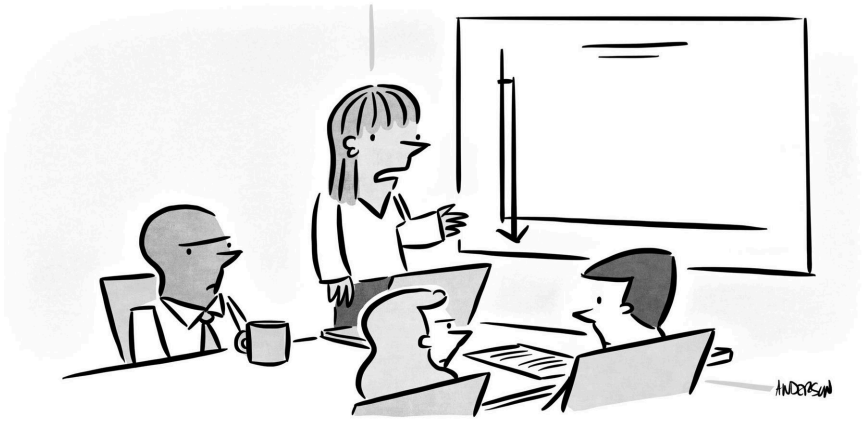
The “errors” that mean you’re doing it right

> *“If you don’t make mistakes, you’re not working on hard enough problems.”*
>
> *—Frank Wilczek, 2004 winner of Nobel for Physics*

Intellectually we know that failures[1] are inevitable when we’re striving, growing, and learning. In practice we’re not always so understanding when it comes to our teams, our revenue targets, and especially when it comes to flogging ourselves.[2]

Indeed, not all errors signify progress. Some are negligence, or just bad luck. We don’t always learn from failure; in fact, sometimes there’s nothing in particular to learn.



credit[3]

“OK, lesson learned. Going forward no one listen to Don.”

The following “errors” are the natural by-product of good decisions, or the result of a fundamentally positive circumstance that is attended by the proverbial “good problem to have.” Most demand a response, but they should be regarded as a necessary side-effect of success, and celebrated as such.

Re-adding features/bugs you removed from the backlog

If you’re not adding back feature-requests or bugs you cleaned out of the backlog, you’re not cleaning out enough.

Backlogs grow without bound unless they are culled. 1000 tickets is the same as 100 tickets, except that you haven’t identified which 10% are most important. Which means you’re definitely not working on the most important ones. But if you delete things, it will sometimes turn out we needed to do it after all. That’s a sign that you’re handling your backlog well. (In part for this reason, you should have multiple backlogs,[4] except for the work you’re doing right now.)

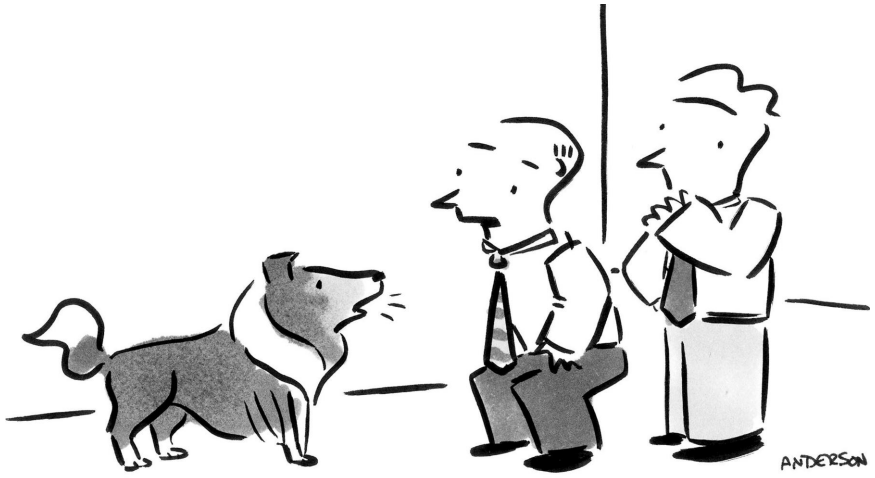
Pivoting a strategy just after creating it

A strategy that never changes is wrong, and the most likely time to discover that it's wrong is just after you wrote it down, because you have the least practical experience with how it intersects with real life.

If you're not pivoting a strategy that turned out to be wrong, you're penalizing the company with months or even years of useless work, followed by rework (if you haven't run the company into the ground), putting the entire future of the company at risk.

Great strategies[6] are hard to create, and released with great fanfare: Sparkling documents, inspiring presentations, pulpit-thumping speeches, reorganized teams and strategic-pillarized work. So the last thing you want, is come back in a month and say, "just kidding, we were wrong about something important."

