

Extreme Publishing: Postmortems

brian d foy

Abstract

The tenets of Extreme Programming include user stories, short release cycles, and story postmortems. I illustrate these self-referentially by applying them to how we publish *The Perl Review* but you can apply them to almost any project on which you might work.

1 Project velocity

Extreme programming (XP) identifies parts of the project in terms of “user stories” that describe what the users, in this case *The Perl Review* readers, authors, and editors, need from the project and how they interact with the product. A user story tells the tale of a particular person’s interaction with whatever the project produces.

The XP process emphasizes short release cycles and frequent testing so that the development team knows the health of the project on a day-to-day basis, rather than wondering if the system will work when they reach their delivery deadline. Furthermore, the short release cycle allows the users to constantly evaluate if the product meets their user story requirements.

This cycle of daily, or shorter, testing and integration allows the team to constantly evaluate the difficulties involved with the user stories and adjust the project schedule to meet reality. No matter what the project diagrams show or what management might say, project tasks take a certain amount of time. Denying that fact just makes matters worse at delivery time, and does not improve the situation during the next cycle.

The “project velocity” measures how fast these tasks progress. In the last issue of *The Perl Review*, I wrote out my user stories for the production of the journal and made estimates about how long each part would take. I thought that producing the PDF document would take the most time, and that receiving articles from authors would be relatively easy. It turns out that it is actually the other way around. Had I not taken time to think about the differences in my estimates and reality, I would have a more difficult time planning the next issue, and may even make the same mistakes again.

Once I finished the basic research about the things I needed to do with \LaTeX , I discovered that a lot of people already had the same idea so I could reuse packages I found on the Comprehensive \TeX Archive Network, whose idea inspired the creation of the Comprehensive Perl Archive Network (CPAN). The Google Groups website (<http://groups.google.com>) saved me hours of work. When I went back to my user stories that required \LaTeX work, I gave them a high project velocity. Some things that I estimated a week to complete I finished in an afternoon. Indeed, for this issue, I spent less than an hour with new \LaTeX work, and I knew that I would.

Dealing with authors, however, took me much longer than I expected. Naturally, as any manager does, I failed to account for people’s lives outside of *The Perl Review*, which often demands much more of them whether it is their day job or their newborn daughter. The project velocity for submissions is slow, and I

need to take that into account in my planning. I need to start the article submission process much sooner than the production process. I need to solicit and secure articles a couple of issues ahead of publication time to keep things moving since the project velocity for submissions is actually longer than the production cycle for my first two issues.

2 Story postmortem

Once the team finishes a particular user story and has had time to not think about it (a cooling-off period), they can evaluate how well they did on the story. Part of that measures the project velocity, but the postmortem gives the team a chance to recognize other strengths or areas for improvement. Some people focus on the weaknesses, but if a team can identify strengths they can emphasize those in the next iteration.

Post-project evaluations occur in many different institutions, including the military. One of the most surprising books I have read is *Business is Combat* written by an United States Air Force fighter jet pilot. In it, James Murphy explained about how everything he does as a fighter pilot is about process and very little about flying, and applies this to the business cycle. No matter how tired he is or how long he was in the air his crew chief interviews him after each flight and he goes to the mission debriefing. Since the mission post-brief happens after absolutely every mission, the grumblings are virtually non-existent.

As software developers, if we decide on a process, and stick to the process, once we get used to it, its just part of the normal work day. In XP these things happen in short cycles, rather than big event quarterly meetings, so the postmortem process can take 30 minutes a week. A half hour should not kill the project schedule, and if done correctly saves that much more in the future.

We do not have this sort of thing as an industry-wide mind set the way professions like law and medicine, or in this case, the military does. Individual enterprises may have internal standards, but in my experience the book goes out the window at crunch time. You cannot improve things if you do not take the time to improve them.

In the United States Army, which makes me follow this process while on active duty, after any training or real-life mission the team conducts an after action review (AAR), which the Army explains extensively in its field manuals. The Army, as well as the other branches of service, has a manual for everything. The US Army has been around for 226 years and had a manual, the Blue Book drill and ceremony manual, before the United States of America won its independence.

The key to the AAR is improvement at both the team and individual levels. The AAR leader starts the session by asking the team what they meant to accomplish, which keeps first things first. The team does most of the talking while the leader maintains the process. From there, the team identifies what actually happened and how that affected the attainment of the goal. The team identifies strengths, which they should maintain, and areas for improvement. The language is important; we made two lists – “Maintain” and “Improve”, not “Success” and “Failure”. Focus on the positive to move things forward.

Our goal with the last issue was to publish a Perl magazine in PDF format. Out of all of the things that we did and some of the things that we did not do, we met our goal. We developed a layout, learned quite a bit of L^AT_EX, and published some articles. We list those in the “Maintain” list. We cannot forget about those because past performance does guarantee future success. We need to get articles sooner and edit them better. We list those in the “Improve” list. I discuss some of these in the rest of the article.

