Adoption Lifecycle

Technology Adoption and Information Distribution Lifecycle: Blogs

The big point here is that not only do blogs and ebooks address the whole Technology Adoption Lifecycle, but that only blogs and in-progress ebooks address the passionate innovators and early adopters who will be effective evangelists for a given technology or book.

Realtime Publishing

Before moving on, I want to take a closer look at the in-progress book programs. Publishing in-progress books is one thing; using a lightweight, semi- or fully-automated toolchain is another. Lean Publishing does not refer to the process of making in-progress books by typesetting Word documents: if the process adds enough friction, you can't iterate fast enough. In this regard, publishers like the Pragmatic Programmers and O'Reilly have constructed their own toolchains for producing books. These toolchains have historically been a bit heavy or obnoxious–any toolchain involving the author writing in DocBook, which is XML, counts as the latter–but they are improving. O'Reilly is probably the most forward-thinking of these publishers. Not only have they added AsciiDoc as an alternative to DocBook, but in February 2011 they announced a program called Realtime Publishing whose values align closely with the Lean Publishing values.

The page[33] which describes O'Reilly's Realtime Publishing program makes an excellent case for Lean Publishing, so I'm going to quote it at length[34] here.

> Realtime Publishing
>
> If you've been following the publishing industry at all, you've no doubt heard the screams of pain. When the tech bubble burst in 2000, book sales went into free fall. And although the rest of the technical industry has recovered and thrived in the past decade, at best publishing has performed only slightly worse than the previous year. The bloodbath has ended, but the wounds are still oozing.
>
> It's easy to point to reasons, but most of those reasons are at best half-right. It's easy to look at the shrinking shelf space allocated to computing in the retail chains—but that's surely an effect, rather than a cause. If readers were buying the books, the bookstores would be selling them.

I actually hadn't noticed this. But that's kind of the point: I didn't go into bookstores to buy print books any more! But it's true: in 2012, when I visited Bolen Books in Victoria to hear Cory Doctorow speak, I couldn't even find their computer book section anymore! This section used to be primarily why I went to Bolen, and where I had spent hundreds of dollars as a poor university student just over a decade earlier.

[33] http://oreillynet.com/oreilly/authors/welcome/realtimepublishing.csp

[34] I want to emphasize how much respect I have for O'Reilly. I did a double major in computer science and psychology in university, and O'Reilly books were the most helpful books for me as I learned how to program. I had published the first version of this book as a manifesto in 2010, and Leanpub is based on the Lean Publishing ideas, but when I read O'Reilly's announcement of Realtime Publishing in 2011 I felt like the Lean Publishing ideas had arrived: this was O'Reilly adopting essentially as much of the Lean Publishing model as any traditional publisher had to date. The fact that it was O'Reilly showed that one of the smartest publishers in the room thought in a very similar way to how I did.

> …
>
> Here are the issues that really face publishing, and what O'Reilly is doing to address them—for our good and for our authors'. First, the pace of technical change: if anything it's gotten faster over the past decade. A 400-page book takes roughly a year to write, particularly for the kind of author O'Reilly usually works with: technical experts who have other jobs. A print book that takes a year to write, and 3 months in production, is bound to be six months out of date by the time it hits the shelves. That's really not a service for anyone.

Hearing O'Reilly state this, just as I had in "The Technology Adoption Lifecycle and Computer Programming Books"[35], was shocking in its candour.

> Second, in an open source world, technologies tend to fragment. Developers have a host of alternatives—but no alternatives have enough users to make a book profitable. A few years ago, Ian Darwin counted over 80 Java web frameworks. If this was just a peculiarity of the Java world, we could perhaps laugh. But there are many Python frameworks; many Javascript frameworks (I've been told "hundreds", but I haven't tried to count); over two dozen NoSQL databases; many new programming languages. Books like Head First Java that cover very general topics are still profitable, but it's hard to justify publishing on Wicket. Or Sinatra. Or Apache Jackrabbit. Publishers either have to pick winners early on, or they have to sign books on everything and benignly neglect the books

[35]I had originally written and published the "The Technology Adoption Lifecycle and Computer Programming Books" as part of the first version of the Lean Publishing Manifesto in 2010.

> on technologies that don't thrive. Neither strategy is a recipe for publishing success.

The interesting point here is that benign neglect is essentially what we do at Leanpub: we provide the toolchain, and it's up to the authors to do the writing and marketing. A book which is not a publishing success by traditional publisher's standards (> 5000 copies) can still be a publishing success by the author's standards, especially at Leanpub's royalty rates of 90% - 50 cents: a $20 book earns the author $17.50, so if that book sells 4000 copies this is $70,000 to the author. Many computer programming book authors would be happy with this type of failure, especially since their books earn them consulting leads and let them raise their consulting rates.

> That's the real problem: not Google, not retail, but the rate of technical change and the tendency of technologies—particularly open source technologies—to fragment into hundreds of competing implementations. How do we address these issues, so that we and our authors can prosper in the coming years?

I 100% agree. I couldn't have said it better myself!

> O'Reilly is introducing a new model for publishing. We're calling it "realtime publishing." It's enabled by ebooks, which enable us to answer a question we've been asking for a few years: why can't we publish books the way developers release software?

Yes! Yes! Yes!

> First, the demands of a physical book require a certain heft. A print book that's too small gets lost on bookshelves. With ebooks, size ceases to be an issue. And this has important ramifications. It's possible to write on smaller, more specific topics: instead of 1200 pages

> on Javascript, 70 pages on working with Canvas, plus 70 pages on the class system, etc. All of these can be separate publications. By replacing 1200 pages of print with a collection of shorter ebooks, we can drastically reduce the time it takes to bring content to market. Instead of a year or years, we're talking about a month, a couple of months at the outside.

This seems to be creating a false dichotomy between "really short < 100 pages books which are suitable for Realtime Publishing" and "traditional length books that are not". O'Reilly should take the next leap and open this up to books of all lengths and topics.

> Furthermore, not only are print books out of date as soon as they hit the market, it's a year or more before they can be updated. There's retail inventory that has to be sold, or it will be returned and destroyed. It's not good to tell a retail bookstore manager that the stock he bought two months ago, or even 9 or 12 months ago, is out of date. Ebooks don't have this problem: they can be updated real-time. Whenever the author fixes a bug, describes a new feature, updates the book for a new release, all she has to do is request a new release.

Yes!

> Our production staff pushes the build out to the servers and notifies customers so they can download a new copy. O'Reilly is already doing that with our ebook products. Eventually, books will update themselves, the same way mobile apps do: you'll open your ebook reader to see a message saying "update 3 books now?" I don't know when that feature will appear in Kindle, iBooks, and Google Reader, but I believe it will be sooner rather than later. The difference between "early

> access” and “first edition” is disappearing: ebooks can be, and will be, release-early, release-often products.

Yes!

> Realtime publishing also addresses the fragmentation problem. Many factors contribute to the cost of the book: printing is less than most people think, editorial and production much more. There’s also inventory cost, and the risk associated with producing books that don’t sell. Ebooks aren’t necessarily less expensive to publish, but they are certainly less risky. There’s no inventory, and because they can be shorter (books under 200 or so pages “disappear” on bookshelves), they don’t require as big an investment to develop. If producing a 500-page print book costs $40,000, producing a 50-page ebook should cost $4000. That difference directly affects our ability to take risks on new technologies that don’t have large, well-defined audiences. The net result is that we can take publish on many more technologies. We can afford to build a series of short ebooks around technologies that might not otherwise make the cut. That also means that we can start ebooks earlier, without waiting to see whether any given project is a “winner.”

Producing a 50-page ebook costs the author nothing but time when using the Lean Publishing approach on Leanpub. So, the costs O’Reilly describe must essentially be all labor costs. So, this brings up the question of risk: At what point should a publisher like O’Reilly start an ebook project using this approach?

Well, if the project began with almost no expense, it could be “as soon as there was a willing author”. This would imply that as much of the work that makes up the $4000 of cost get shifted to the end of the process, so that it can only be spent on books that look like they will earn it back. But then if this was done, it would imply that some authors will be essentially kicked out: they'd go into a project expecting a fancy production process at the end an not get it. So to use this kind of approach requires a phased approach where something starts out essentially self-published and then the publisher has an option to pick it up and polish it if it shows traction.

There's a really simple way to put this, in kind of a Zen koan style:

> *Q. How does a publisher succeed in 2013?*
> *A. Don't publish failed books.*

The full publishing process could act as a force-multiplier on successful books that are first published by their authors with either little or no help from their publisher.

> Why do we believe in this strategy? There's no shortage of pundits who will tell you that 2011 is the year of the ebook. But we don't have to look at pundits. We can look at our own sales: in the past year, O'Reilly has come to expect at least 35% of the dollar sales on new titles to come from ebooks (including Safari), and we expect that percentage to grow. That growth is driven by the need to get information now: you can click and have the book on your device in seconds, rather than waiting for Amazon to ship it, or for you to take a trip to the local bookstore. And ebooks are available as soon as the book finishes our production process—several weeks before they've been printed and percolated out from the printer to the warehouse to the bookstore. So the need for information, for ideas, is already driving the increase in ebook sales, even before we've implemented realtime publishing.
>
> …

> Authors need to work in DocBook or AsciiDoc. These formats integrate smoothly into our toolchain, and let us pump out new releases without extended production processes.
>
> …

So close! AsciiDoc isn't as good as Markdown in my opinion, but it's a hell of a lot better than DocBook.

> …
>
> We're increasing author royalties to 25% for all ebook products in this real-time program. That holds both for books that are initially planned as ebooks, and ebooks that are extracts from other books. For customers who want print, books will be available through print-on-demand; a 10% royalty applies to POD sales. There will be no advances.

This is less than 1/3 of what Leanpub pays for a $10 or $20 book–our 90% - 50 cents royalty rate means a $10 Leanpub book earns $8.50 (85%) and a $20 Leanpub book earns $17.50 (87.5%)–but it's more than double the 10% that most publishers pay. So it's a huge step in the right direction.

> We expect authors to stick with their ebooks post-publication, as they would with any other project: we want the books kept up-to-date. Releasing an update will be as simple (for you and for us) as committing changes to an O'Reilly SVN archive. We realize that authors don't want to be wedded to their books for the rest of their careers, but we would like authors to be responsible about finding someone to carry on the work when they move on.

This is, in my opinion, completely unreasonable and totally counterproductive. Books have to have a "done" point, otherwise authors will go insane. And authors shouldn't feel the need to "appoint a maintainer" the way open source project leaders do. Only a very, very small minority of books are constantly-updated "living" books. In most cases, a book gets to a done point and then stops. Then, after the book becomes obsolete, the author considers writing a new edition about the new version of whatever he or she is writing about.

