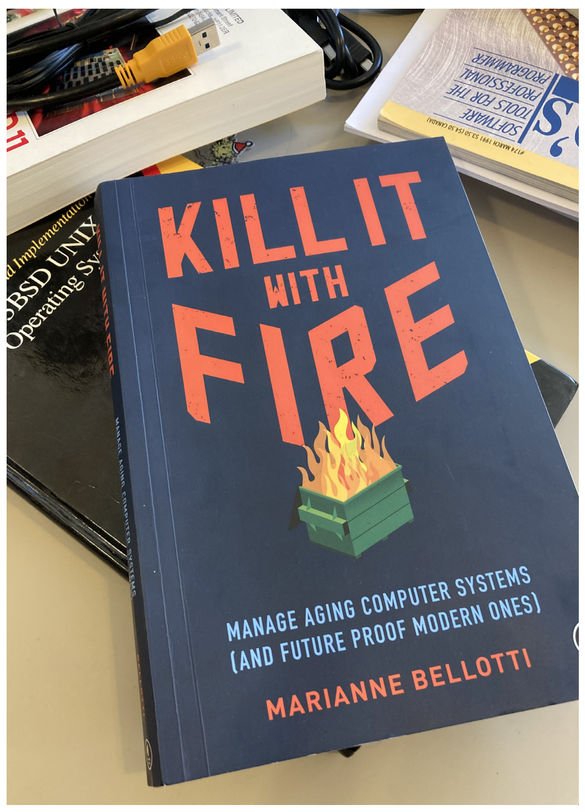
BOOK REVIEW

Kill It with Fire: Manage Aging Computer Systems (and Future Proof Modern Ones) by Marianne Bellotti

BOOK REVIEW BY TOM JONES



FreeBSD is a legacy system. Wait! Before you start finding things to throw, it is worth sitting and thinking about how we use this term. Legacy systems are the things we are stuck with, many of the things we are stuck with are old, creaky and have seen much better days. Legacy comes with a notion of neglect. Legacy systems are painful to use. I know I have a bias here, but I don't think FreeBSD is painful to use. But it does have one heck of a legacy. FreeBSD is quickly coming up on its 30th year as an established open source project. Our roots extend much further back into time, UNIX topped 50 a couple years ago and BSD UNIX is going to do so, too.

Throw a stone in any direction (not at me!) and you are going to hit a tool or subsystem that feels like it has been dredged out of the past. Large chunks of FreeBSD are like sharks or crocodiles, perfected long ago and mostly forgotten by evolution.

On the other hand, throw another one of your stones (again, not at me!) and you'll hit something in FreeBSD which is painfully modern and cutting-edge. While this seems counter to my initial claim, I still think that FreeBSD belongs in the legacy system crowd.

Kill it with Fire (KIWF) by Marianne Bellotti, takes the same stance as I do on FreeBSD (or maybe it helped me get to this position). It begins with an important point about legacy systems, they are almost by definition, incredibly successful. Many software projects struggle to see the light of day at all, but legacy projects not only make it into continued production, but they also become integral components of much larger systems. So much so, that something about their age causes them to become a liability.

FreeBSD is certainly a massive success--we haven't managed to take over the entire world, but we are still part of the conversation about modern operating systems. For many of the engineers that use our platform, it has been a key component of their ability to have outsized success.

But you really can't argue that FreeBSD isn't a legacy system, and it is a legacy in two ways. First, it has components that are aging and a little long in the tooth. Second, FreeBSD, being almost too easy to maintain, becomes a legacy component in other systems.

KIWF doesn't aim to be a technical manual to help you maintain a legacy code base, for that it points you to the renowned Working Effectively with Legacy Code by Michael C. Feathers. Instead, *KIWF* talks about the political and social mechanisms that lead systems to be considered legacy and the approaches we need to take within an organization to help move on from them or to solve the major hindrances they generate.

Many organizations that use FreeBSD as a component see our favorite OS as a legacy component. It is notably more difficult to hire engineers to work on FreeBSD systems--the supply side is much smaller than what is available for Linux or Windows platforms. FreeBSD also has a marketing problem. We may be loved by the engineers that use our platform, but there are few startups using FreeBSD as a core selling point.

Both are political issues for the continued use of FreeBSD. It is easy to sell someone on a new idea to replace a painful product as new is without bounds and limits and we all know the real pain of the tools we use day to day.

KIWF advocates for understanding why a system has succeeded enough for it to be considered legacy and provides some tools for approaching a design for new systems or refactoring to improve the existing components.

KIWF carries the subtitle: *Manage Aging Computer Systems (and Future Proof Modern Ones)*. The lessons from this book carry forward and should help you build protections into systems so they can succeed more easily and be more manageable in the long run when they do.

Many organizations that use FreeBSD as a component see our favorite OS as a legacy component.

KIWF is not a tutorial. While the introduction promises exercises, they can be a little thin. They do serve well as a starting point for helping your thinking about existing systems and your understanding of how to manage modernization projects. The advice here can help bridge the gap to the humans behind the maintenance and management of existing systems. There is advice for breaking down problems with teams, working through modernization projects, and keeping motivation high when replacing a component or updating it to match better with more modern practices.

KIWF is a great read if you have to defend the continued use of FreeBSD or another similar component in your environment. The measurement practices suggested by Marianne can be used to get quantified information about failure rates and usage, great tools for understanding how things are used, and evaluating any moves to other platforms.

I found a lot of value in this book; it has helped with my thinking about FreeBSD and how our OS is used in real environments. If FreeBSD is a part of your environment, or if you want to move from something else, then *Kill it With Fire* is a quick read that will help you have harder data to work from.

TOM JONES, FreeBSD Developer and co-host of the BSDNow Podcast, wants FreeBSD-based projects to get the attention they deserve. He lives in the North East of Scotland and offers FreeBSD consulting.