Delta Testing 101

Continuous User Testing in Agile Software Development



centercode

centercode

Table of Contents

The State of Modern Software Development 1

How Delta Testing Supports Modern Software Development 4

- Continuous Planning for Agile Teams 4
- Replenishing the Tester Pool 5
- Rewarding Testers with Subscription-based Incentives 7
- Engaging Users Between Updates with Passive Testing 7
- Maintaining Mountains of Feedback with Aging 8
- Closing the Feedback Loop with Customer Verification 9
- Measuring Product KPIs with Cadenced Surveys 9
- Ongoing Comparison for Extended ROI 10

Delta Testing Solutions 12

Is this resource for you?

You are...



An engineer



A product manager



A quality professional

You might be...

- Concerned with maintaining release quality while keeping pace with agile development and continuous delivery
- Struggling to apply traditional beta testing processes to your continuous delivery practices

centercode

THE STATE OF MODERN SOFTWARE DEVELOPMENT

The evolution of customer expectations has changed the software development industry. In the last fifteen years, consumer interest in cutting-edge technology has skyrocketed. And right alongside it, concerns over reliability, interoperability, automation, user experience, and frequency of release have become the norm for dev teams in every vertical.

It's not just about giving your customers what they want - it's about getting it to them faster than your competitors. This focus on drumming up deliverables quickly has given rise to speed of delivery-focused strategies like agile development, automated testing, continuous integration and continuous delivery (CI/CD).



48% Agile

44% Hybrid (Waterfall + Agile)

2% Waterfall

6% Other

92%

of tech firms are using agile or an agile-hybrid

Ironically, as companies embrace these strategies to improve the customer experience more quickly, software development teams find themselves sacrificing quality for speed. And while it's widely understood that customer testing increases quality, many organizations are instead barrelling forward, praying nothing big fell through the cracks.

Why are 7 out of 10 professionals using beta testing to collect customer feedback before launch, but only 40% performing regular user testing after launch?

centercode

The reason for this disparity is two-fold. First, they're seeing that standard beta processes aren't compatible with agile development and continuous delivery. Second, they say they're stretched too thin.

1 IN 3 professionals says that there isn't enough time

2 IN 5 professionals say that there aren't enough resources

Delta Testing: Aligning Beta Practices with Continuous Delivery

As industry leaders seek a compromise between the need for ongoing customer feedback - which increases product quality - and the demanding pace of continuous delivery, more and more are adopting the process of delta testing.

Delta testing runs alongside agile development, using customers to provide feedback as dev teams continue to build, test, and iterate on new releases. As the evolution of traditional beta testing practices, it supports product development with customer input, offering real-world, real-time feedback and use cases. But unlike beta, delta testing features a distinct set of continuous processes that facilitate post-launch user testing for every new build.

What is delta testing?

In mathematics, "delta" refers to the difference between two points. Delta testing uses target market customers to test and provide feedback on the specific features of your product that have changed since the last build. Feedback collected from delta testing provides validation of acceptance between releases or builds.

