

Extreme Publishing

brian d foy

Abstract

The tenets of Extreme Programming include feature stories, simplicity, short release cycles, planning for change, and on-site customers. I illustrate these self-referentially by applying to them to how we publish this magazine, which I call Extreme Publishing, but you can apply them to almost any project on which you might work.

1 Short stories

In Extreme Programming (XP), customers and developers describe product features in terms of user stories which are short descriptions of the feature along with the value that the feature provides. The team divides the stories by the value that they add to the product, how long they take to implement, and how important they are to other features. The *Extreme Programming* series recommends using index cards for the stories so it is easy to move them around, annotate them, rip them up, or separate them into stacks.

Once the team sorts the stories, they decide how soon they can deliver at least some value. You do not have to completely finish the entire project to make something useful, and you should start with the least amount of work to finish the first story that provides value.

When I first thought about starting a new Perl magazine, I gave myself a six-month cooling off period to do my homework about what it would take to do everything. I started creating stories. Here is a list of some of the story titles I have at the moment.

- Get subscribers
- Get articles from authors
- Edit articles
- Layout magazine
- Send magazine to printers
- Send magazine to subscribers

Based on these stories, XP tells me to find out how to deliver the most value to the customer as soon as possible by dividing them into short release cycles, where “short” means something measured in a small number of weeks rather than months. Anything much longer than a month is too long. Let developers go off for months at a time without deliverables and you set yourself up for a lot of anguish.

I start going through the stories to see what I can put off until later. Do I need to send something to the printers and then subscribers or can I put that part off? That depends on what the customer and I think the value really is. Does the customer want content or form first? How much work do I have to do to deal

with printers and subscribers and how much immediate value does that add? It turns out that dealing with subscribers, printers, and the post office is a lot of work, and that particular work delays immediate value.

I decided, after talking to a lot of people, that if I could bootstrap the magazine to the point where I could publish a print version, then you, as the customer, get more value sooner. I maximize value by publishing an electronic version, but I also finish more stories by publishing PostScript (or its lesser, more accessible cousin, PDF) because I can send postscript to a service bureau which can turn it into a magazine. Thus, I make the first milestone an electronic “preview” issue.

2 Simplicity

The key to XP is simplicity, whether in design, process, or implementation. Work smarter rather than harder. Developers break up stories that are complex into two or more simpler stories. I broke down the goal, sending you a print magazine, into much more simple stories, such as laying out the content, so that I could complete stories quickly, which maintains a sense of accomplishment, which XP calls the *project velocity*. Higher project velocities keep teams motivated. A continuing series of small successes motivates more than the spectre of one big failure.

Simplicity also requires me to not go too far off the beaten path. I could have used software like Adobe Framemaker or Quark, which require me to worry about all sorts of details. Some printers require these formats, and have long troubleshooting lists for them. I could have re-invented the wheel by starting my publishing quest by writing a Perl module to handle the layout. After months of work on that maybe I would have something moderately useful. However, my goal is not to write Perl modules, or any code for that matter. None of my stories say that I need to produce code, and any time I start coding I am doing something that should support the project, rather than *be* the project.

I chose T_EX, or specifically L^AT_EX, which I already knew, I did not have to pay for, produces beautiful output almost completely on its own, and already has magazine modules on the Comprehensive T_EX Archive Network, which is the sort of thing Perl people like. Good service bureaus (and in this case, your printer) can deal with postscript without the hassle of some of the commercial formats. Additionally, L^AT_EX, in its simplest form, can be analogous to POD, which a lot of Perl people already know. A competent programmer could write a workable pod2latex in a half hour if it did not already come with Perl. The editors can accept submissions in POD, a perly thing to do, which lets the author think about the writing rather than the format.

3 You are the customer

Another major tenet of XP is the Onsite Customer, which I simply call the Customer. The Customer adds to the value because he can work directly with the developers, modify and enhance the stories, and learn about the product along the way. At the end of the process the Customer gets what he wants because he has been fully engaged along the way. Too many teams try to guess at what the Customer wants. They end up with what they want, and the Customer ends up with something they have to tolerate.

Now that I have completed my first milestone, which was to put this into your hands, you get to be a Virtual Customer. Although XP would prefer that you actually come over to sit in my apartment while I do this, perhaps as a pair publisher, is just not going to happen. Later we might have some place which you can visit, and then you can be an Onsite Customer. Until then, its your job to help me modify and enhance the stories for the next milestone. What can we do to provide the most value in the next release cycle, which we measure in weeks? How has your idea about our end product, a magazine devoted to Perl, changed?

The stories in the pile for the next release cycle include some of the same stories from the first cycle, which we have to repeat, but now have some practice doing, as well as some new ones. We incrementally add new features so we do not be bogged down at the start of the project.

- Produce an A4 layout
- Sell some ads to cover production costs
- Collect subscribers for future issues
- Include a book review column
- Include an ongoing Perl Golf column

If you do your part, and we do our part, at the end of the process, when we have a completed product, you should have something close to your dream Perl magazine. One editor has already told me that if I do not include an article about beer in every issue, he will be disappointed. So, maybe most people will get their dream Perl magazine at least.

4 Future directions

I want to include at least something that mentions XP, or other things that help us work smarter rather than harder, in each issue. Some XP concepts I did not discuss include unit and acceptance tests (with PerlUnit perhaps), planning for change, the 40 hour work week, or pair programming. If you would like to write about these, and can relate them to Perl, let us know.

Remember to be our Virtual Onsite Customer, too. We do not plan too far into the future so that we can be more flexible. If you do not speak up, we can not give you what you want.

If you have not experienced XP before, check out some of the references. Get your managers to accept the XP methodology, then, once they have signed-off on XP, show them the chapter on the 40-hour work week.

5 References

You can read more about XP in these fine books:

Extreme Programming Explained - Kent Beck

Extreme Programming Installed - Ron Jeffries, *et al*

Extreme Programming Examined - Giancarlo Succi and Michele Marchesi