Editor: **Ciera Jaspan**
Google
ciera@google.com

Editor: **Collin Green**
Google
colling@google.com

# Measuring Developer Experience With a Longitudinal Survey

Sarah D'Angelo, Jessica Lin, Jill Dicker, Carolyn Egelman, Maggie Hodges, Collin Green, and Ciera Jaspan

**MEASURING DEVELOPER EXPERIENCE** should always start with the developers themselves. At Google, one of the ways we do this is through surveys. We've been running a quarterly large-scale survey with developers since 2018. We call it the *Engineering Satisfaction Survey*—or *EngSat* for short. It has been one of our most robust data sources and has served as a foundation for many of the topics we have covered in this series.

## Introduction

In this article, we will discuss how we run EngSat, some of our key learnings over the past six years, and how we've evolved our approach to meet new needs and challenges. But first, we should discuss why you might run a survey about developer experience in the first place.

## Why Run a Survey?

Surveys are inherently human centered; we are directly asking developers what they perceive, think, and feel. Surveys are also flexible and fast; launching a survey and changing questions is easier than creating a new logs-based metric. Additionally, surveys present an opportunity to get data on experiences that are not possible to measure objectively (for example, satisfaction) and to ask open-ended questions to gather more insights into why developers feel the way they do. Across the industry, we see companies running developer experience surveys to better understand their developers.[1,2,3] Because surveys can be run with small or large teams and require less instrumentation and infrastructure than logs-based measurement, they are a good place to start for organizations that are just beginning to investigate developer productivity. Surveys can serve as a solid initial foundation on which to build such an effort. At Google, we use our survey to complement our logs-based data sources and tell a complete story about what developers are doing and how they feel. Our ability to tell rich stories has expanded over the course of our team's tenure, and our EngSat survey has been around since the very beginning.

## Running EngSat

### How It Started

When we first sought to launch EngSat, our goal was to understand the needs of software developers at Google at scale and complement our existing logs data.[4] To differentiate ourselves from other employee surveys at Google, we focused on experiences that are unique to developers. We asked about our internal developer tool offerings and developer-specific tasks. We also emphasized topics that are typically difficult to measure (for example, technical debt, flow, or code quality) or not reflected in our logs data (for example, satisfaction or hindrances to productivity), enabling us to get some measurements of these difficult concepts and as a basis to conduct further research (for example, topics featured in previous articles in this column).

To ensure a relevant survey that would be generally useful in a sustained survey program, we partnered with subject-matter experts to develop and test the questions to ensure that they were representative of

*Digital Object Identifier 10.1109/MS.2024.3386027*
*Date of current version: 12 June 2024*

developer workflows. We also leaned on established best practices for survey design.[5] Having an interdisciplinary team of researchers and engineers allowed us to develop an effective instrument for measuring developer satisfaction and productivity.

However, we still faced resistance; surveys often face the same criticisms regardless of quality, scale, or domain. For example, critics contend that surveys are subjective (for example, susceptible to mistakes and misinterpretations), biased (for example, respondents may have an incentive to respond inaccurately), and more reflective of how people feel than what is actually happening. These criticisms are especially likely to emerge when survey results suggest something negative or counterintuitive. We worked to challenge these perceptions of survey research by pairing our results with logs-based measures and using the combined picture to reveal the value of how engineers feel at work and to corroborate the survey data with additional measurements. We also committed to running the survey at a regular and predictable cadence with high consistency in the survey's substance, which allowed us time to demonstrate the value of longitudinal data and reinforce the utility of the program.

## Key Features of the Survey Program

We've run EngSat on the last two weeks of each quarter for the past six+ years (25 quarters), building, refining, and evolving the survey over time. Running EngSat regularly for this long has been possible because of our systematic approach to the staffing, process, and infrastructure underlying the program.