If the author had to appoint a maintainer, then when the book became obsolete, he or she would be competing against the maintainer: the author with a new edition, and the maintainer who was keeping the original book current for longer than expected.

> …
>
> This is just the beginning of the ebook story. We will see many innovations in the coming year—some from us, and certainly some from other publishers. These are exciting times: we're re-inventing publishing in terms that are meaningful for the computing industry.
>
> …

Yes! At Leanpub we're happy to be part of this process.

From Blogs to Books

Lean Publishing also draws heavily on lessons from blogs. We've discussed blogs in the section on the Technology Adoption Lifecycle, and we've discussed serial fiction at length. In a sense, blogs are just serial non-fiction. Now, blogs typically don't make much money for their authors. However, if the blog leads to lucrative books, then this low monetization is just deferred monetization.

It's important to recognize that the writing (I hate the word "content") in a good blog is often easily turned into a good book. Examples of books based on repurposed blog posts include 37signals' *Getting Real* and *Rework*, Paul Graham's *Hackers and Painters*, Eric Ries' *Startup Lessons Learned* and *The Lean Startup* and Joel Spolsky's many books. This is a fantastic thing. These successful blogs have built authentic communities around the author, and the community is happy to support the books that result from the blog. This often leads to very good and very successful books.

Let's look more closely at two examples now.

Getting Real and *Rework*

37signals is a Chicago-based software company that, in their words makes "frustration-free web-based apps for collaboration, sharing information and making decisions." Their most famous products are Basecamp (a project management tool) and Highrise (a contact manager). Besides these innovative products, they are also known for their extremely popular *Signal vs. Noise* blog and for employing David Heinemeier Hansson, creator of the Ruby on Rails web application framework. (Ruby on Rails was extracted from Basecamp.)

Besides these accomplishments, Jason Fried and David Heinemeier Hansson of 37signals are also bestselling authors. They wrote a book called *Getting Real* about the software development process they use to create web applications. The most innovative thing about *Getting Real*, however, was the following: it was essentially an edited version of what 37signals had been saying on *Signal vs. Noise* for years. Yet, with a little curation and editing, they managed to sell over 20,000 PDFs directly from their website–which, at $19 each, represented over $380,000 in profit–in the process. And good for them: *Getting Real* was worth the $19, even to people like myself who were already regular readers of *Signal vs. Noise*. It was especially worth it, however, to people who had not followed the blog and wanted a heavily curated and edited version of it, to catch up to 37signals' way of thinking.

In 37signals' terms, what they were doing was to "sell your waste products." They framed their innovative ideas as the waste products of creating innovative software, like sawdust is a waste product at a lumber mill.

Now, what's better than making one bestselling book out of your blog? Making two, of course! The second book was *Rework*, a New York Times bestseller which sold over 300,000 copies. Essentially, *Rework* takes the *Getting Real* ideas about how to develop web applications and applies them to business in general.

What's better than getting two bestselling books out of your blog? Three, of course! They're working on a new book, called *Remote*, about telecommuting. (This is why there have been posts about telecommuting on *Signal vs. Noise* recently, of course. Where better to get product/market fit, beta test ideas and build advance buzz than on an extremely popular blog?) With the recent debate about telecommuting at Yahoo!, their timing is pretty good. So chances are they'll have a third bestselling book in a year from now.

Startup Lessons Learned and *The Lean Startup*

Next, we consider Eric Ries. He created the notion of The Lean Startup, based on his experiences and the Customer Development teaching of Steve Blank.

But the process that Eric has followed in developing his Lean Startup ideas in writing is interesting, as it is, unsurprisingly, very much a Lean Publishing process.

First, Eric developed his voice by blogging. One of his earliest posts, from September 18, 2008 was entitled "Lo, my 5 subscribers, who are you?"[36]. In this post, which is essential reading, he explains the benefit of being small, in that you can get to know your early customers better than you can when you have thousands of them. So, he conducted an email survey, and got feedback from these early readers, who he considered his early customers or "earlyvangelists".

In that post, written when he only had 5 subscribers, Eric wrote:

> And since I have a blog, I have a way to ask questions directly to you. If you have a minute, post your

[36]http://www.startuplessonslearned.com/2008/09/lo-my-5-subscribers-who-are-you.html

> answers in a comment, or email me. Here's what I want to know:
>
> 1. First of all, the NPS question[37]: On a scale of 1-10 (where 10 is most likely), how likely is it that you you would recommend this blog to a friend or colleague?
> 2. How did you hear about it?
> 3. What led you to become a subscriber, versus just reading an article and leaving like everybody else? (or, if you're not a subscriber, what would it take to convince you?)
> 4. What do you hope to see here in the future?
>
> Thanks, you loyal few. I am grateful for your time and feedback.

He used the lessons he learned well, as shown in the subscriber count on his two subsequent surveys: on January 24, 2010, his survey was entitled "Lo, my 18891 subscribers, who are you?"[38]; on September 6, 2010, his survey was entitled "Lo, my 57692 subscribers, who are you?"[39]. Yes, those post titles read like a victory lap, but like most victory laps, they are well deserved.

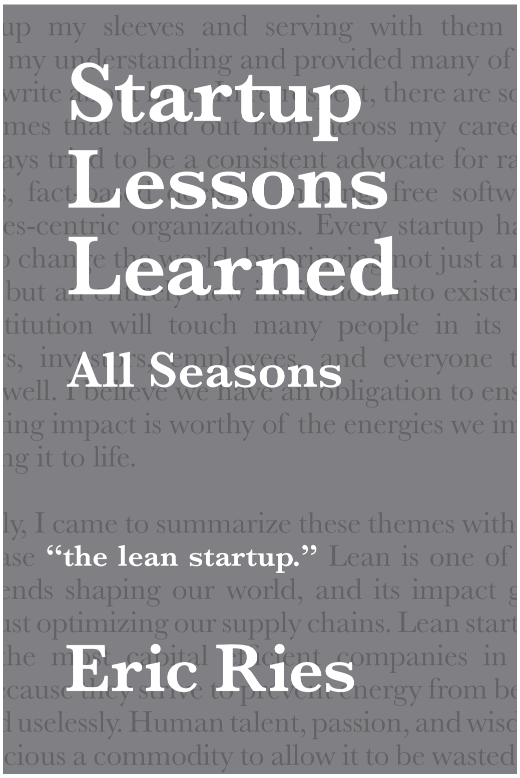
The natural evolution of the ideas in Eric's blog was, of course, a book. Eric's blog became *Startup Lessons Learned*[40], which was the very first Leanpub book.

[37]http://en.wikipedia.org/wiki/Net_Promoter_Score

[38]http://www.startuplessonslearned.com/2010/01/lo-my-18891-subscribers-who-are-you.html

[39]http://www.startuplessonslearned.com/2010/09/lo-my-57692-subscribers-who-are-you.html

[40]http://leanpub.com/startuplessonslearned



Startup Lessons Learned

Now, unlike *Getting Real*, this was not a curated and edited book. Instead, it was a verbatim export of Eric's blog, automatically generated and organized by month chapters. It's currently about a thousand pages, and it's actually a decent read, even though it's just a verbatim export.

However, if Eric had put the level of effort into it that Jason Fried and David Heinemeier Hansson had, he would have been many orders of magnitude more successful than he was. Now, this experience helped us launch Leanpub, so I am eternally grateful to Eric. But to this day I'm convinced that Eric cost himself hundreds of thousands of dollars in direct revenue by not just doing the extra bit of curation needed to make *Startup Lessons Learned* more of a real thing.

Eric did do this work, however. That work produced his second book, *The Lean Startup*[41], which I consider to really be his first real book. *The Lean Startup* was created with a traditional publisher, not that it actually matters that much–it would have been massively successful regardless of who published it. It became a New York Times bestseller, much like *Rework* did for Jason Fried and David Heinemeier Hansson after their *Getting Real* blog book.

Now, obviously, not every print book that is written after a blog and a blog book will become a New York Times bestselling print book. But the Lean Publishing process did help, as it ensured that the ideas would have product-market fit. For Eric, this process was his blogging; *Startup Lessons Learned*, the book version, was just an interesting experiment along the way.

The main point is that Eric Ries developed his ideas by blogging, and then by publishing a book version of his blog *before* producing a traditionally-published book with a publisher. Eric's bestselling book had already benefitted two years of feedback from an engaged community of blog readers. He had created a community around his book that was so engaged that it even had *its own conference*, the Startup Lessons Learned conference, *before the book was written.* Eric is every publisher's dream; of course his first "real book" would be a bestseller! (*The Lean Startup* was timed to go along with the second annual Startup Lessons Learned conference. We were at the first one, selling printed copies of *Startup Lessons Learned.*)

Flexible Rails

From looking at the historical precedents and immediate precursors to Lean Publishing, I now turn my attention inward, to the event which began my journey to Lean Publishing. It began almost accidentally, in 2006, as a side effect of the way I began writing

[41]http://theleanstartup.com/

Flexible Rails.

I'm now sitting here 7 years later, digging through the digital pieces, and realizing that I almost had Lean Publishing figured out back then–but that I missed the point for a few reasons.

1. Much of the core Lean Publishing approach was so obvious to me that I didn't ever recognize it as an idea. It was just the right way to do things.
2. I was distracted by the specific idea I was working on (*Flexible Rails*) that I didn't question whether the approach could be systematized (in Lean Publishing) and productized (in Leanpub). In fact, even when I had flickers of that idea I shied away from thinking it through for some reason. It could be that I had what Paul Graham has called "schlep blindness"[42]–the act of productizing Lean Publishing into Leanpub has definitely been a schlep.
3. I had not yet suffered enough. To recognize the profound importance of lightweight tools and processes you need to suffer through heavyweight tools and processes. Since I'm a stubborn masochist, this took me two books worth of suffering to fully grasp.

I alluded to all this in the introduction to this book, of course. I'm now going to tell the story of *Flexible Rails*, as it relates to the evolution of the Lean Publishing ideas. First, however, I need to give a bit of background about *Flexible Rails* itself, so that you have a clue what I'm talking about.

A Self-Described Alpha Book

Flexible Rails was a book about how to combine two technologies, Adobe Flex and Ruby on Rails, to build dynamic web applications called "Rich Internet Applications". Like many things, it seemed like a good idea at the time (2006-2007).