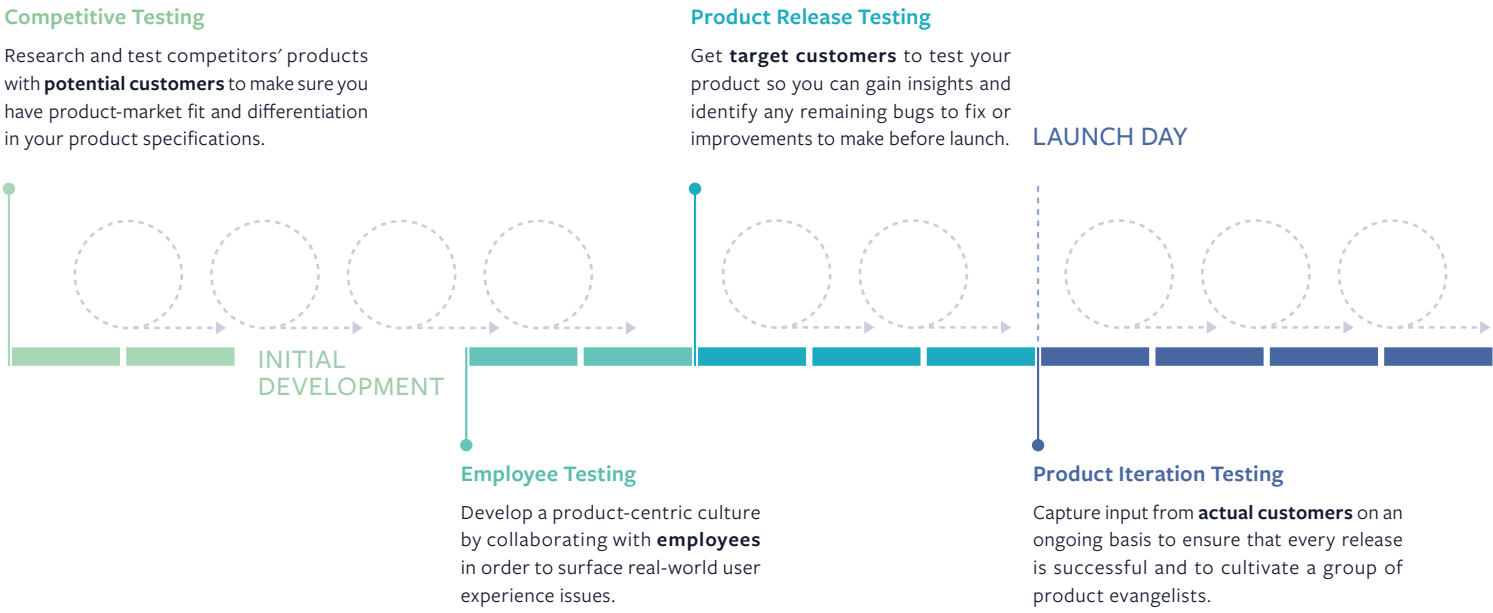
How is a delta test structured?

Delta tests collect issues, ideas, and praise from real customers in live environments using feature-complete products over an extended period of time. This feedback is ultimately prioritized and displayed as actionable recommendations to fix, improve, and promote ("FIPs") those products based on insights from testers and test objectives.

But unlike the time-boxed structure of beta testing, delta testing allows for ongoing engagement through a use case called product iteration testing. This ongoing form of delta testing is deployed in sprints that are aligned with the organization's existing agile development and continuous delivery cycles. As dev teams build and iterate on software, testers provide immediate feedback on those features and fixes before release.

When is delta testing used?

Companies use delta testing throughout the entire product development life cycle. It replaces alpha testing, dogfooding, beta testing, UAT, field testing, and continuous user testing. There are many different use cases for delta testing, and each one is focused on getting feedback from distinct, targeted audiences early and often. Here are the four most common use cases:



Who manages delta testing?

Engineering/development, product management, quality assurance, and program management all have a stake in and may contribute to delta testing results, since they are directly involved in defining, building, and refining the product from one release to the next.

Why do companies need delta testing?

Many software development teams struggle to balance the speed of continuous delivery with the quality of their releases. Delta testing helps them bridge that gap to avoid negative outcomes like higher support costs and lower customer satisfaction.

centercode

Bringing customers into the post-launch product development life cycle comes with a whole host of benefits, including:

- More effective roadmap and release prioritization
- Reduced support costs
- Higher customer retention and customer satisfaction
- Faster testing cycles, leading to faster development

HOW DELTA TESTING SUPPORTS MODERN SOFTWARE DEVELOPMENT

These are the specific processes and practices that differentiate delta testing from traditional beta processes and help software testing teams validate software updates on a continuous basis.