Will you lose credibility? Will anyone believe the new strategy? Will people think "management doesn't know what it's doing?" These are risks you have to take, because executing the wrong strategy is far worse. Indeed, this is the expected result of a new strategy; it's highly unlikely you



credit[5]

"What's that boy?! A paradigm shift?!"

got everything right the first time. The best way to communicate, is to say everything in this paragraph out loud, so everyone knows that you know that they know, and that you're putting the company first, and ensuring that no one is doing work that we secretly know is the wrong work.

Refactoring infrastructure after growing 10x

If you're not refactoring your infrastructure after a tenfold increase in growth, you over-engineered your original infrastructure.

Having scaled WP Engine to millions of websites serving tens of billions of requests daily, I can tell you that scale is hard,[7] and that you don't know what will break under scale until you're already scaling (because you have to handle things that you can't even measure yet[8]). If you over-engineer your original product, you're simply not shipping your SLC fast enough,[9] and your naïve attempt at engineering massive scale from the start is just another form of premature optimization.[10]

Adding words because messaging was too terse

If you're not adding back words because the messaging was so terse that it became confusing, your marketing is too verbose, too fluffy, and probably doesn't know what it's trying to say.

People don't read. Tweets are short. Google Ads are shorter. Email titles are shorter. People bounce off home pages in three seconds. No one reads the paragraph of text in the dialog box. You're not even reading this paragraph.

It's 100x more likely that your messages aren't punchy enough, aren't specific enough,[12] than that they're so brief as to be unintelligible. Nowadays you can use ChatGPT prompts to get you 80% of the way there, so you have no excuse.

Adding back features you removed

If you're not adding back features you removed, you're not removing enough.

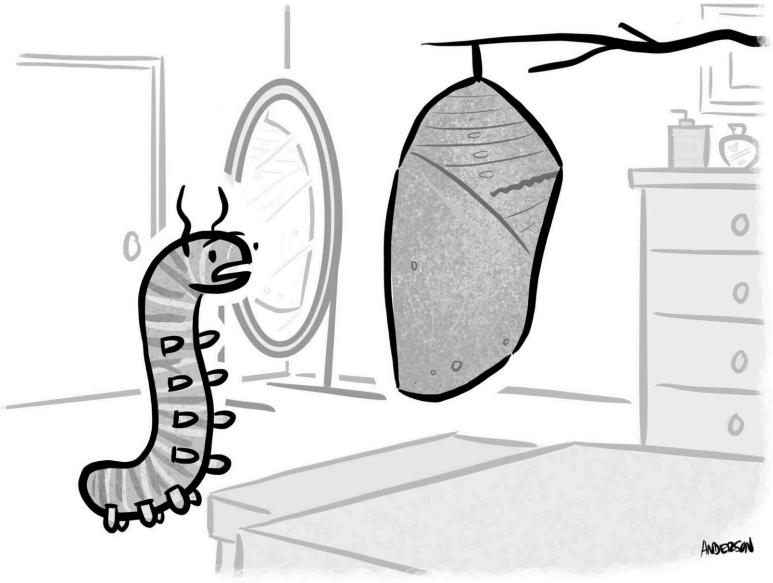
credit[11]

Many products never remove features. This indicates we're not being critical enough, not weeding our garden, not learning what customers really want,[13] not understanding what's useful, not admitting when we got it wrong, not shifting when the market shifts. When we do remove a feature, sometimes it will turn out the feature really was important after all. While of course in a perfect world we wouldn't have made that mistake, it's a natural consequence of weeding.

Fixing lots of bugs just after a major release

If you're not fixing bugs due to releasing quickly, you released too late.

While releasing garbage is a bad policy,[9] it's also bad to wait until "everything's perfect." Windows 95 shipped with tens of thousands of known bugs,* and was heralded as one of the greatest software innovations and most successful product releases of its time. Contact with customers lets you know which bugs are more important to fix next, and always reveal new bugs that you weren't going to find on your own anyway.



"Could you hurry it up, sweetheart? We're going to be late!"

credit[14]

Waiting too long to scale support or sales

If you held onto support and sales for too long, rather than hiring a team, you learned a lot about your customers, and a lot about how to do Support and Sales.

It's a classic funded-startup mistake to scale out either of these organizations too soon. Without a system in place, with materials, knowledge bases, and scripts, new hires don't know how to do the job. A distributed-work environment makes this 10x more challenging. The second you put someone else between you and a customer, your pace of learning and

* There were so many, there's an entire book[15] explaining how to work around 1000 of them.

understanding falls off a cliff. Wait until it's breaking for lack of scale,* then scale.

Letting someone go soon after hiring

If you held onto someone even though you knew isn't was never going to work, you're doing a great disservice to that person, and your team, and your company, and yourself.

Of course this shouldn't be done capriciously, but no one benefits from dragging it out. One likely outcome is that you lose good people, because they see you building a team they don't want to be a part of. Another is a deluge of meetings, complaints, side-conversations, and general worsening of morale. And lower productivity, as competent people cover for the incompetent. You have to face the truth[16] and act quickly. If this is happening a lot, it's also urgent that you fix your hiring process; in the meantime, the rule still applies.