[42]http://www.paulgraham.com/schlep.html

In this section, I'm going to analyze what I wrote in 2006 in the "Acknowledgments for the Alpha Version" section in the first release[43] of *Flexible Rails*, to get a sense of what I was thinking about both the technology combination and the path I had chosen for the book:

> On January 31, 2006, after over a year and a half of working with Flex and over six months of playing with Rails (building toy apps, reading *Agile Web Development with Rails*, etc) I finally realized that for many applications Rails was the perfect server-side technology to complement Flex—and on the flip-side, that Flex offered capabilities that were either difficult, impossible, buggy or merely annoying to do on with JavaScript / AJAX / DHTML on the client side. (Especially if, like me, you're not a JavaScript guru like Thomas Fuchs.) Furthermore, while RJS templates are very promising, at the end of the day you are still dealing with the joy of HTML, JavaScript, CSS and browser compatibility issues.

Now, back in 2006, what did I do when I had a really good idea? Procrastinate, usually:

> So, I did what I always do whenever I have a Really Great Idea: I registered a domain name. I wanted a name which would be good name for promoting a possible book5 about using Flex and Rails together, so the natural choice was flexiblerails.com. I also got flexiblerails.net and .org since I was so sure of how good an idea this was.
>
> I then did what I also typically do whenever I have a Really Great Idea:

[43]This was on Lulu, but I removed it when it moved to Manning. So, there's no proper URL for this. I'm just looking at my own PDF copy of the first version.

> *Nothing.*
>
> Between the demands of my job and my two-year-old son, I was too busy, too tired, etc. Besides, I had a lot of Really Great Ideas (and domain names to go with them!), and I wasn't acting on any of them— what made this one worth doing? So, time passed.

Wow, I sound a lot like Hugo Lindgren did! For a software developer, I think that domain names and half-baked prototypes are the equivalent of a shoebox of unfinished manuscripts. When I try explaining what I do to people like my son or my father, I think I finally know how to do it.

I then carried on, explaining how I finally got going, and the process that led me to ship the first version:

> So, here we are. I hope that this book is useful to you. I'm releasing it in early alpha form, with many revisions to come in the evenings and weekends over the months ahead.

When I launched *Flexible Rails*, I described it as an "Alpha Book". Here's what I meant:

> Almost everything in "Web 2.0" (which seems to have been reduced to meaning "2006") is being released in "Beta". In fact, the Pragmatic Programmers originated and popularized the concept of a Beta Book. (Unlike most betas, their Beta Books are actually very useful.) The concept of a Beta Book is that of a book which is mostly complete, but contains of errors, typos etc. A reader who purchases a Beta Book gets regular upgrades until the final version. However, future editions must be purchased again.
>
> This book is being released as an Alpha book. Why Alpha? Well, not only is it going to be full of errors,

> typos, etc, but it is also rather incomplete. (My guess is that it's about 20% to 40% complete.)

I explained my rationale for launching early as follows:

> I am releasing it at a slightly earlier stage than the typical Beta Book for the following reasons:
>
> - The code in it is at the point where it would be useful for someone who is getting started on a project like this (even if it saves them only an hour, the book is worth it).
> - Since I have no technical or copy editor, the earlier I get feedback from the community the better focused the book can be. For example: Are code examples in a table with the explanation beside them readable, or should I just list all the code with a number in an explanation column (and then provide numbered explanations at the end)?

It turns out the answer to this question about the code formatting was an emphatic "NO". Based on feedback, I completely reformatted the over-200 page book. (Had I released the book earlier, I would have saved many hours of time!)

> Are the long code examples understandable, or are they too long?

The answer to this was that they were understandable, and probably the best feature of *Flexible Rails*. I believed that readers should be able to fully follow along based solely on the material in the book, and build a non-trivial code example. In this regard, *Flexible Rails* helped set a bit of a trend, as most code books beforehand had much shorter code examples, and the "build a non-trivial thing as part of reading the book" approach became popular after it. I really don't know how much of an impact *Flexible Rails* had in this regard; I do know that I belabored the point about the superiority of long form code samples at length to Manning, like I belabor most points to everyone about everything.

I also had questions about the shorter code samples:

> Are the short "add three lines of code here" examples easy to follow, or do readers have problems figuring out where to put the code? These are the types of questions which will be better answered if I release the book as early as possible.

Here was how I described my audience for the book:

> - This book has three potential audiences that I am aware of: (1) Flex developers who are possibly interested in Rails, (2) Rails developers who are possibly interested in Flex, and (3) developers without significant Rails or Flex experience who are considering doing a project in either or both. I have no idea right now about the distribution of what my actual audience will be—i.e. which type of developer the topic of this book will appeal to the most. (For example, it could be that most Rails developers hate the idea of Flex—or of using a Windows PC for a few months until Flex Builder is released for Mac—so fully that this book is only interesting to Flex developers choosing a server-side technology to integrate with.)

Releasing the book early let me test these assumptions and see what type of developer was really reading the book.

Finally, I had realized I wanted to build a community:

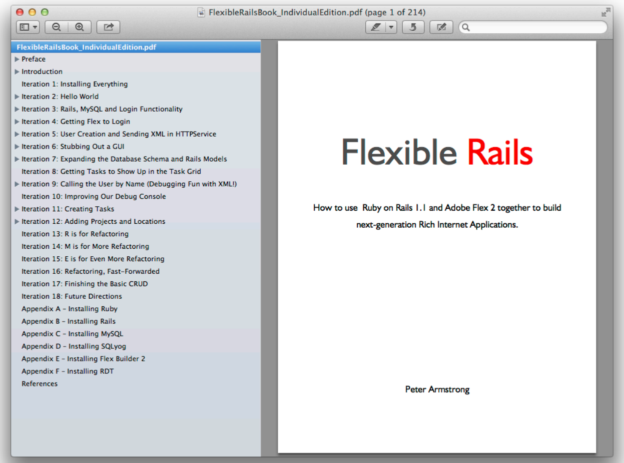
> - I want this book to be the starting point for a lively discussion at the http://www.flexiblerails.com

> blog. Getting a first draft released which gives a hint about the possibilities in the combination of the technologies is essential for this.

This idea of getting early feedback and building community is a core part of the Lean Publishing ideas.

However, you need to do more than just publish early to build community. You also need to publish often, and put effort into the community itself. Doing this can have good results, while the book is in-progress and for the book launch itself.

I released 25 self-published versions of *Flexible Rails* in the period from September 9, 2006 to September 16, 2007.



FlexibleRailsBook_IndividualEdition.pdf (page 1 of 214)

FlexibleRailsBook_IndividualEdition.pdf
- Preface
- Introduction
- Iteration 1: Installing Everything
- Iteration 2: Hello World
- Iteration 3: Rails, MySQL and Login Functionality
- Iteration 4: Getting Flex to Login
- Iteration 5: User Creation and Sending XML in HTTPService
- Iteration 6: Stubbing Out a GUI
- Iteration 7: Expanding the Database Schema and Rails Models
- Iteration 8: Getting Tasks to Show Up in the Task Grid
- Iteration 9: Calling the User by Name (Debugging Fun with XML!)
- Iteration 10: Improving Our Debug Console
- Iteration 11: Creating Tasks
- Iteration 12: Adding Projects and Locations
- Iteration 13: R is for Refactoring
- Iteration 14: M is for More Refactoring
- Iteration 15: E is for Even More Refactoring
- Iteration 16: Refactoring, Fast-Forwarded
- Iteration 17: Finishing the Basic CRUD
- Iteration 18: Future Directions
- Appendix A - Installing Ruby
- Appendix B - Installing Rails
- Appendix C - Installing MySQL
- Appendix D - Installing SQLyog
- Appendix E - Installing Flex Builder 2
- Appendix F - Installing RDT
- References

Flexible Rails

How to use Ruby on Rails 1.1 and Adobe Flex 2 together to build next-generation Rich Internet Applications.

Peter Armstrong

First alpha version of *Flexible Rails*

To give you a sense of the process, these were the releases by release date, page count (8.5x11 pages), reader count and total revenue before the next release.

Release Date	Page Count	# Readers	Total Royalties
2006-09-09	214	30	$560.00
2006-09-15	212	44	$784.00
2006-09-26	233	55	$960.00
2006-10-05	276	107	$1816.00
2006-11-08	309	121	$2040.00
2006-11-18	312	133	$2232.00
2006-11-30	342	143	$2392.00
2006-12-07	366	145	$2424.00
2006-12-09	371	183	$3032.00
2007-01-11	399	319	$5208.00
2007-04-01	506	335	$5464.00
2007-04-09	529	417	$6776.00
2007-05-17	689	458	$7456.00
2007-06-03	399	465	$7568.00
2007-06-05	401	501	$8144.00
2007-06-18	448	510	$8288.00
2007-06-21	471	517	$8400.00
2007-06-24	476	549	$8912.00
2007-07-04	440	619	$10,078.40
2007-07-29	475	644	$10,478.40
2007-08-06	481	647	$10,549.60
2007-08-06	482	647	$10,549.60
2007-08-07	484	671	$10,956.80
2007-08-19	497	743	$12,178.40
2007-09-16	495	830	$13,593.60

By the end of this process in September 2007, *Flexible Rails* the #73 all-time book on Lulu by revenue, as an unfinished PDF.

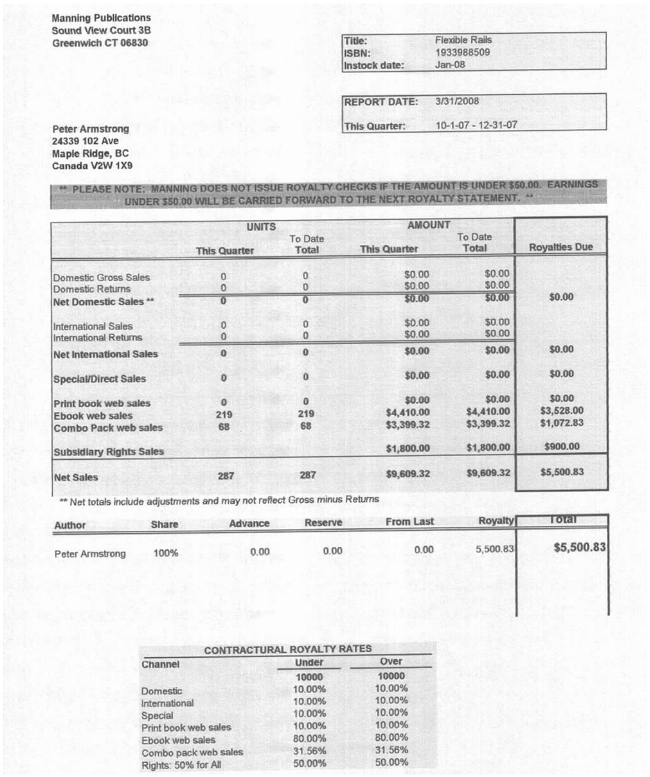
Releasing really early lets you gauge demand: when I released Flexible Rails, I considered the very real possibility that only 5 people in the world would care. If this happened, I could have just sent them their money back (or even double their money back), and I wouldn't have wasted the time writing a book nobody wanted. But in the first 6 days I had 30 readers, and I knew that I was committed. At that point, I had no choice but to finish writing the book. This is essential, since the whole process took about 1000 hours. Without public commitment, it would have been easy to stop.