Let's take a look at the following core concepts of delta testing:

- Continuous Planning
- Tester Replenishment
- Subscription-based Incentives
- Passive Testing Phases
- Feedback Aging and Backlog Grooming
- Closing the Feedback Loop
- Cadanced Product KPI Surveys
- Ongoing Release and Process Comparison

CONTINUOUS PLANNING FOR AGILE TEAMS

Agile development prioritizes flexibility, and agile teams are changing direction with the changing needs of their products. This means they're continuously reprioritizing their roadmap, grooming backlogs, and planning sprints closer to the release date.

To align with these practices, delta testing leverages continuous planning. Unlike the holistic feature mapping strategy of beta tests, continuous planning puts a spotlight on specific topics to keep tester focus trained on priority

product areas. This allows software testing teams to set tightly scoped, impactful test objectives on short notice, without having to map out the entire test beforehand.

Product Iteration Delta Tests	Beta Tests
Use continuous planning to scope test objectives as they are defined for sprints or new releases.	Use initial planning to scope test objectives in their entirety.
Focus testers on a specific set of topics. Testers cannot see the full map of features and experiences.	Guide testers through a map of topics that covers large product areas, features, or experiences that require validation.
Define objectives in response to upcoming sprints and new releases; tighter scope allows for greater flexibility.	Deprioritize new objectives that pop up in the course of the test; the structured approach minimizes scope creep.

REPLENISHING THE TESTER POOL

Because product iteration delta testing is an ongoing program, an active tester pool is critical to a quick turnaround for feedback. Maintaining the population of that pool - and keeping those testers engaged sprint after sprint - is therefore equally critical to its execution.

It can be difficult to retain qualified users without a proactive strategy in place. This is where tester replenishment and engagement monitoring come in. These practices combat the inherent challenge of keeping testers engaged for long periods of time to ensure your tester pool can leap into action at a moment's notice.

Here are a few of the moving parts:

Tester Team Minimum

The minimum number of users you need in your tester pool to accurately validate your product updates. The admin team determines this number during planning to help identify when there's a risk of undershooting the ideal number of testers. The minimum threshold is the trigger point for tester replenishment.

centercode

Tester Opt Out

Even dedicated testers have lives outside of your project and might need to take a break from testing. Testers who actively opt out (instead of lapsing into inactivity without notice) help you proactively manage the tester population. Reward their initiative by welcoming those testers back into your project if they want to continue testing at a later date.

Engagement Monitoring

The process of monitoring tester participation to identify when testers have become inactive. After a predetermined period of inactivity - for example, failing to log in for two weeks - you can either place the user on probation or remove them from the project immediately. Unlike testers who opt out, users who become "inactive" are not invited back to participate.

Tester Replenishment

The practice of replenishing the tester pool when the population dips close to the minimum threshold. By recruiting and onboarding a reserve of targeted users, you can replace testers who opt-out or go dormant without putting your results at risk.

Product Iteration Delta Tests	Beta Tests
Replenish testers to the project pool on a recurring basis as participants opt out or become inactive.	Add surplus testers to the tester minimum to ensure the project meets participation goals. Testers who drop out or become inactive are not replaced.
Allow testers to opt out of the project at any time. Testers who take the initiative to opt out can return at any time.	Projects have a defined endpoint. Testers who choose to opt out cannot rejoin the project mid-test.
Remove inactive users from the project after a period of inactivity to maintain high ongoing participation.	Remind testers to participate on a regular basis but does not remove inactive users mid-project.

centercode

REWARDING TESTERS WITH SUBSCRIPTION-BASED INCENTIVES

While rewards should not be your testers' sole motivation, rewarding them with incentives is important. It shows that you appreciate the time and energy they're putting toward your product. The most common incentives during beta testing are gift cards and products. But recurring options that are high value, reasonably priced, and easy to control are better suited to the continuous nature of product iteration delta tests.

Subscription-based rewards that compliment the product your testers are working with fall neatly within these parameters. For example, testers in a smart speaker project might receive a streaming audio subscription for their continued participation. For enterprise software or B2B products, you could offer integration subscriptions.

