How do you evaluate new technologies?

W. Adam Koszek

Koszek ORG

wojciech@koszek.com

2022-10-05T17:47:28Z

"How do I get started with X" is probably one of the most frequently asked questions on technical forums. With 2.8 billion results from Google, somehow technical "AWS" manages to top the search results page:

As a software crowd, we all start to use new technologies at some point. While many technologies are fairly self-explanatory (having no "democratize" or "disruptive" in their name), many of them aren't, or simply have no feature that distinguishes them from other available options. How do **you** select new components for your newest product?

On the other hand, even experts can get lost within the sea of available tools,, and making a well informed decision about the appropriate tools for the job is sometimes problematic. Redis or Memcached? Sass or Less?

Quiz: With little time on your hands you want to pick a scriptable tool for your software bug tracker. Which tool would you use? Assume the tool is a commercial one.

Before becoming exposed to the software business and dirtying my hands with high-end Open Source code, I always believed that the industry's senior engineers and architects had secret cabal meetings, during which such decisions would be made. Meetings during which architecture and choices would involve only guys with beards, long hair and glasses. Youngsters with <= 25yrs of industry experience wouldn't be welcome.

Then the top guru of the clan wearing a dark cloak would speak up and sequentially ask each each of the masters whether their bag of spells could be used. Masters would each try to defend their own bag of spells with arguments, discussions, disagreements and fist fights until finally the strongest spell would win.

And in fact, yes, industry seems to work like that - perhaps without the spells, masters and cloaks, but the schema is very similar. In fact, the whole thing is

pretty primitive both for the commercial and the Open Source world, and it's quite likely that one of the following events takes place:

1. Technology is there and is so superior that you'll swallow the bullet and select it regardless of any inconveniences (e.g. it's written in Ruby and you like Python), since rewriting stuff from scratch makes absolutely no sense.
2. The most experienced person will select the technology for the team and you need to live with that choice.
3. The least experienced person responsible for doing real work will select the technology. Nobody cares, since only that very person will have to work using this technology.
4. Nobody will select any technology and the goal will be achieved in the simplest, often the least optimal way possible.
5. Some products will be evaluated quickly, often in a state of misconfiguration, and then some random decision will be made

For long term products there's always time allotted to test-drive a couple of these solutions and pick the one which is judged to be the best. Beware: "the best" is a subjective term. For Open Source developers who read forums and hang out on IRC it may indeed lead to selecting the best product out there. (Perhaps that's why the industry likes to pull in open-source people?) If you're running a startup in the Bay Area, "the best" may mean selecting the sexiest, possibly least stable technology and letting your crew waste some time on making it actually work. Yes, your developers will waste some time writing libraries which exist in the more mainstream choices available, but they'll be so happy, that the Ubers or Facebooks of this world won't seem quite so attractive to them. Industry veterans on the other hand will select whatever they know best and make it work really well, regardless.

How often have you seen a senior guy using a primitive arcane tool just because he/she knows it very well? How often have you thought to yourself that you would have done it better? How many times have you tried, and then reverted back to the method that just works?

"I'm a startup and I'd like to get my technology into the enterprise market"

In the past, I was a part of Xilinx and from what I could observe "the best" in a big company will quite likely mean the best price/quality ratio plus some combination of answers to the following questions:

- Does it have a support line?
- Is the technology stable and well established?
- Is the documentation easy to understand?
- Is it easy to use?

Open Source acceptance in the enterprise market keeps growing, but the projects which have no commercial body end up with a problem: if there's nobody to call in cases of technical problems, it's often too much of a responsibility for your Directory or VP to agree to bringing the technology on board only to struggle rewriting it one year later via a replacement commercial product. Unless support is available, it will be tossed away.

Have you ever participated in a meeting where technology choices have been made? What influenced your decision to use one technology over the other?

I'd like to hear about it.