Throughout the process of writing and self-publishing *Flexible Rails*, the total number of readers remained fairly small: under 1000. However, I still earned more than many authors did at the time from technical books–before the book was even done.

Furthermore, the buzz I had built ensured that when it went over to Manning it did well for me:

- In Q4 2007, the last 3 months that *Flexible Rails* was in-progress, I earned $3,528 in ebook royalties alone
- In Q4 2008, the quarter that *Flexible Rails* launched, I earned $9,976 in ebook royalties and $18,173.59 in total royalties

The exact details of those two quarters are shown in my royalty statements below:



Manning Publications
Sound View Court 3B
Greenwich CT 06830

Title:	Flexible Rails
ISBN:	1933988509
Instock date:	Jan-08

REPORT DATE:	3/31/2008
This Quarter:	10-1-07 - 12-31-07

Peter Armstrong
24339 102 Ave
Maple Ridge, BC
Canada V2W 1X9

**** PLEASE NOTE: MANNING DOES NOT ISSUE ROYALTY CHECKS IF THE AMOUNT IS UNDER $50.00. EARNINGS UNDER $50.00 WILL BE CARRIED FORWARD TO THE NEXT ROYALTY STATEMENT. ****

	UNITS		**AMOUNT**		
	This Quarter	To Date **Total**	**This Quarter**	To Date **Total**	**Royalties Due**
Domestic Gross Sales	0	0	$0.00	$0.00	
Domestic Returns	0	0	$0.00	$0.00	
Net Domestic Sales **	**0**	**0**	**$0.00**	**$0.00**	**$0.00**
International Sales	0	0	$0.00	$0.00	
International Returns	0	0	$0.00	$0.00	
Net International Sales	**0**	**0**	**$0.00**	**$0.00**	**$0.00**
Special/Direct Sales	**0**	**0**	**$0.00**	**$0.00**	**$0.00**
Print book web sales	**0**	**0**	**$0.00**	**$0.00**	**$0.00**
Ebook web sales	**219**	**219**	**$4,410.00**	**$4,410.00**	**$3,528.00**
Combo Pack web sales	**68**	**68**	**$3,399.32**	**$3,399.32**	**$1,072.83**
Subsidiary Rights Sales			**$1,800.00**	**$1,800.00**	**$900.00**
Net Sales	**287**	**287**	**$9,609.32**	**$9,609.32**	**$5,500.83**

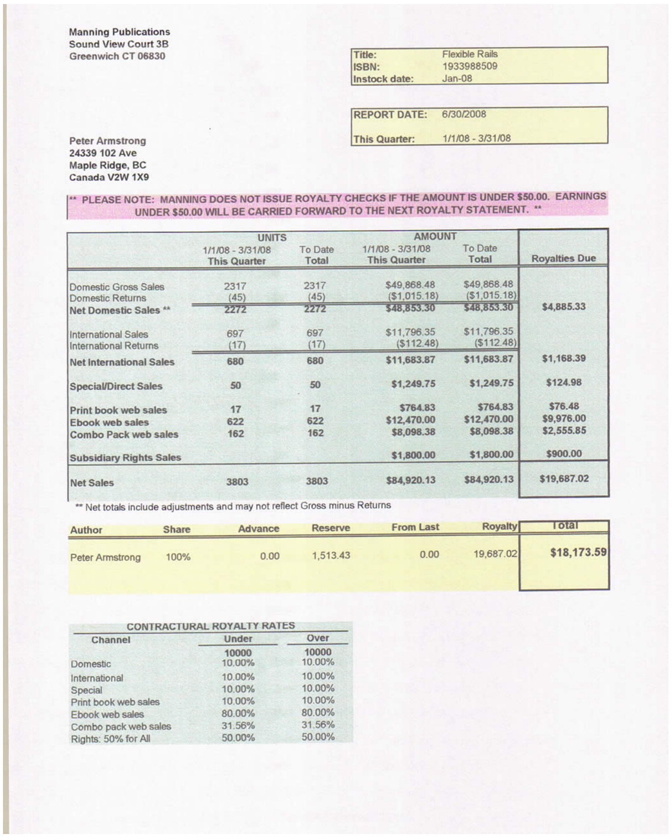
*** Net totals include adjustments and may not reflect Gross minus Returns*

Author	**Share**	**Advance**	**Reserve**	**From Last**	**Royalty**	**Total**
Peter Armstrong	100%	0.00	0.00	0.00	5,500.83	**$5,500.83**

CONTRACTUAL ROYALTY RATES

Channel	**Under** 10000	**Over** 10000
Domestic	10.00%	10.00%
International	10.00%	10.00%
Special	10.00%	10.00%
Print book web sales	10.00%	10.00%
Ebook web sales	80.00%	80.00%
Combo pack web sales	31.56%	31.56%
Rights: 50% for All	50.00%	50.00%

Q4 2007, the last in-progress quarter



Manning Publications
Sound View Court 3B
Greenwich CT 06830

Title:	Flexible Rails
ISBN:	1933988509
Instock date:	Jan-08

REPORT DATE:	6/30/2008
This Quarter:	1/1/08 - 3/31/08

Peter Armstrong
24339 102 Ave
Maple Ridge, BC
Canada V2W 1X9

** PLEASE NOTE: MANNING DOES NOT ISSUE ROYALTY CHECKS IF THE AMOUNT IS UNDER $50.00. EARNINGS UNDER $50.00 WILL BE CARRIED FORWARD TO THE NEXT ROYALTY STATEMENT. **

	UNITS		AMOUNT		
	1/1/08 - 3/31/08 This Quarter	To Date Total	1/1/08 - 3/31/08 This Quarter	To Date Total	Royalties Due
Domestic Gross Sales	2317	2317	$49,868.48	$49,868.48	
Domestic Returns	(45)	(45)	($1,015.18)	($1,015.18)	
Net Domestic Sales **	**2272**	**2272**	**$48,853.30**	**$48,853.30**	**$4,885.33**
International Sales	697	697	$11,796.35	$11,796.35	
International Returns	(17)	(17)	($112.48)	($112.48)	
Net International Sales	**680**	**680**	**$11,683.87**	**$11,683.87**	**$1,168.39**
Special/Direct Sales	**50**	**50**	**$1,249.75**	**$1,249.75**	**$124.98**
Print book web sales	**17**	**17**	**$764.83**	**$764.83**	**$76.48**
Ebook web sales	**622**	**622**	**$12,470.00**	**$12,470.00**	**$9,976.00**
Combo Pack web sales	**162**	**162**	**$8,098.38**	**$8,098.38**	**$2,555.85**
Subsidiary Rights Sales			**$1,800.00**	**$1,800.00**	**$900.00**
Net Sales	**3803**	**3803**	**$84,920.13**	**$84,920.13**	**$19,687.02**

** Net totals include adjustments and may not reflect Gross minus Returns

Author	Share	Advance	Reserve	From Last	Royalty	Total
Peter Armstrong	100%	0.00	1,513.43	0.00	19,687.02	**$18,173.59**

CONTRACTURAL ROYALTY RATES

Channel	Under 10000	Over 10000
Domestic	10.00%	10.00%
International	10.00%	10.00%
Special	10.00%	10.00%
Print book web sales	10.00%	10.00%
Ebook web sales	80.00%	80.00%
Combo pack web sales	31.56%	31.56%
Rights: 50% for All	50.00%	50.00%

Q1 2008, the launch quarter

Publishing in-progress helps build buzz for finished book launches, a fact that is overlooked even today. And yes: Launches matter.

All told, *Flexible Rails* earned me about $50,000. Also, it was translated into German and Japanese, which was personally gratifying. More importantly, it also helped me launch my consulting company, Ruboss, about 5 years ago and gave us enough credibility to attract our first customers. Furthermore, Ruboss went on to build Leanpub as a result of these experiences. So, while the royalties were nice, more importantly, writing *Flexible Rails* changed my life.

In short, with *Flexible Rails*, I published early, published often, listened to my readers, built a strong community around the book and had a successful book launch of the finished book.

Before moving on, I want to look more closely at the community dynamics around the book, as this went even better than planned.

Community Building

One of the most interesting things about *Flexible Rails* was the process of writing with an active community. This is interesting since it shows how social the writing of a computer programming book can be–and I'm normally not a social person in small groups.

Right at the beginning with *Flexible Rails* I created a Google Group for it.

Why?

There were two main reasons:

1. I wanted there to be a place for people to discuss the book, with me and with each other.
2. I needed a way to distribute updates, and uploading files to a Google Group seemed like a nice low-effort approach.

The second reason was actually because of limitations of Lulu[44]. I was self-publishing *Flexible Rails* on Lulu as a PDF. Lulu is a fantastic website for selling completed print books. However, the process of selling an in-progress book on Lulu was cumbersome.

For example, Lulu did not tell you who bought your book, or provide any way for you to contact them. So, how do you give them free updates to your book?

The way that I solved this was as follows: Lulu allows you to include a "thank you note" with a purchase, which is included in the confirmation email which is sent to the reader when they purchase the book. This thank you note was something that had to be the same for all books you had published. Luckily for me, that number

[44]http://lulu.com

was 1, so I could use the thank you note as a way of providing instructions to people who had purchased *Flexible Rails*.

So, that's what I did. In the thank you note, I provided instructions for how to download the book from a secret URL (http://www.flexiblerails.com/Rumpelstiltskin/SecurityThroughObscurity.html, which I still consider to be one of my best jokes), and also from the Google Group.

I actually had two Google Groups, one[45] for announcements and discussion and another[46] for announcements only. That may have been overkill. I posted every release of *Flexible Rails* to the "files" sections of both Google Groups, and then sent an email to both groups.

Here's what I said about this process in my thank you note:

> Yes, this is terribly inefficient. However, I'm actually going to argue that this is a feature, not a bug:
>
> The way Lulu is set up, ALL of your order information is kept private from me. This ensures that your privacy is protected. However, it does mean that it is impossible for me to automatically add your email address to the Google group when you purchase the book–I do not have access to it! With privacy comes inconvenience, I guess.