If you don't want to incentivize your testers with a subscription, you can also send them swag or gift codes on a regular basis. There are countless consumer and business-facing services that appeal to your testers.

Product Iteration Delta Tests	Beta Tests
Reward testers continuously during the project period.	Distribute rewards to active testers at the end of a test period.
Offer a low-cost subscription that compliments the tested product and/or has a high value to the tester.	Offer a single incentive, such as a gift card, functional test unit, or a product that compliments the nature of the test.
Use subscriptions that are easy to control or maintain in case testers opt out or become inactive.	Reward testers based on specific participation milestones, like the number of completed surveys.

ENGAGING USERS BETWEEN UPDATES WITH PASSIVE TESTING

While product updates often occur on a regular cadence, they may not always require active attention from testers. In the event of delays, pauses, or minor updates, you can call for a period of passive testing.

Passive testing is a phase of natural product usage where testers can still submit feedback, but do not need to focus on a specific product area. It is best practice to both tell your testers when a passive phase is happening and to modify your expectations for tester engagement accordingly.

Common uses for passive testing are:

- Regression testing your new build
- Verifying resolved issues with testers
- Assessing general product usage

Product Iteration Delta Tests	Beta Tests
Commonly cycle into a passive testing phase as a time buffer or soaking period to keep testers engaged between builds.	Use active testing cycles, unless a delay or the need for additional feedback makes passive testing necessary.
Anticipate periods of passive testing and set that expectation with testers.	Go into a passive testing phase during a delay or extension and communicate with testers accordingly.

MAINTAINING MOUNTAINS OF FEEDBACK WITH AGING

Delta tests generate a goldmine of data for software development teams and their stakeholders. But when it's sitting disorganized inside a backlog or ticketing system, it quickly becomes more overwhelming than genuinely useful. By nature, the issues, ideas, and praise generated by continuous testing will add up over time. The longer backlog items go unresolved, the harder it is to prioritize feedback - and the more certain it is that valuable insights will slip through the cracks.

To regulate the natural tendency for backlog build up, product iteration delta tests leverage a practice called *aging*. This is the process of archiving stagnant feedback after a predetermined time frame so that actionable, priority feedback is easier to find.

While your time frame for determining when feedback is inactive will be specific to your program, you may want to consider archiving it if:

- The item has been resolved
- Your testers and stakeholders aren't talking about the issue anymore
- The feedback is not relevant to your current roadmap

centercode

Product Iteration Delta Tests	Beta Tests
Archive feedback after an extended period of inactivity.	Only archive feedback when a team member closes it.
Feedback becomes outdated as teams iterate and build new versions over time.	Feedback becomes outdated, but it's less likely to happen before launch.

CLOSING THE FEEDBACK LOOP WITH CUSTOMER VERIFICATION

Live technical environments and usage in the wild offer test cases that in-the-lab testing just can't simulate. In addition to surfacing real-world issues, the ongoing nature of product iteration delta tests offer a unique opportunity to verify those fixes with testers. This is called *customer verification*, or "closing the feedback loop." Closing the feedback loop mitigates the risk of issues going untested until they're in the software of thousands of users.

Closing the feedback loop with customer verification:

- Ensures a positive user experience for your testers
- Make sure issues are resolved in unique enviroments
- Saves development time if the issue is a corner case and difficult to reproduce

Product Iteration Delta Tests	Beta Tests
Feedback is surfaced by a testers, addressed by dev and/or QA through new updates, then verified again with the testers.	Feedback is surfaced by testers and assessed by dev and/or QA teams. (Due to schedule constraints, testers often don't get to see or test the updates that result from their feedback.)

MEASURING PRODUCT KPIS WITH CADENCED SURVEYS

Capturing product metrics and key performance indicators (KPIs) is essential to understanding the full output of your investments in delta testing. In addition to routine quantitative data - such as the number of issues surfaced, or

comparisons between releases - you can leverage tester sentiments and opinions to paint a holistic picture of your product's progress.