Consistent and adequate staffing is the first key to the program. We've established a rotation so that one user experience (UX) researcher and one engineer jointly own the program each quarter. The UX researcher primarily drives the management and evolution of the survey instrument and program operations and communications. The engineer primarily drives the management of our automated data processing and analysis infrastructure (more later). Adequate staffing is important, but perhaps surprisingly, the rotation of staffing has also aided the program's success. Because of the rotation, we've had an incentive to create repeatable and well-documented processes. The institutional knowledge does not stay within one person. This allows the program to be run by anyone on our team at any time. The rotation is also a useful lever in stakeholder management; our staff can better support each other in difficult conversations, there is some peer pressure (team culture) to maintain consistency with established practice, and stakeholders come to recognize the way that the program itself works (rather than attributing challenging issues to individuals).

The second key to the program is the process. We have established a process for preparing, running, and analyzing the survey that we follow every quarter (see Figure 1). Each of the phases in the lower part of Figure 1 is associated with reference documentation, examples from prior runs of the survey, templates, and—where appropriate—analysis code and automation. Put more simply: we automate things where we are able, we provide templates and examples where we can't automate, and (regardless) we write everything

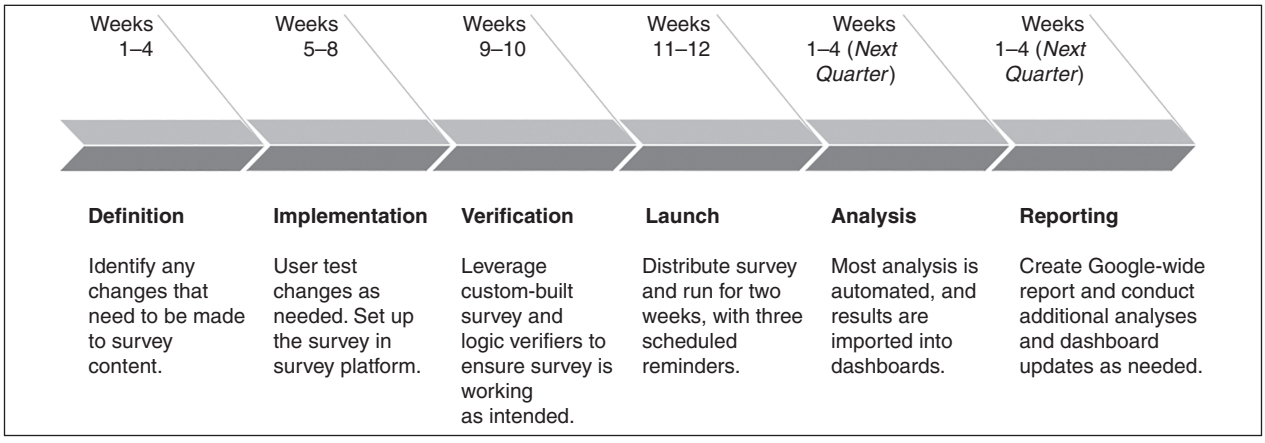



**FIGURE 1.** The end-to-end process for running the EngSat survey.

down. The process (and its associated artifacts) make the program more efficient to execute, more consistent by design, more portable across staff, and helps us accrue improvements over time.

The third key part of running this program efficiently is the infrastructure and automation supporting EngSat. We've developed a common infrastructure that enables us (and other survey programs at Google) to clean and aggregate the survey data so they can be made available in dashboards and reports. There are a few critical parts to this. First, we found that it was important to define a consistent approach to survey data formatting. We use a single storage format for common question types, so we can easily combine survey data from multiple surveys and adapt to changing survey tools. Second, we have established a few standard scripts that help us prepare, pull, and analyze the data. We use a survey verifier script that ensures that the survey we deploy is compatible with our automation and checks for differences from prior surveys, highlighting any changes made that can be double-checked. We then have scripts that pull response data from our survey tool, convert it to our survey data format, and store it in tables for easy analysis. Lastly, we have a set of aggregation and analysis scripts that automatically run after the survey closes. These scripts support our reporting of results by automatically calculating confidence intervals and running a nonresponse bias analysis.

## How It's Going

The ability to track the sentiment of Google developers over multiple years has given us unique insight into change over time. A few examples include 1) how we were able to understand the impact of a global pandemic, 2) how we leverage EngSat to validate logs-based metrics, and 3) how surfacing technical debt as a key pain point helped drive change.