Ironically, with Leanpub, we've had to weigh similar issues as we've built our feature set.

The process I set up clearly wasn't easy! I needed secret URLs, two Google Groups and a lot of email! However, even though it took a lot of effort, it worked.

The first message I sent to the *Flexible Rails* Google Group was the following:

> Aug 18, 2006: Test message to Google Group.

[45]http://groups.google.com/group/flexiblerails
[46]http://groups.google.com/group/flexiblerails-announce

I sent this message a few weeks before first publishing the book (on September 9), but even this content-free message got a reply after the book was released and a few people were on the list:

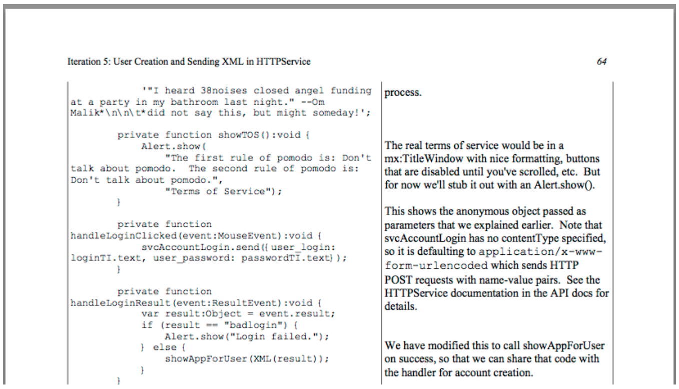
> ignore this reply to your test message - i, too, am testing that i can test messages to the group.
>
> planning to read the alpha over the w/end so hopefully will be able to add something constructive next week.
>
> btw peter i've read a few chapters and i like your dry sense of humour - are you by chance a fellow brit?

For me, this reader encouragement was really helpful. Somebody from Britain was reading my book, and they liked it!

Feedback

Besides encouragement, I got lots of meaningful information about how to improve the book. This was both in terms of spontaneous feedback and in responses to questions I asked.

For example, when I launched the book I was using a 2-column code + explanation format that looked like this:



Iteration 5: User Creation and Sending XML in HTTPService *64*

```
            '"I heard 38noises closed angel funding
at a party in my bathroom last night." --Om
Malik*\n\n\t*did not say this, but might someday!';

        private function showTOS():void {
            Alert.show(
                "The first rule of pomodo is: Don't
talk about pomodo.  The second rule of pomodo is:
Don't talk about pomodo.",
                "Terms of Service");
        }

        private function
handleLoginClicked(event:MouseEvent):void {
            svcAccountLogin.send({ user_login:
loginTI.text, user_password: passwordTI.text});
        }

        private function
handleLoginResult(event:ResultEvent):void {
            var result:Object = event.result;
            if (result == "badlogin") {
                Alert.show("Login failed.");
            } else {
                showAppForUser(XML(result));
            }
        }
```

process.

The real terms of service would be in a mx:TitleWindow with nice formatting, buttons that are disabled until you've scrolled, etc. But for now we'll stub it out with an Alert.show().

This shows the anonymous object passed as parameters that we explained earlier. Note that svcAccountLogin has no contentType specified, so it is defaulting to `application/x-www-form-urlencoded` which sends HTTP POST requests with name-value pairs. See the HTTPService documentation in the API docs for details.

We have modified this to call showAppForUser on success, so that we can share that code with the handler for account creation.

2-column layout

This seemed like a good idea when I started, but as the book writing progressed it got tedious and I started to question the format. Some readers had criticized it, and it was annoying to deal with while writing, since it meant my explanations needed to roughly match the length of the code.

So I sent an email to the Google Group with the subject of "POLL: code example format change".

> Hi all,
>
> The code samples in the 2-column Code + Example format have been the subject of some well-deserved criticism: they are frustrating to copy and paste out of, and are also fairly ugly and hard to read.
>
> So, before I do a bunch of reformatting, I want to get a sense of how people feel so far...
>
> Poll:
>
> Should the code samples in the 2-column table Code + Explanation format be:
>
> a) left alone
>
> b) one column of code only, with numbers placed in code comments which refer to explanations after the code sample
>
> c) two columns, the first column code (but a lot wider) and the second column an extremely narrow column which contains just numbers which refer to explanations after the code sample
>
> For b and c, the explanations after the code sample would be a numbered list.
>
> I'm a bit on the fence between (b) and (c). The (b) choice gives you an explicit pointer from the code (when you're looking at it in Flex Builder or TextMate)

> to the explanation in the book (which might be helpful), but it might just be a bunch of clutter. However, even if it is clutter, it would be easier to copy and paste out of than choice (c). Note that both (b) and (c) would definitely make the code more readable, however due to the length of some of the code examples there would be definitely some flipping back and forth between code and explanation. Is this a good tradeoff?
>
> If you have an opinion about this, please reply either to the list or to me at peterarmstrong@gmail.com.
>
> Thanks!
>
> Peter

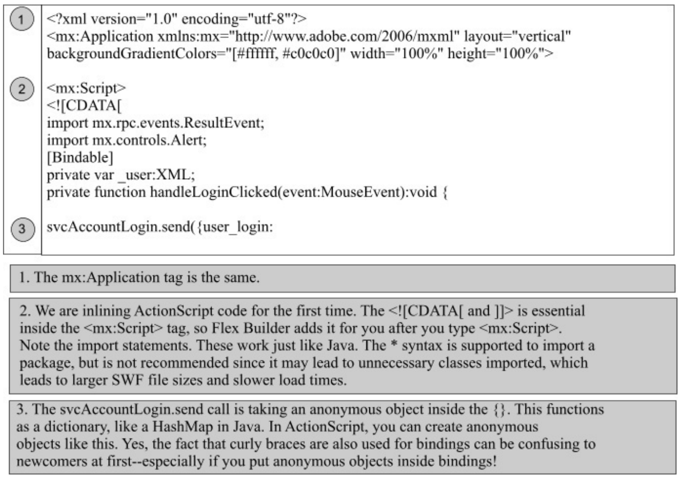
I got lots of meaningful feedback about this, from people living all over the world. Someone in The Netherlands (!) liked option (b). But this was my favorite response:

> Hi Peter, et. al.,
>
> I think this is a pretty important subject - I thought I was the only one that disliked the current format!
>
> I'm happy to go with option 2. I've tried to visualise it here:
>
> http://www.ipauland.com/layout2.jpg
>
> I did flirt with boxed inline explanations
>
> http://www.ipauland.com/layout.jpg, but now favour option 2.
>
> It's a great book and importantly for you, it's currently the only book, so you have a good opportunity here. One of the really interesting things about the book is that you show us what goes wrong and why, when people do the 'obvious' thing. I like that, but it may age this book very quickly if the RoR software changes that behaviour.

Paul

To emphasize what happened here: *a reader of my in-progress book cared enough about it to create mockups of what it could look like with a better layout.*

Here was one of Paul's mockups:



```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml" layout="vertical"
backgroundGradientColors="[#ffffff, #c0c0c0]" width="100%" height="100%">

<mx:Script>
<![CDATA[
import mx.rpc.events.ResultEvent;
import mx.controls.Alert;
[Bindable]
private var _user:XML;
private function handleLoginClicked(event:MouseEvent):void {

svcAccountLogin.send({user_login:
```

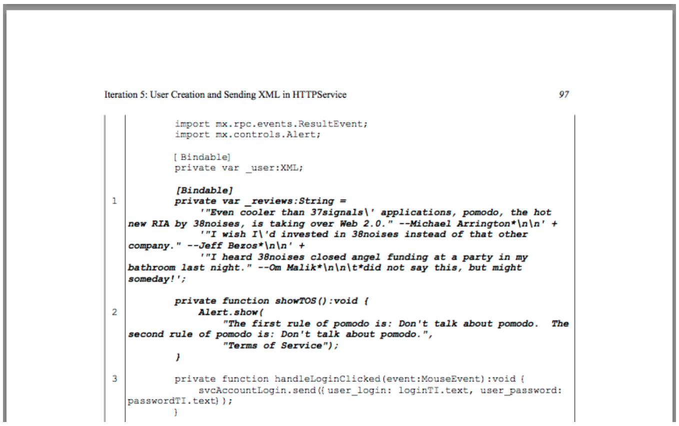
1. The mx:Application tag is the same.

2. We are inlining ActionScript code for the first time. The <![CDATA[and]]> is essential inside the <mx:Script> tag, so Flex Builder adds it for you after you type <mx:Script>. Note the import statements. These work just like Java. The * syntax is supported to import a package, but is not recommended since it may lead to unnecessary classes imported, which leads to larger SWF file sizes and slower load times.

3. The svcAccountLogin.send call is taking an anonymous object inside the {}. This functions as a dictionary, like a HashMap in Java. In ActionScript, you can create anonymous objects like this. Yes, the fact that curly braces are also used for bindings can be confusing to newcomers at first--especially if you put anonymous objects inside bindings!

Improved layout mockup

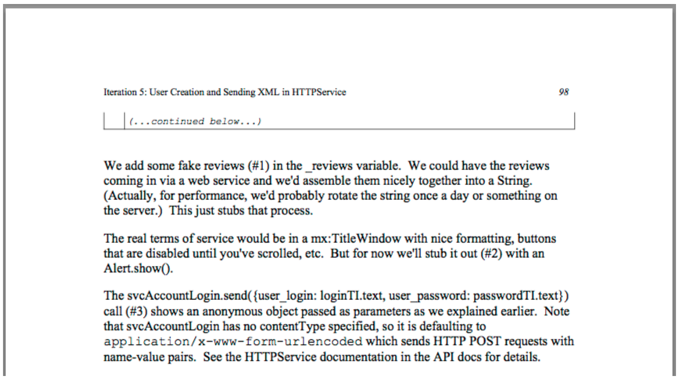
I liked it so much I basically went something similar in the next version.