Unlike with other quantitative data points, trying to capture metrics that cover more qualitative topics - like customer attitudes - can be difficult during continuous testing. An intense focus on one product area during a single delta testing cycle will color your testers' opinions of the whole product. It's the classic "you can't see the forest for the trees" situation.

Stretching out the length of time between customer satisfaction surveys mitigates bias and gives your users enough distance from their recent experiences to more accurately assess their entire experience. By deploying surveys on a quarterly, bi-annual, or annual basis, you can capture and monitor conclusive product KPIs throughout the year.

Product Iteration Delta Tests	Beta Tests
Gather product KPIs like NPS and mock reviews on a quarterly, bi-annual, or annual basis.	Gather product KPIs in a final, comprehensive survey at the end of the testing period.

ONGOING COMPARISON FOR EXTENDED ROI

Since "delta" is a measure of the difference between two points, it offers a unique opportunity to make ongoing comparisons from project to project and build to build. These comparisons allow you to demonstrate the ROI of your delta testing efforts and experiment with your processes for continuous improvement.

Monitoring the impact of implemented feedback as the product evolves showcases the value of your program to stakeholders. You can use the product KPIs you collect over time to highlight the impact of your delta testing programs in a quantifiable way.

In addition, the rolling format of product iteration delta tests support ongoing experimentation. Comparing metrics - for example, the number of feedback submissions and level of project engagement - shows where you were most successful and what you can improve. This enables you to refine your processes from sprint to sprint.

centercode

Commonly used metrics to compare releases are:

- Delta Health Score
- Delta Success Score
- Delta Impact Score
- Number of issues, ideas, and praise
- Satisfaction scores (gathered on a bi-annual or quarterly basis)
- Gap items not found in QA
- Popularity of an issue

Product Iteration Delta Tests	Beta Tests
Generate a continuous flow of data that can be compared over time and across iterations.	Generate a snapshot of pre-launch data, which is rarely compared across iterations due to constraints within the product schedule.
Compare metrics between software builds and updates.	Capture metrics, but does not usually compare metrics because significant iterations before launch are rare.
Support rapid, ongoing improvement of testing processes and practices.	Take longer to iterate on processes because projects are not continuous and accumulate less directly comparable data.

centercode

Delta Testing Solutions

Leading product ops teams use Centercode to identify real-world issues to fix, areas to improve, and what their users love in order to build amazing products.



Centercode's delta testing platform is built on decades of experience user testing the world's most successful products. Our platform equips you with the automation, best practices, and processes you need to get actionable user insights for any use case.

PRODUCT LAUNCH

Engage your **Target Market** to successfully bring new products and services to market

USE CASES:

- Beta Tests
- User Acceptance Tests
- Field Tests

PRODUCT GROWTH

Recruit **Existing Customers** to continuously mature live products and services

USE CASES:

- Progressive Delivery
- Early Access
- Customer Advisory Boards

CUSTOMER ZERO

Gather **Employee** feedback to enhance your products while fostering a product culture

USE CASES:

- Alpha Tests
- Dogfooding
- Employee Tests

COMPETITIVE EDGE

Enlist **Competitors' Customers** to rapidly gain actionable intel about their products

USE CASES:

- Competitive Intelligence
- Market Testing
- Comparative Research

DELTA TESTING PLATFORM

Get a live demo of Centercode to experience the easy way to run your user tests, including:

- Auto-generated test designs
- Ted, your tester engagement bot
- Smart feedback prioritization
- Instant KPIs and dashboards

SCHEDULE A DEMO

WHY CENTERCODE?

Increase Revenue
½ ☆
INCREASE
in product ratings

Release Sooner
50%
FASTER
test cycles

Gain Participation
75%
INCREASE
in tester engagement

Save Time
40% +
DECREASE
in time managing tests