The postmortem can be as simple as that, especially with a good team that has worked together for a while. The postmortem becomes a fluid part of the process rather than something to schedule or tolerate.

3 Stories for this milestone

In the last issue I mentioned several stories that I thought might be important for this issue, and I solicited comments for the community about how important they might actually be to readers.

3.1 Produce an A4 layout

I asked a couple of people across the Pond how well the US-Letter layout of issue printed on A4 printers. Despite some extra white space, since A4 is somewhat larger than US-Letter, no one complained. If people start to complain about the US-Letter format, I might look at this user story again, but I probably will not have to do that once we get to the print version. I am still thinking about our international readers, though. I would like to produce the print version as close to the regional markets as possible, so a couple of people volunteered to research service bureaus in the European Union. Other markets need other volunteers.

3.2 Sell some ads to cover production costs

We still need to work on this, but besides labor costs, which has so far been volunteer work and nominal printing costs (although when you print it at work it's free), our production costs are very low. I have been researching service bureaus, and when we need to move to print we will have to pay real money to vendors. Once we get that far, we will want a commission-based sales manager to handle that for us. Anyone interested in doing that? I should also mention that Neil Bauman of GeekCruises (<http://www.geekcruises.com>) has been very helpful with suggestions about the business and production aspects of magazine publication.

3.3 Collect subscribers for future issues

Many readers wrote in to add their name to a list of future subscribers. I still do not know when a print issue will be available, but I do have a better way to collect subscribers. You can visit our web site to add your name to our list of future subscribers (<http://www.theperlreview.com/subscribe.html>).

3.4 Include a book review column

We had a book review in the last issue, and we do not have one in this issue, although we have three that we could have published. We were going to publish a review of *Perl Debugged*, but we want to save that for an upcoming issue we hope to devote to debugging. Some publishers sent me some review copies, and a couple of people have already submitted reviews. You can send us your own reviews (http://www.theperlreview.com/book_reviews.html). Publishers can also send review copies of their books to me (publisher@theperlreview.com).

3.5 Include an ongoing Perl Golf column

A couple of people got really excited about conducting Perl golf competitions, and we received a couple hundred entries. Dave Hoover and Jerome Quelin volunteered to run the competitions and have set up

a SourceForge project for things that support that. (<http://perlgolf.sourceforge.com>). The results of last month's golf challenge appears in this issue.

4 Stories for the next iteration

4.1 Author submission process

Authors need to know what to do and what happens during the editorial process, and the editors need to know how to move that process along. I need to figure out the process because the way I do it now takes too long and causes too much confusion. Since I identified this as the story with the lowest project velocity, I think a lot of my time in the next publication cycle, especially early in the cycle, goes towards making this situation easier.

This is a big task, however. When you encounter big tasks in XP, you create smaller tasks.

4.1.1 Define the process

I need to get together with the editors to codify how we do things, then publish that in the authors guide on the web site. Parts of the process include defining the intended writing style and laying out the technical and editorial review process. All of this starts when an author submits an article proposal from our website.

4.1.2 Let the authors know the status of their article

We are not full time editors, so we do not spend our full-time job interacting with authors. The authors should be able to check the status of their articles on our website. Editors can do this already, so we just need to go a couple of extra steps to turn that around to an author's perspective.

4.1.3 Assign editors to authors

I did not really have a good way to do this, and I did it pretty miserably this time around. Authors should get comments from one person who shepherds the article to publication. The author should know who his editor is, and the editors should know who their authors are. This should move the article along a consistent path rather than several editors trying to turn it into their conception of an article.

4.2 Get some ads

I need to figure out the advertising process. If this magazine makes it to print, advertising revenues can cover some of the operating expenses which means lower subscription fees. The ad sales process is well known and the people who buy ads know what needs to be done, but including ads in the magazine can be tricky. Ad placement, size, and scheduling issues concern me more, so I want to make the mistakes when I can fix them (by regenerating the PDF file) instead of when I have to give the money back (when I mess up in print).

5 Conclusion

I discussed a couple of points involved in the XP process: the project velocity and the postmortem. The project velocity helps us manage time by identifying long-lived tasks, while the postmortem evaluates tasks and identifies areas for improvement. Each of these help us do better during the next iteration. I illustrated these with examples from the publication process for this magazine by initially focusing on what we wanted to accomplish, what we did right, and how we could improve.

6 References

Business is Combat: A Fighter Pilot's Guide to Winning in Modern Business Warfare, James D. Murphy, Regan Books, March 2000.

You can read more about XP in these fine books:

Extreme Programming Explained, Giancarlo Succi & Michele Marchesi, Addison Wesley, 2001.

Extreme Programming Installed - Ron Jeffries, Ann Anderson, Chet Hendrickson, & Kent Beck, Addison-Wesley, October 2000.

Extreme Programming Examined - Giancarlo Succi & Michele Marchesi, Addison Wesley, 2001.