Iteration 5: User Creation and Sending XML in HTTPService 97

```
1         import mx.rpc.events.ResultEvent;
          import mx.controls.Alert;

          [ Bindable]
          private var _user:XML;

          [Bindable]
          private var _reviews:String =
              '"Even cooler than 37signals\' applications, pomodo, the hot
new RIA by 38noises, is taking over Web 2.0." --Michael Arrington*\n\n' +
              '"I wish I\'d invested in 38noises instead of that other
company." --Jeff Bezos*\n\n' +
              '"I heard 38noises closed angel funding at a party in my
bathroom last night." --Om Malik*\n\n\t*did not say this, but might
someday!';

          private function showTOS():void {
2             Alert.show(
                  "The first rule of pomodo is: Don't talk about pomodo.  The
second rule of pomodo is: Don't talk about pomodo.",
                  "Terms of Service");
          }

3         private function handleLoginClicked(event:MouseEvent):void {
              svcAccountLogin.send({ user_login: loginTI.text, user_password:
passwordTI.text} );
          }
```

Improved Flexible Rails Layout #1



Iteration 5: User Creation and Sending XML in HTTPService 98

```
(...continued below...)
```

We add some fake reviews (#1) in the _reviews variable. We could have the reviews coming in via a web service and we'd assemble them nicely together into a String. (Actually, for performance, we'd probably rotate the string once a day or something on the server.) This just stubs that process.

The real terms of service would be in a mx:TitleWindow with nice formatting, buttons that are disabled until you've scrolled, etc. But for now we'll stub it out (#2) with an Alert.show().

The svcAccountLogin.send({user_login: loginTI.text, user_password: passwordTI.text}) call (#3) shows an anonymous object passed as parameters as we explained earlier. Note that svcAccountLogin has no contentType specified, so it is defaulting to `application/x-www-form-urlencoded` which sends HTTP POST requests with name-value pairs. See the HTTPService documentation in the API docs for details.

Improved Flexible Rails Layout #2

Incorporating reader feedback meant that I could get even better reader feedback. For example, at one point I was trying to figure out a few things:

- How experienced with Flex and Rails was the typical reader?
- How familiar with REST (a software architecture approach) was the typical reader?

- What version of Rails should I use? (Writing about Rails was writing about a rapidly moving target.)

So, since I had a Google Group full of readers, guess what I could do?

Ask them!

(I sound like Eric Ries here. Hooray for me!)

I sent out an email with the following poll in it, which consisted of 9 multiple choice questions:

1. Do you already have or plan to buy the second edition of Agile Web Development with Rails?

    - [] yes
    - [] maybe
    - [] no

2. Before this book, what was your Flex experience?

    - [] none
    - [] beginner
    - [] intermediate
    - [] expert

3. Before this book, what was your Rails experience?

    - [] none
    - [] beginner (read some tutorials online)
    - [] intermediate (read AWDwR and built at least one complete Rails app)
    - [] expert

4. Have you already read the first edition of AWDwR?

    - [] yes
    - [] no

5. What is your knowledge of REST?

- [] none
- [] only what I read in AWDwR 2nd ed.
- [] I've read about it in AWDwR 2nd ed. and in online tutorials and used it in a Rails app already

6. Do you have any interest in seeing RESTful Rails controllers talking to Flex?

    - [] yes
    - [] no
    - [] what the heck are you talking about?

7. When Rails 1.2 comes out, when will you upgrade to it?

    - [] not for 6 months or so until it's well-tested
    - [] within a week
    - [] I'm already using it to follow along with AWDwR 2nd. ed.
    - [] I've been on Edge Rails for over a year and only read this book for the Flex stuff

8. Regarding basic Rails stuff (e.g. has_many, belongs_to) that does not get affected in any way by using Flex with Rails, do you want me to...

    - [] assume you already know it
    - [] give a 1-2 sentence summary and refer to AWDwR
    - [] explain it briefly
    - [] explain it as though you will never buy another Rails book

9. Do you use

    - [] Flex Builder 2 with Windows
    - [] Flex Builder 2 Beta with Mac
    - [] Flex Framework SDK with Windows
    - [] Flex Framework SDK with Mac

Once again, I got invaluable reader feedback, with many readers filling in the poll answers and emailing me or the Google Group. This changed the entire direction of the book, and ensured that the book would be relevant for about a year longer than it would have been.

More importantly from the perspective of Lean Publishing, it showed me how engaged a community of readers can be. If you make the effort to reach out to your readers and include them in your creative process (especially for a technical book!), the benefit you get is astounding.

As I wrote *Flexible Rails* in public, I was approached by a few publishers, who I turned down. (I had made a few promises, such as discounts on the print book, which the publishers could not easily keep. Also, I was making a great royalty percentage, and decent money as-is.) I then wrote an email to the Google Group explaining my reasoning about turning down a book deal. The subject was "Flexible Rails is staying independent: BaaS". And in the email, I started realizing that the process which had been created was something new.

> …
>
> One thing that a couple of readers have said to me is that they saw this book as more of a service and community than "just a book". I agree: this is something that just kind of evolved accidentally over the last 10 (!) months, but which I'm really happy about. I have felt that this book is as much a group conversation as anything else, and there is no reason to change that now.
>
> So, I am starting to see Flexible Rails as an example of a new model in publishing: BaaS (Books as a Service). [I couldn't help parodying SaaS.] Since PDF books, like software, have essentially zero variable cost, they should also have a proper upgrade process–just like

> software. (This is especially true with a topic which is as fast-moving as the Flex + Rails combination.)
>
> In my opinion, this is the next step in the evolution of the "Beta Book" process pioneered by the Pragmatic Programmers. I am not saying that it is right for every book–a book like Getting Real probably wouldn't have benefited from it as much. However, I think that the BaaS model is the right model for this book and potentially for many others. The excellent royalty rates and control to the author that Lulu offers enables a lot more creativity in this regard.
>
> Anyway, that's enough navel-gazing for now–back to working on the WebORB iteration :)
>
> …

Then a funny thing happened: people pushed back, arguing that when my book was done, that I should take a book deal since it would grow the community, and that I deserved it. I ended up doing so, with Manning. Not wanting a deal gave me the negotiating power to get a *really good one.* Plus, Manning enabled me to keep my promises, and the people I interacted with at Manning were first-class. Finally, the book that resulted was much better for their efforts. But, compared to my subsequent two attempts at Manning books, the book that resulted was also better because I started the traditional publishing process *later*, after I had product/market fit and traction.

Practice

A Sports Metaphor for Publishers

If you're a publisher, you may feel that the world has become a very different place. Instead of the warmth of paper, you're confronted with the cold, unfamiliar reality of ebooks.

What if 75% of the revenue and 97% of the profit from the books you sell ends up coming from ebooks? How do you proceed? How do you succeed?

The answer may lie in the following thought experiment.

Imagine a world in which there were no minor leagues for sports.

Say the only time that people played sports were in elementary and high school and at the pro level–a world in which, once you're 18, if you're not a pro, you have nowhere to improve your skills and establish your reputation. (Pretend that universities decided to focus on education, or that they got scared of concussions or other liability risks, or that they got bored of all the revenue.)

Also, imagine that at the pro level, teams were compelled by their current collective bargaining agreements to maintain their rosters at roughly the same number of people as now, and to offer similar contracts as nowadays, but to relatively unproven talent.

How would pro teams proceed?

With great difficulty.

Sure, they would still get it right with some superstars, such as high-profile basketball (LeBron James, Kobe Bryant) and football[47] (Wayne Rooney, Lionel Messi[48]) players who made it clear to the world from a very early age that they were destined to perform at the highest level. And they would choose many of the same successes that established themselves later. But the percentage of

[47]By which I mean the "kick the ball with your feet" variety (soccer), not the "throw and catch" North American variety.

[48]Of course, if they could not offer those contracts until the age of 18, Lionel Messi literally would not have measured up to his current standard.

successes would be a lot worse, and the corresponding percentage of expensive failures would be a lot higher. Professional sports is big business, and teams do everything they can to reduce risk. In baseball, hockey and football in particular, there is an elaborate progression of youth and minor leages, so that talent can be assessed early, and the most talented young players developed. (In football, the transfer fees paid for top players justify the expense of the entire pyramid. It's like a stock market crossed with a soap opera, and it's almost more entertaining than some of the matches themselves.)

Similarly, publishers take steps to avoid risks. They use an author's track record and the estimated target market of the book to judge the potential of the book and the likelihood of a successful outcome.

This could be taken much further. Earlier, we discussed O'Reilly's Realtime Publishing program, which proposed to replace the $40,000 cost of producing a 500 page print book with a $4,000 cost of producing a 50-page ebook. However, there's no reason that a 50-page ebook has to cost $4,000, or that a 500-page ebook or print book has to cost $40,000. *After all, publishers should only spend meaningful amounts of money on ebooks or print books when there is an expected return on investment.*

But how could this work?

Say instead of spending $4,000 up-front on ebooks, publishers created an equivalent of the minor leagues that sports teams used.

Pretend a new book project started like this:

1. Proposal accepted for publisher's Lean Publishing program via a standard contract.
2. The author writes and publishes in-progress versions of his or her book on the publisher's Lean Publishing site. An automated toolchain is used so that these versions are produced at essentially no cost to the publisher.
3. During this process, the author can have up to 3 hours of development editor time, for early guidance.
4. A completed manuscript is produced using this process.

At this point, the cost to the publisher isn't $4,000. Instead, it should be doable for about $400.

So, the publisher can accept 10x as many books into this process, compared to a $4,000 per-book process–and 100x as many books compared to a $40,000 per-book process.

Also, most books should earn the publisher over $400 *while part of this process* even with relatively low sales: the margins on each ebook sale are essentially pure profit. (In this way, ebooks do resemble software, and publishers should very much want to become software companies!)

So, given this, publishers can essentially accept almost as many books into their Lean Publishing programs as they consider suitable: the Lean Publishing programs should be self-funding. Furthermore, there's no reason that the books need to be 50 page ebooks. The manuscripts can be anything from 50 pages to hundreds of pages: the author is doing *all the work*, writing the book and engaging with a community of readers. Then, when the manuscripts are finished, the publishers can make educated bets on the best manuscripts and put them through the full editorial and production process, and invest additional marketing work behind them etc.

This can be done with **considerably** more information than publishers historically have had about a book. Here are some of things that the books coming out of this type of Lean Publishing process that you would be further investing in would have:

- a finished manuscript
- hundreds of readers
- buzz on social media (Twitter, Facebook, etc.)
- thousands of dollars in revenue
- growth in readership, revenue and buzz shortly before launch

In a word: **traction**.

This is the exact same thing that a Series A investor looks for from companies that have spent their seed round money building a product. You can be an incubator as well as a publisher.

As a publisher, imagine if all the books you spent non-trivial amounts of money on publishing had *all of the above going into your full publishing process.*

Your job would be to take mildly (*Flexible Rails*) or wildly (*Fifty Shades of Grey*) successful books and *make them more successful.*

Sounds like a nice way forward, if you ask me.

Now, as a publisher or as an author, do you need to build all this yourself?

