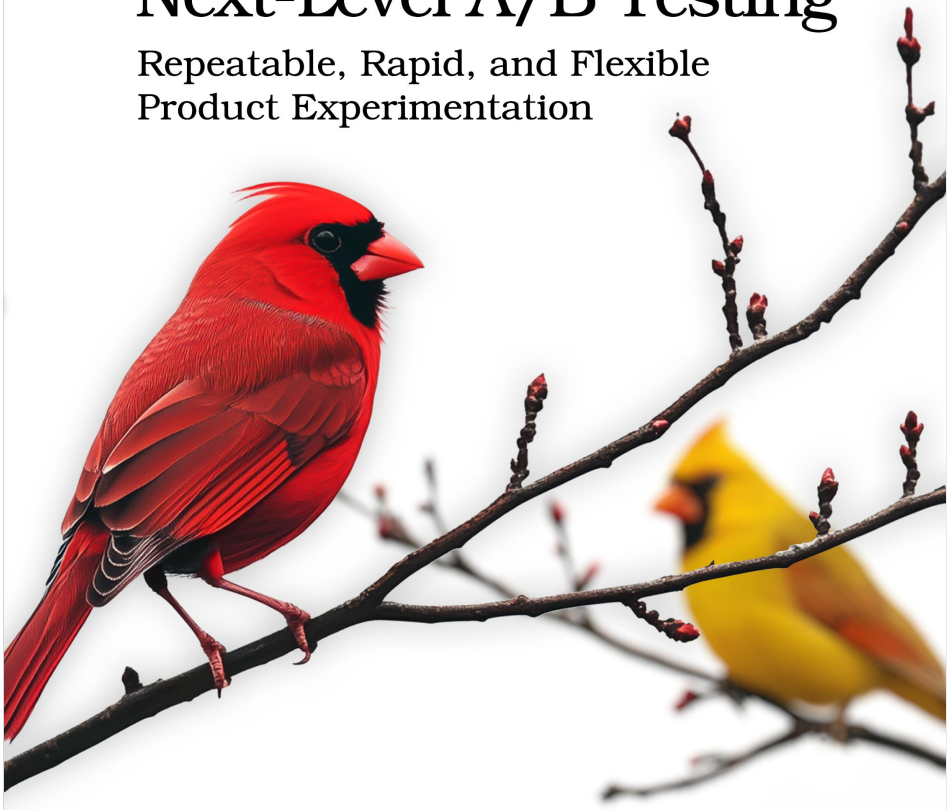
The Pragmatic Programmers

Next-Level A/B Testing

Repeatable, Rapid, and Flexible Product Experimentation



Leemay Nassery
edited by Vanya Wryter

This extract shows the online version of this title, and may contain features (such as hyperlinks and colors) that are not available in the print version.

For more information, or to purchase a paperback or ebook copy, please visit https://www.pragprog.com.

Aligning on Experiment Goal

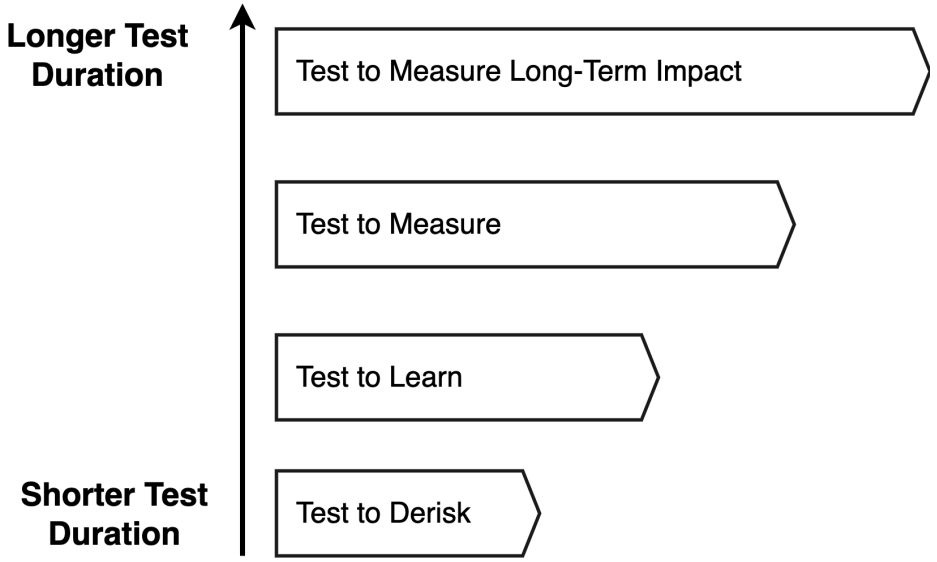
The number one rule of A/B testing is simple; ensure you can answer this question: What are you trying to learn from the experiment?

Sometimes, you aim to detect a positive effect in metrics by introducing a specific change to your users. Other evaluations may aim to ensure a change is non-inferior, no worse than the control given a predefined margin. If you're having difficulty defining the objective for evaluating a change in the scope of an experiment, consider categorizing your test into one of the following categories:

1. Test to derisk: The experiment's goal is to reduce the risk of degrading the user experience or introducing poor-performing system changes to all your users at once.
2. Test to learn: The experiment's goal is to learn more about the change or prototype to inform future product development and give you the confidence to continue investing beyond the initial prototype. You'll configure smaller sample ratios and feature-level metrics in your test design, decreasing the time it takes to gain initial insights.
3. Test to measure: The experiment's goal is to compute how a change impacts your user, product, and business metrics that are more sensitive. Since your objective is to understand the impact before potentially enabling the feature for all users, you can also refer to this category as "test to launch."
4. Test to measure long-term impact: The experiment's goal is to measure the effect of a feature or change on longer-term metrics such as retention, churn, and monthly active users.

Each test category has a distinct goal, so aligning on the objective is critical before configuring your experiment, as the goal does influence the test design.

The following image illustrates the hierarchy of test categories in relation to their general duration, a key factor of experiment design.



In the spirit of reducing testing duration, experiments with the goal of learning and derisking can be shorter, which makes them very attractive if the goal is to gain early insights. A test to learn should not have the same metrics as a test to measure, as your goal is to gain early insights into the idea's validity and not necessarily understand company-level impact. Keep in mind that after conducting a test to learn, you should run a test to measure the impact on key metrics with higher significance.

A typical example of a test to learn is when you're at the beginning stages of prototyping a new feature that requires early insights to gauge the product's initial instincts on the overall design. In this case, a practical next step is a shorter experiment to conduct ad-hoc analysis and understand user engagement at a smaller scale with less statistical significance.

As for the test to derisk category, there are also use cases where a non-user-facing change is A/B tested to understand latency and response time effects. When measuring engineering performance in the scope of an experiment, your goal is to derisk a change that could cause an incident or outage. By enabling the change for a subset of users, you'll gain more confidence before it's available for all users. A test to derisk can be shorter if the metrics aren't product-oriented, as your goal is to capture sufficient data to understand system-level metrics.

If the goal is to increase confidence but not necessarily measure the exact effect in a statistically significant way, then opting for a shorter-duration test

where the intent is to derisk or learn is a great option to increase your experimentation rate. Now that you can use the four testing categories, let's consider another solution that impacts testing design.