### Measuring Developer Experience Before, During, and After a Global Pandemic

EngSat was starting its third full year of data collection when the COVID-19 pandemic began, and the onset of rapid and unpredictable change in work during the pandemic revealed the power of a stable survey instrument. Having years of consistently collected data as a baseline allowed us to see a change in key measures before, during, and after developers were working from home due to COVID-19. As we've discussed in this column previously,[6] we saw a notable decrease in self-reported productivity between Q1 2020 and Q2 2020 (as many people started working from home for the first time). We also added open-ended questions to EngSat that enabled us to gather insights about how developers were adapting to working from home and what we could do to help (capitalizing on the agility of a survey relative to logs-based metrics). Insights from the new items allowed us to motivate improvements to remote access and connectivity, key pain points for developers. Over time, we saw self-reported productivity rebound and even exceed prepandemic levels. Having a consistent survey program in place allowed us to understand the impact of a global event, drive change, and measure improvements.

### Validating Other Metrics

Another critical use of EngSat data is in validating the development of our logs-based metrics. When we identify a behavior we want to try to measure with our logs, we use our survey questions to help. First, we can identify engineers who have varying experiences with the behavior to formulate a deeper understanding. For example, when we wanted to create a measure to measure when developers experience flow or focused work,[7] we started by interviewing developers who reported different frequencies of achieving flow or focused work on a recent EngSat survey. Then, after we established a logs-based approach to measuring flow or focused work, we validated it with EngSat data to ensure that our metric was representative of self-reported experiences. Now, when we report our quarterly

The ability to track the sentiment of Google developers over multiple years has given us unique insight into change over time.

EngSat results on self-reported flow or focus, we can pair that with our logs-based metric to confirm that developers' subjective experiences of flow continue to align with focused behavior observed from the logs and to better understand what is driving changes in developers' experiences at Google. For example, we can analyze what other factors measured on the survey or via our other logs-based metrics predict a higher incidence of flow or focused work. We have applied similar techniques for measuring developer friction as well.[8] While we often collect behavioral data by other methods for metrics validation projects,[4] this use case highlights the benefits of pairing survey data with logs-based measurements to tell a holistic story about the developer experience.

### Reducing Technical Debt

EngSat has also been used to drive multiyear Google-wide initiatives that hinge on visibility, motivation, and finding common ground in data. For example, in 2019, we highlighted technical debt as a top hindrance to productivity, which motivated various efforts across Google aimed at reducing technical debt. As discussed in our article on technical debt,[9] we partnered with internal teams to develop best practices, management plans, and initiatives aimed at addressing technical debt. Since the dissemination of these resources company wide, paired with signal boosting from various internal teams and through leadership, we've seen large improvements in the technical debt sentiment on EngSat. The ability to identify an opportunity for improvement, motivate change, and measure the impact of various initiatives over time is a key value of the EngSat survey.

## The Evolution of EngSat

Since 2018, EngSat has served as a foundation for our team's research and has had a significant impact. However, as with any long-running program, we have faced challenges, and the survey program has evolved over time in response. The two primary problems likely to affect any survey program (including EngSat) are 1) increasing survey length and 2) decreased response rates.

It should come as no surprise that it is easier to add questions than it is to take them away. As new workflows emerge, there is a desire for new questions to meet those needs (for example, the influx of AI in developer tools). Simultaneously, stakeholders depend on existing questions. Addressing the tension between adding new questions, keeping consistent questions, and decreasing survey length is not easy. With EngSat, we have identified a subset of "evergreen" questions that we have committed to keep consistent. These are high-level questions around productivity, satisfaction, velocity, quality, and flow that we see value in consistently measuring. Outside of those questions, we are more flexible. During EngSat's tenure, we have run two major streamlining initiatives aimed at reducing duplicative or obsolete questions. These efforts are intended to both maintain the health of the survey (for example, reducing redundancy and length) and increase response rates by reducing the time it takes to complete the survey. While there is no silver bullet to address decreasing response rates, reducing the time it takes to complete the survey and demonstrating value to the survey taker are two strategies that we focus on.