Thankfully, no.

Instead, you can use Leanpub to do the full Lean Publishing process. Everything from the writing process to the storefront just works. This is true whether you are an author or a publisher.

Leanpub: Lean Publishing as a Service

I'm the cofounder of Leanpub. We're building Leanpub so that every author and publisher can use the Lean Publishing process without having to build all the machinery needed to do so. We see ourselves as building "Lean Publishing as a Service." We're putting our time and money where my mouth is. Leanpub is what I wish had existed when I was writing *Flexible Rails.*

We provide a lightweight, fully automated toolchain, to both self-published authors and to publishers looking to implement a Lean Publishing process.

We launched in 2010 as as a free service for self-published authors. In 2012, we launched our publisher program. This is also a free service. We're a free writing workflow which is funded by our optional storefront. (But since the storefront is so good for Lean Publishing, people use it and we both make money.)

In 2013, we're launching our "white label" publisher program, so that publishers can use Leanpub for Lean Publishing completely under their own brand.

Here's how Leanpub works:

1. Authors write in Markdown.

2. Authors save their work in a Dropbox folder.
3. Authors preview or publish PDF, EPUB and MOBI with one click.

This is entirely free.

Besides generating the ebooks in all 3 formats, we provide an attractive storefront.

This storefront features a variable pricing feature, with effective minimum and suggested price sliders.

We pay excellent royalties (**90% minus 50 cents per copy**, so an author or publisher earns **$8.50 on a $10 book**, and **$17.50 on a $20 book**) and automatic update distribution.

Leanpub has evolved along with the ideas in this manifesto. When we launched, we were entirely focused on self-published authors; specifically, on technical book authors and bloggers. As we have grown, we have branched out into serial fiction, cookbooks, and anything else that our authors feel like trying on Leanpub.

At Leanpub, we encourage authors to launch a Minimum Viable Book and then iterate.

What's a Minimum Viable Book? It's the smallest in-progress subset of your book that you could sell and be able to claim with a straight face that it is worth money *right now*.

For a novel, why not publish in serial? Release the first few chapters. What worked for Mary Elizabeth Braddon can work for you.

For a technical book, once your book can save an advanced reader a couple of hours–which should be true after you have written a couple of chapters–you have written a Minimum Viable Book–if you can save an advanced reader two hours, your book will save a novice reader twenty hours. In both cases, the reader will have received excellent value for his or her money, as saving an expert two hours or a novice twenty hours should be worth a few hundred dollars in any technical profession.

Now, it's up to you whether you price the book at what you think it will end up being worth when it's done, or whether you price it at what you think it's worth right now. There's no right answer here, but you should indicate to readers what you are doing. At Leanpub, not only do we offer variable pricing, we also have an unconditional 45-day 100% refund policy; we want your readers to be happy.

After publishing the first version of your book, the next step is to start marketing and selling your in-progress book. You need to blog and tweet about it. If it's a programming book, getting it on the front page of Hacker News helps!

At this point, you should see some slow trickle of sales and followers.

Now comes the hard part!

Don't think that when you've written a Minimum Viable Book that you are even close to being done your book.

If you really released the book early enough, you should have only spent about 10% of the total time you will end up spending on the book. Chances are it will have taken you at least 40 hours to write the Minimum Viable Book, meaning that you should expect to spend at least 400 hours to complete a first draft.

After you've released the first version, you need to stay focused and keep the stream of releases flowing. Thankfully, assuming you have some early readers, your early readers will be there to provide encouragement–or to nag you if you haven't updated your book in two months!

Also, while you're doing this, you also need to blog and tweet about your book, engage with your readers, etc.

Lean Publishing is hard work, not a Get Rich Quick scheme. Unlike much book writing today, however, it is also not a Get Poor Slow scheme.

If you are using the above process as a self-published author, you will have some decisions to make after your first draft is done:

Do you self-publish (and if so, ebook only, or also a print book?) or do you work with a publisher?

The fact that this is even a question shows how much the world has already changed.

If you decide to self-publish a print book, sites such as CreateSpace do a *great* job of taking a PDF (such as one produced by Leanpub) and producing and selling it, providing order fulfillment and remitting your royalties. You should also stick your Leanpub ebooks in the Amazon KDP and Apple iTunes Connect storefronts for discoverability purposes. Amazon KDP is really easy to work with.

If you have built a strong following, however, you may have publishers fighting over you. After all, you will have already done the two hardest parts: finishing a book draft and building an audience. For a publisher, this means you are starting far ahead of other authors, so you have negotiating power. You should recognize that good publishers do add value to a book. If you have a completed manuscript, you can still get the benefit of much of this value. Things like copy editing, indexing and print book distribution and marketing are areas where publishers excel and authors especially benefit from their services.

Now, you can also use the above process on Leanpub as a publisher who is working with authors. Besides being the best way in the world for authors to self-publish in-progress ebooks, we have also added a publisher program[49], so that the full Leanpub toolchain is available for publishers to use. And in 2013, we're building a "white label" version of this, so that your brand is prominent and the Leanpub brand essentially disappears.

The always-quotable Marc Andreessen recently stated[50] that "software is eating the world." We agree. But does this mean that every publisher needs to become a software company? We hope not. We want to provide the infrastructure, so that publishers can focus on what their core competencies: working with their authors to make the best books possible, produce really well-designed print books, and to market these books as best as possible. We feel that the ebook production process should be something that publishers,

[49]https://leanpub.com/p

[50]http://online.wsj.com/article/SB10001424053111903480904576512250915629460.html

like authors, get for free as a byproduct of how manuscripts are written. And rather than having to invent this process themselves, they can just use ours.

The process of helping build Leanpub, and seeing what our authors have done with it, has taught me some valuable lessons. Frankly, it has pushed my thinking farther than it could ever have grown in a vacuum–which is, of course, essentially the entire point of using a Lean approach. This book has evolved along with Leanpub. Like Leanpub, it has evolved from focusing exclusively on self-published authors, to focusing on any author or publisher who wishes to apply the Lean Publishing principles. And like this book, Leanpub has been in customer development for the past couple of years.

We have evolved Leanpub in a very Lean way, engaging with our early adopter authors, both on Twitter and in our excellent Leanpub Google Group[51]. And as we expand Leanpub with our publisher program, we will again evolve it with feedback from our early adopter publishers.

If this sounds interesting to you, either as an author or publisher, please talk to us. My email is peter@leanpub.com[52]. Or, you can email our whole company by emailing hello@leanpub.com[53].

Thanks very much for reading.

[51]https://groups.google.com/group/leanpub?pli=1
[52]mailto:peter@leanpub.com
[53]mailto:hello@leanpub.com

Appendices

Besides the usual stuff you find in appendices, I am also including a nerdy discussion of Lean Publishing outcomes which will probably only appeal to you if you like statistics. Since there's a statistically-significant (!) chance that you don't like statistics, I put it in an appendix.

Real Artists Reject the Null Hypothesis

In the philosophy of science, something is "not even wrong"[54] if it cannot be falsified, that is, if there is no way to devise an experiment which can test the supposed theory. Writing is similar too. Some books are good; many books are bad. Some books (not necessarily the same ones!) are successful; many books are failures. However, many books even fail to become failed books. These books are *not even bad* – they are abandoned.

The sad thing is that it's a mathematical certainty that some of these books would have been good, if they had only been completed. These books could have enriched the authors' lives and that of their readers.

So, the problem can be summarized as follows: How do we help authors ensure that the books which should get written are written? And how should we help authors ensure that the books which should not get written are not written?

This problem is very similar to the concept from statistics of Type I and Type II Errors[55]. A Type I Error is a false-positive: a true null hypothesis is incorrectly rejected. A Type II error is a false negative: a false null hypothesis is incorrectly not rejected.

What's a null hypothesis? For books, it's this:

Null Hypothesis (H0): The book should not be written.

The different combinations of outcomes can be expressed in the following table:

[54]http://en.wikipedia.org/wiki/Not_even_wrong
[55]http://en.wikipedia.org/wiki/Type_I_and_type_II_errors

Outcome	**H0 is true**	**H0 is false**
Reject the null hypothesis (H0). Book is written.	False Positive (Type I Error)	True Positive (correct)
Fail to reject the null hypothesis (H0) Book is not written.	True Negative (correct)	False Negative (Type II Error)

This table contains two good outcomes and two bad outcomes. The bad outcomes are as follows:

- False Positive (Type I Error): In this outcome, we have a book which *should not be written* (null hypothesis is true), but the author rejects the null hypothesis and *writes the book.* The author and many of the book's readers are unhappy, or the book has hardly any readers.
- False Negative (Type II Error): In this outcome, we have a book which *should be written* (null hypothesis is false), but the author does not reject the null hypothesis and *does not write the book.* The author and many potential readers miss an opportunity.

The good outcomes are as follows:

- True Positive: In this outcome, we have a book which *should be written* (null hypothesis is false), and the author correctly rejects the null hypothesis and *writes the book.* The book is good, and the author and many of the book's readers are happy.
- True Negative: In this outcome, we have a book which *should not be written* (null hypothesis is true), and the author correctly does not reject the null hypothesis and *does not write the book.* The author's time is not wasted, and neither is that of the people who would have read the book.

Lean Publishing helps ensure that both errors above happen less often, and also improves both good outcomes.

This is how Lean Publishing helps mitigate the errors:

- False Postive (Type I Error): In this outcome, we have a book which *should not be written* (null hypothesis is true), but the author rejects the null hypothesis and *writes the book*. With Lean Publishing, the author publishes and sells the in-progress book, in order to get feedback and traction. If the book should not be written since there is no market, hardly anyone will buy it, which should give the author a clue that the book may be a bad idea. If the book should not be written since the author is a terrible writer (either in general, or in the specific genre), hopefully the author will receive enough negative feedback to either improve or stop.
- False Negative (Type II Error): In this outcome, we have a book which *should be written* (null hypothesis is false), but the author does not reject the null hypothesis and *does not write the book*. Sometimes, this decision is made after much of a book is written, but the book is abandoned. With Lean Publishing, the author publishes and sells the in-progress book. So, if it is an error for the author to stop writing, this helps prevent that error. Readers encourage authors, both with their feedback and by having actually purchased the book.

Here's how Lean Publishing helps improve the positive outcomes:

- True Positive: In this outcome, we have a book which *should be written* (null hypothesis is false), and the author correctly rejects the null hypothesis and *writes the book*. This is the case where a book which should be written is written. All the benefits of Lean Publishing described in this book apply. The author publishes the in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until he or she has the right book and build traction once he or she does.

