The “Opposite Test”

I’m sick of generic feature/benefit bullet points. They’re too easy to make fun of. Here’s a sampling from a website that will remain nameless to protect the guilty:

- Easy to use
- Robust features
- Innovative systems
- Customer-first

Really, it’s easy to use? As opposed to what, difficult and temperamental? Robust, huh? Great, because from here it looks tenuously held together, the slightest breeze threatening to crumble its delicate construction, so it’s good to know that, actually, it’s robust. Oh I’m sorry, the *product* isn’t robust, the *features* are robust, whatever *that* means.

If you want to *not* stand out from the crowd, use statements that everyone uses. Would anyone claim to be non-innovative? Anyone claim that they put customers third?

credit[1]

So here’s my[*] Opposite Test: For each feature/benefit bullet point, construct its negative and see if that statement is ridiculous. Would anyone be able to construct a rational strategy with that negative? Perhaps a competitor already has! If the negative is indeed ridiculous, if it would be impossible to have a product or positioning or strategy that included the negative, it means this bullet point is trivial, obvious, mandatory, or at least undifferentiating from the competition.

It means it’s weak, it’s boring, and most importantly, it’s meaningless. And it’s taking up space (on the page and in your brain) that should be occupied by meaningful, powerful, differentiating things, upon which you’re basing your product strategy.

[*] Although I’m proud to have come up with this independently, this is a well-worn idea with many famous proponents, such as Roger L Martin (“Is the opposite stupid on its face?”[2]) and Al Ries and Jack Trout in *Positioning: The Battle for Your Mind* (“Anytime you come up with a positioning idea, test it out by asking yourself this question: If a competitor did the exact opposite, would it make sense? If it doesn’t, then you’re saying nothing.”) It’s also sometimes called the Reversibility Test. This convergent thinking only lends more credibility to the idea.

Let's apply the test. The negative of "Easy to use" is "Difficult to use." That would be a pretty funny statement! No one would ever claim that, so throw it out.

The negative of "Enables communication" is "Blocks communication." Crazy; no one would admit their tool does that.

The negative of "Stores files as big as 100 terabytes" is "Cannot handle huge files." Not ridiculous, in fact this is sadly true of many systems. It passes the test.

The negative of "Fully open source" is "Closed source." Of course that *is* the strategy used by most companies, so it passes the test.

"Fully backward-compatible, even after twenty years." This passes the test. Most software introduces breaking changes at some point, to enable new architecture or new features. In fact this is an important strategic decision. Backward-compatibility is important when you have millions of users with on-premise software where some components are 10 years old and no longer updated (like plugins in WordPress), and therefore compatibility is a feature. But it's bad in that it hamstrings designers on UX innovation, product managers on workflow and feature innovation, and engineers on architecture and performance innovation; sometimes a breaking change is required to stay relevant and modern.

Here's a good one from our own product[3] at Smart Bear: "Integrates with seven version control tools." Negative: "Not integrated with version control" or "Integrated with [one tool]." Not particularly funny; in fact each of these statements *are* true of all our competitors. And it can be a good strategy to be deep and feature-rich against a single API, rather than support a wide number of APIs, limited to features that are common to all. So this statement differentiates ourselves in a specific way, and the opposite is a valid—and actually-practiced—product strategy.

Another good way of understanding whether you're passing the Opposite Test is **whether there are negative consequences** from your statement, which you acknowledge. Continuing the examples above: The advantage of supporting multiple version control systems is compatibility, but the advantage of supporting only one is a deeper, more sophisticated workflow integration. And since a single team probably uses only one version control system, a single team might prefer the "one version control" system over the "multi." Thus, selecting "multi" anyway has a direct negative consequence, and therefore it is a real decision.

If you're using generic bullet points now, you'll find that replacing them isn't easy! You have to really think about what's strongest about your product, about how specifically it beats the alternatives, and how make it pithy. This is a useful exercise in itself.

One exception to the Opposite Test: You can use a generic if it's your single biggest differentiator, where you're truly 10x better than the competition along that dimension, and therefore you really "own" this concept as your identity and value.

A good example here is "Fastest." The negative is funny ("Slow operation means lots of time staring into stagnant progress bars"[4]). But if you make it your highest priority, it can work. Make your bi-line "The fastest ." Prove it with benchmarks. Explain how speed is not only about saving operator time (the obvious benefit) but how it enables entirely new features. For example, perhaps operation X is typically so slow that people can't take advantage of it. But since your system can complete operation X in two milliseconds, suddenly it becomes a feature. Even if a competitor technically has the feature, you make it practical.

All this is just another way of saying: Be specific,[5] avoid buzzwords,[6] be fully committed to your ideal customer[7] and what they value,[8] and tell the truth.

It's a critical component of having a great strategy,[9] and great positioning.

The current version of this article:

https://asmartbear.com/opposite-test/

More articles & socials:

https://asmartbear.com

References

1. https://andertoons.com/filing/cartoon/7897/trays-on-desk-actions-and-equal-opposite-reactions
2. https://medium.com/@rogermartin/is-the-opposite-of-your-choice-stupid-on-its-face-5b247ffd7f94
3. https://codecollab.com/
4. https://www.codinghorror.com/blog/archives/001058.html
5. https://longform.asmartbear.com/specificity/
6. https://longform.asmartbear.com/authentic-is-dead/
7. https://longform.asmartbear.com/icp-ideal-customer-persona/
8. https://longform.asmartbear.com/needs-stack/
9. https://longform.asmartbear.com/great-strategy/