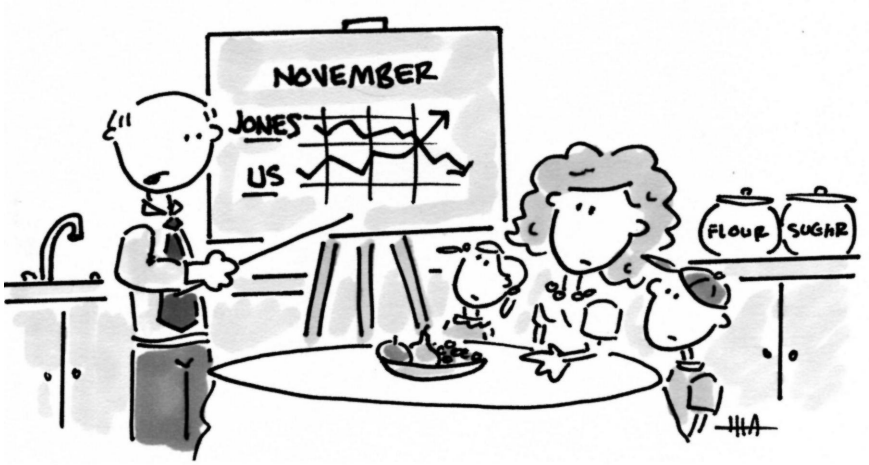
Ignoring a competitor's move that turned out to be important

If you're not ignoring most of your competitor's moves, then you're playing their game, not yours.

It's essential to stay focused on your unique value proposition and not get sidetracked by every move your competitors make. For instance, when a major competitor of Dropbox launched a similar service at a lower price, Dropbox chose to stay the course, focusing on their superior user experience and brand loyalty, rather than engaging in a price war.

Sometimes it will turn out that you really do need to react, but that signal will come from customers, in the form of them asking for features "because so-and-so has it" or cancelling and going to a specific competitor. That indeed demands a reaction, but only because you're seeking what's

* Once at scale,[7] this rule no longer applies; at that point, you're mismanaging the company



credit[17]

genuinely best for your customers, not because you're reacting to everything that competitors do.

Rejecting a lucrative, distracting deal

If you're not rejecting lucrative deals that don't align with your strategy, then you don't have a strategy.[6] If you're not rejecting relationships that don't align with your core values, you don't have core values.[18]

Money is too tempting to reject. Money is one of the main reasons you're building a company in the first place. Money is what keeps the doors from closing and enables the next set of things you want to do. It's even wise to say "yes" instead of "no,"[19] so long as there's enough money in it.

But money is not more important than strategy, and it cannot be more important than your values, otherwise you're saying that you don't actually have either one. There's always a way to make more money—a different product, different industry, or breaking the laws or being unethical. There's a reason why you're taking the path you're currently on.

Not all problems are indicative of poor decisions. It's easy to be hard on ourselves, but sometimes we should do just the opposite:

Celebrate our devotion to good decisions and good strategy, even when they have negative consequences.

That means you *have* a strategy, and have the ability to make the tough, wise decisions.

HT Hassy Veldstra[20] *for finding the Staedtler advertisement.*

Current version of this article:
https://longform.asmartbear.com/good-problems-to-have/

More articles & socials:
https://asmartbear.com

REFERENCES

1. https://longform.asmartbear.com/fail/
2. https://longform.asmartbear.com/impostor-syndrome/
3. https://andertoons.com/sales/cartoon/8669/lesson-learned-no-one-listen-to-don
4. https://longform.asmartbear.com/jit-backlogs/
5. https://andertoons.com/dog/cartoon/5038/whats-that-boy-paradigm-shift
6. https://longform.asmartbear.com/great-strategy/

7. https://longform.asmartbear.com/scale/
8. https://longform.asmartbear.com/scale-rare/
9. https://longform.asmartbear.com/slc/
10. https://news.ycombinator.com/item?id=23096321
11. https://twitter.com/hveldstra/status/1750900705634742423
12. https://longform.asmartbear.com/icp-ideal-customer-persona/
13. https://longform.asmartbear.com/customer-development/
14. https://andertoons.com/insect/cartoon/8816/chrysalis-late-butterfly
15. https://www.amazon.com/Windows-95-Bug-Collection-Work-Arounds/dp/0201489953
16. https://longform.asmartbear.com/failure-to-face-the-truth/
17. https://andertoons.com/family/cartoon/311/as-you-can-see-weve-not-only-kept-up-with-jones-but-surpassed-them-next-up-nelsons
18. https://longform.asmartbear.com/emptiness/
19. https://longform.asmartbear.com/say-yes/
20. https://twitter.com/hveldstra