- True Negative: In this outcome, we have a book which *should not be written* (null hypothesis is true), and the author correctly does not reject the null hypothesis and *does not write the book.* In this outcome, a book can either get not written when it's at the idea stage or at the partially-completed manuscript stage. Authors often toil on manuscripts for a long time before abandoning them. Lean Publishing helps expedite this process, as the book is published in-progress. If it is terrible or has no market (or both), this can be discovered while wasting as little time as possible.

Lean Publishing is not a silver bullet though.

One valid criticism is that while helping with reducing Type I Errors (false positives where authors write books they should not), it can make some authors vulnerable to Type II Errors (false negatives where authors do not write books that they should write). Say a self-published author is good at writing but terrible at marketing. If she uses the Lean Publishing approach, it could be that she will not get the word out very effectively, and thus have few sales. This may cause her to abandon the book prematurely. However, if she had persisted in writing the book, she could possibly have written enough good content that authentic reader buzz could have happened, or that she could have attracted the interest of a publisher that was good at marketing. Or, even if the book was a commercial failure, the act of writing it could have enriched her life in other ways. This is a valid point. The two ways to respond to it are as follows:

1. If the book is being written to be a commercial success, a failure in marketing in the in-progress stage would likely also happen at the finished-book stage. So, this is not as much of a concern.
2. If the book is being written regardless of commercial success, then the value of publishing in-progress is for feedback, not money. This feedback can be useful even if it's only from

> a handful of readers. In this sense, the Lean Publishing approach can resemble a writing support group for the author.

This criticism aside, Lean Publishing helps ensure that books which should be written are, that books which should not be written are not, and helps authors judge which is which. It then helps amplify the success of books which should be written, and help authors move on to the next book for the books which should not be written.

Acknowledgments

This is a short book, so I'll keep the acknowledgments short too.

First, I'd like to thank my cofounder Scott Patten, for believing in me and in the Lean Publishing idea enough to create Leanpub with me. None of this would have happened otherwise. We've been through the wars to get everything to where it is now.

Next, I'd like to thank Len Epp, who besides doing great work on Leanpub, also gave me great feedback on this book. At one point I thought I was done, but Len read a draft and essentially pointed out that people liked to read stories, and I should try telling some. I've known Len for more than half my life, and I'm really happy to have the chance to work with him on Leanpub.

Ken Pratt and Steve Luscher are two of the most talented software developers I've ever had the privelege of working with, either in Vancouver or in Silicon Valley, and I'd like to thank both of them for helping make Leanpub what it is today.

The Masters of Publishing (MPub) community at Simon Fraser University in Vancouver has been fantastic as well. John Maxwell (no, not Mary Elizabeth Braddon's publisher husband, a different John Maxwell!), Suzanne Norman and Rowland Lorimer have been very encouraging of my ideas, inviting me to speak at SFU about Lean Publishing on a couple of occasions, as well as to speak at the Mini TOC that we had in Vancouver in fall 2012. This talk led to my speaking at TOC 2013 in New York, and to my doing the research into Victorian serial fiction as practiced by Charles Dickens, Wilkie Collins and Mary Elizabeth Braddon that made the ideas finally come together. (I'd also like to thank Suzanne and John for many stimulating discussions about publishing, and also for recording my Mini TOC talk!)

Speaking of TOC, anyone who is involved with TOC or a Mini TOC knows that Kat Meyer from O'Reilly is absolutely amazing. I'm especially grateful to her, since it's because she liked my Mini TOC talk that I ended up speaking at TOC 2013 in New York.

Speaking of academia, I'd also like to thank Len's brother Mike Epp, who thought that my Lean Publishing ideas were interesting years ago, and had me talk about them to a class of his graduate students way back when I was calling them "Publishing 2.0". It was really helpful to get early encouragement for my ideas, since at that point I was used to only talking about technical matters (Flex and Rails) instead of just a pure idea talk. This helped get the ball rolling.

Last but not least, I'd like to thank my wife Caroline and my son Evan for putting up with me for yet another book. While this one wasn't as much of an effort as the others, it was still distracting and disruptive. Furthermore, since this book is so intertwined with Leanpub, in that sense this story has been, and continues to be, much more effort. Thanks for supporting me through it.

References

- Braddon, Mary Elizabeth, *Lady Audley's Secret*, (London: Oxford University Press, 2012). Originally published in serial from 1861 to 1862.
- Carnell, Jennifer, *The Literary Lives of Mary Elizabeth Braddon*, (Hastings, The Sensation Press, 2000).
- Chintz, David E., *T. S. Eliot and the Cultural Divide*, (Chicago and London: The University of Chicago Press, 2003).
- Collins, Wilkie, *The Moonstone*, (London: Oxford University Press, 1957). Originally published in serial in *All the Year*

Round from 4 January to 8 August, 1868.

- Cox, Jessica, *New Perspectives on Mary Elizabeth Braddon*, (Amsterdam, Editions Rodopi B.V., 2012).
- Dickens, Charles, *The Posthumous Papers of the Pickwick Club*
- Law, Graham, *Serializing Fiction in the Victorian Press*, (New York: Palgrave Press, 2000).
- Hedrick, Joan D. Harriet Beecher Stowe: A Life. New York: Oxford University Press, 1995: 208. ISBN 9780195096392208
- Tomaiuolo, Saverio, *In Lady Audley's Shadow*, (Edinburgh: Edinburgh University Press Ltd., 2010).
- Tromp, Marlene et. al., *Beyond Sensation: Mary Elizabeth Braddon in Context*, (Albany: State University of New York Press, 2000).
- Wolff, Robert Lee, *Sensational Victorian: The Life & Fiction of Mary Elizabeth Braddon*, (New York & London: Garland Publishing, Inc., 1979).
- "Serialization and the Nature of Uncle Tom's Cabin", by Susan Belasco Smith, in *Periodical Literature in Nineteenth-Century America*
- http://dearauthor.com/features/industry-news/master-of-the-universe-versus-fifty-shades-by-e-l-james-comparison
- http://en.wikipedia.org/wiki/Anna_Karenina
- http://en.wikipedia.org/wiki/Arthur_Conan_Doyle
- http://en.wikipedia.org/wiki/Brothers_Grimm
- http://en.wikipedia.org/wiki/Bugs_Bunny
- http://en.wikipedia.org/wiki/Campfire
- http://en.wikipedia.org/wiki/Charles_Dickens
- http://en.wikipedia.org/wiki/Die_Gartenlaube
- http://en.wikipedia.org/wiki/Fan_fiction
- http://en.wikipedia.org/wiki/File:Classic_bugsbunny.png
- http://en.wikipedia.org/wiki/File:Mary_Elizabeth_Maxwell_(n%C3%A9e_Braddon)_by_William_Powell_Frith.jpg

- http://en.wikipedia.org/wiki/File:The_Writings_of_Charles_Dickens_v1_p38_(engraving).jpg
- http://en.wikipedia.org/wiki/Fyodor_Dostoyevsky
- http://en.wikipedia.org/wiki/Harriet_Beecher_Stowe
- http://en.wikipedia.org/wiki/Homer
- http://en.wikipedia.org/wiki/Laozi
- http://en.wikipedia.org/wiki/Leo_Tolstoy
- http://en.wikipedia.org/wiki/Madame_Bovary
- http://en.wikipedia.org/wiki/Matteo_Bandello
- http://en.wikipedia.org/wiki/Nimrod
- http://en.wikipedia.org/wiki/Not_even_wrong
- http://en.wikipedia.org/wiki/One_Thousand_and_One_Nights
- http://en.wikipedia.org/wiki/Oral_storytelling
- http://en.wikipedia.org/wiki/Robert_Seymour_(illustrator)
- http://en.wikipedia.org/wiki/Sam_Weller_(fictional_character)
- http://en.wikipedia.org/wiki/Serial_(literature)
- http://en.wikipedia.org/wiki/Serialized_novel
- http://en.wikipedia.org/wiki/Sherlock_Holmes
- http://en.wikipedia.org/wiki/Sketches_by_Boz
- http://en.wikipedia.org/wiki/Storytelling
- http://en.wikipedia.org/wiki/Tao_Te_Ching
- http://en.wikipedia.org/wiki/The_Brothers_Karamazov
- http://en.wikipedia.org/wiki/The_Old_Curiosity_Shop
- http://en.wikipedia.org/wiki/The_Tragical_History_of_Romeus_and_Juliet
- http://en.wikipedia.org/wiki/Type_I_and_type_II_errors
- http://en.wikipedia.org/wiki/Uncle_Tom's_Cabin
- http://en.wikipedia.org/wiki/War_and_Peace
- http://en.wikipedia.org/wiki/William_Painter_(author)
- http://en.wikisource.org/wiki/The_Encyclopedia_Americana_(1920)/Pickwick_Papers
- http://fiftyshadesofpopculturetheory.blogspot.ca/2012/03/full-exchange-with-jason-boog-for.html

- http://latimesblogs.latimes.com/jacketcopy/2012/05/the-origins-of-50-shades-of-grey-go-missing.html
- http://online.wsj.com/article/SB10001424052702303734204577464411825970488 html
- http://oreillynet.com/oreilly/authors/welcome/realtimepublishing.csp
- http://tvtropes.org/pmwiki/pmwiki.php/Main/AscendedFanFic
- http://web.archive.org/web/20110617051831/http://www.50shades.com/
- http://www.crushable.com/2012/05/11/entertainment/el-james-snowqueens-icedragon-fifty-shades-of-grey-twilight-fandom-wank-860/
- http://www.guardian.co.uk/books/2012/aug/13/fan-fiction-fifty-shades-grey
- http://www.guardian.co.uk/books/2012/dec/09/el-james-fifty-shades-interview
- http://www.guardian.co.uk/books/2012/jan/30/self-e-publishing-bubble-ewan-morrison
- http://www.guardian.co.uk/books/2012/jul/30/tweet-about-cats-just-write
- http://www.hollywoodreporter.com/news/twilight-fanfiction-hit-office-gets-387539
- http://www.mediabistro.com/galleycat/fifty-shades-of-grey-wayback-machine_b49124
- http://www.nytimes.com/2013/01/06/magazine/be-wrong-as-fast-as-you-can.html?pagewanted=2&_r=1&pagewanted=print
- http://www.standard.co.uk/news/found--tombstone-of-cartoonist-charles-dickens-wrote-out-of-history-6487574.html
- http://www.thepassivevoice.com/05/2012/fifty-shades-a-copy-of-twilight-or-not/
- http://www3.iath.virginia.edu/wnr4c/Raabe.Era.UTC.Diss.pdf