Ensuring the long-term success of the EngSat program is—of course—dependent on getting a healthy response from the surveyed population. Engineers have to spend their valuable time to complete the survey and provide thoughtful responses for us to deliver useful insights. We have two approaches to fostering this healthy response: one related to sampling and one related to reporting. To effectively sample developers, we split the developer population into three random cohorts and only sequentially survey one group each quarter. Grouping into three cohorts instead of four means that individual cohorts are surveyed in different quarters of each year. Sampling in this manner reduces the sample size but enables us to bother engineers less frequently and still study the cohorts over time. Google has a large enough engineering population that splitting the sample in this way still yields a response count large enough to enable statistical power to detect significant differences over time. Ensuring that these samples are random and representative subsets of our overall engineer population means we can still run panel analyses knowing that there are no confounding differences across cohorts. As a complement, we emphasize transparency and accountability in our reports. We widely distribute a summary report and make the de-identified and aggregated EngSat results available to everyone at Google. Anyone at Google is able to see what is being measured and how we are acting on their feedback. Reporting and sharing aggregated data and impact in this manner encourage developers to take our survey year after year.

Even with constant efforts to keep the survey relevant and streamlined, after six years of a relatively stable instrument, we determined

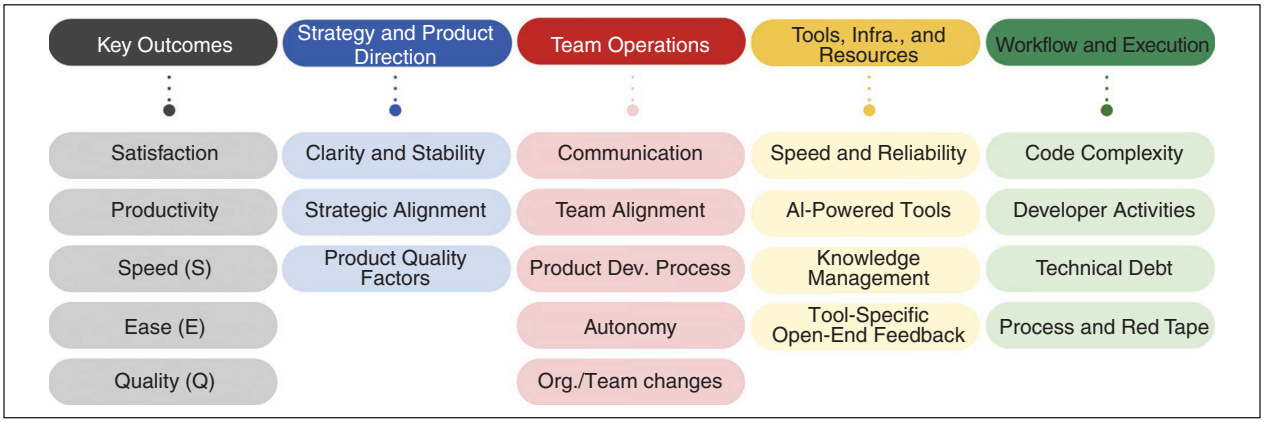



**FIGURE 2.** EngSat outcomes and themes. Org.: organization. Dev.: development; Infra.: infrastructure.

that it was time for a more substantial change. In our recent redesign of the EngSat survey, we focused on ensuring that the results were actionable and scalable. This meant removing our more specific questions that were targeted at smaller developer populations or measuring sentiment toward individual tools so that we could focus on more general questions that are relevant to all developers at Google. This was not an easy decision; there is a lot of value in more focused questions. They allow you to target specific needs and address those with smaller groups, but it can be harder to detect meaningful change due to smaller response rates in particular subgroups and the speed at which changes occur (for example, tools are not changing every quarter). To better serve decisions at the company level, we opted for more generalizable questions focused on five outcome measures and four additional theme areas that are drivers of our outcomes (see Figure 2).

Our experience running EngSat over the years has revealed the importance of understanding developer sentiment and being accountable for the results. We have built a robust developer experience survey program that we believe can be generalized to other organizations. To summarize the key recommendations we've discussed in this article

- Establish a clear and unique goal for your survey (ensure that there are no other survey programs or data sources that you can already use).
- Collaborate with domain experts and researchers to develop an effective instrument.
- Gather stakeholder buy-in (for example, communicate the value of survey data and partner with teams that can act on your results).
- If you are planning to run a longitudinal survey, invest time in the beginning in setting up the right infrastructure, process, templates, and documentation.
- Maintain the health of your survey program (control survey length and sample strategically).
- Be transparent and accountable. The quality of your insights depends on the feedback provided by your developers, so make it clear why they should spend their time on it.

Leveraging surveys to understand developer productivity is a great foundation, enabling you to gather insights quickly, relative to other methods. Maintaining a longitudinal survey effort requires more planning and structure. In this article, we have shared our processes for starting the EngSat program at Google, examples of what the results have enabled over the years, and how we have evolved our program as Google's needs have changed over time. We hope that these insights will help teams and researchers create effective developer experience surveys and expand on our approaches to grow the field.

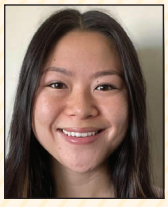
## References

1. A. Noda. "Inside look: Shopify's developer experience survey." DX. Accessed: Mar. 27, 2024. [Online]. Available: https://getdx.com/blog/shopify-developer-experience-survey/
2. B. Perry. "How LinkedIn analyzes developer survey data." DX. Accessed: Mar. 27, 2024. [Online]. Available: https://getdx.com/blog/analyzing-developer-survey-data-linkedin/

## ABOUT THE AUTHORS


**SARAH D'ANGELO** is a user experience researcher on the Engineering Productivity Research team at Google in Canterbury 7999, New Zealand. Contact her at sdangelo@google.com.

**MAGGIE HODGES** is a user experience researcher on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact her at hodgesm@google.com.



**JESSICA LIN** is a user experience researcher on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact her at jsclin@google.com.

**COLLIN GREEN** is a user experience researcher on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact him at colling@google.com.

**JILL DICKER** is a software engineer on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact her at jdicker@google.com

**CIERA JASPAN** is a software engineer on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact her at ciera@google.com.

**CAROLYN EGELMAN** is a quantitative user experience researcher on the Engineering Productivity Research team at Google in Sunnyvale, CA 94089 USA. Contact her at cegelman@google.com.



3. B. Perry. "Inside Peloton's developer experience survey." DX. Accessed: Mar. 27, 2024. [Online]. Available: https://getdx.com/blog/peloton-developer-experience-survey/
4. C. Jaspan et al., "Enabling the study of software development behavior with cross-tool logs," *IEEE Softw.*, vol. 37, no. 6, pp. 44–51, Nov./Dec. 2020, doi: 10.1109/MS.2020.3014573.
5. D. De Vaus and D. de Vaus, *Surveys in Social Research*. Evanston, IL, USA: Routledge, 2013.
6. C. Jaspan and C. Green, "Developer productivity for humans, part 2: Hybrid productivity," *IEEE Softw.*, vol. 40, no. 2, pp. 13–18, Mar./Apr. 2023, doi: 10.1109/MS.2022.3229418.
7. A. Brown, S. D'Angelo, B. Holtz, C. Jaspan, and C. Green, "Using logs data to identify when software engineers experience flow or focused work," in *Proc. CHI Conf. Human Factors Comput. Syst.*, Apr. 2023, pp. 1–12, doi: 10.1145/3544548.3581562.
8. A. Brown, A. Chang, B. Holtz, and S. D'Angelo, "Developer productivity for humans, part 6: Measuring flow, focus, and friction for developers," *IEEE Softw.*, vol. 40, no. 6, pp. 16–21, Nov./Dec. 2023, doi: 10.1109/MS.2023.3305718.
9. C. Jaspan and C. Green, "Defining, measuring, and managing technical debt," *IEEE Softw.*, vol. 40, no. 3, pp. 15–19, May/Jun. 2023, doi: 10.1109/MS.2023.3242137.