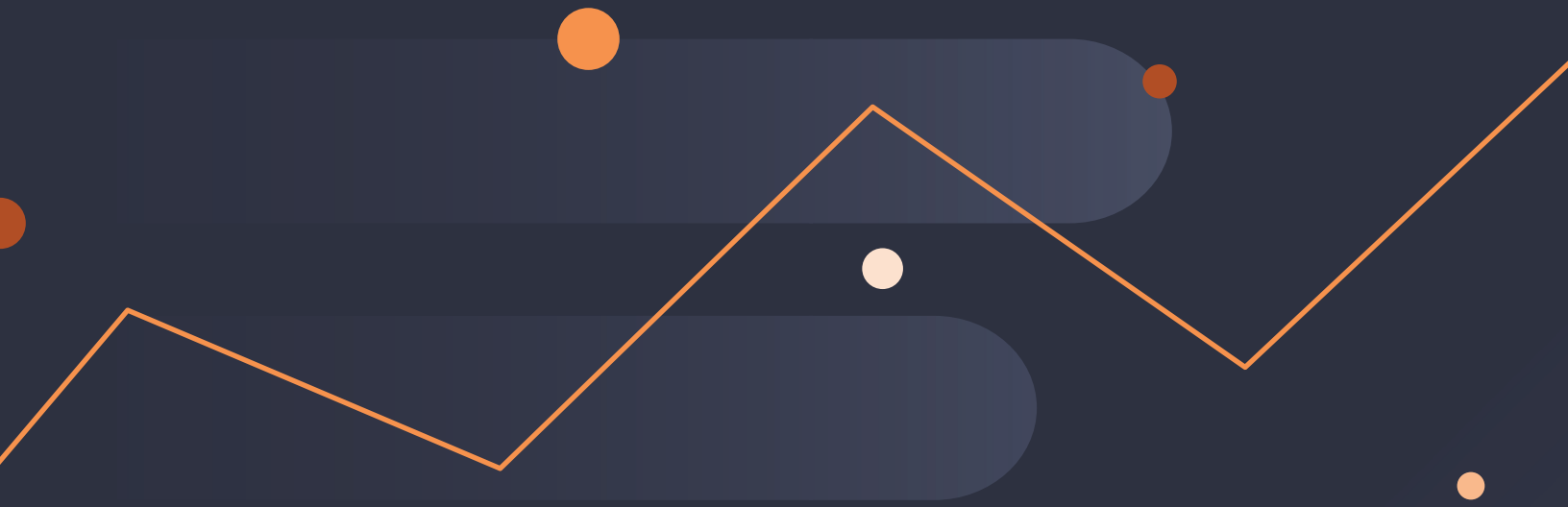
THE MANAGED OPEN SOURCE SURVEY



2020

TIDELIFT

INTRODUCTION

In June of 2020, Tidelift fielded our annual managed open source survey of technologists—including software developers, engineering executives and managers, architects, and devops pros. Over 600 people shared how they use open source software today, what holds them back, and what tools and strategies would help them use it even more effectively.

In this year's survey, we collected data about how the COVID-19 pandemic and ensuing recession are changing the way respondents' organizations think about and use open source. We explored how organizations manage their open source dependencies today, why and when their leaders encourage the use of open source, and the key benefits they get from increasing its use. We also learned more about the open source programming languages development teams turn to most and the policies they have in place around developer contributions to open source projects.

In this report, we'll highlight the most interesting revelations that help us understand how to make open source work even better for development teams and the organizations they work within.

Now, on to the findings!

FINDING #1

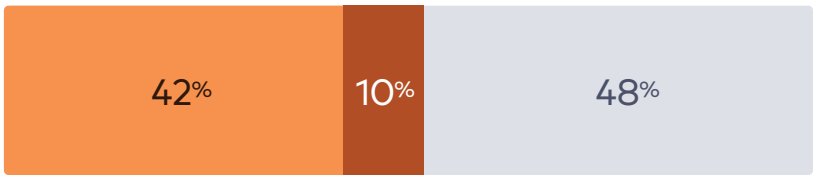
Open source use is rising during the COVID-19 recession

We've written previously at Tidelift about how organizations tend to encourage the use of open source even more when times get tough, in part because it helps them save money. So in this year's managed open source survey, we wanted to learn whether the economic ramifications of the COVID-19 pandemic were once again putting application development budgets under pressure—and indeed they are.

Forty-two percent of respondents reported their organization's application development budget was cut because of the current economic downturn, while only 10% said spending had increased.

42% of application development budgets cut in the current economic downturn

> During the current economic downturn, how is your organization's application development budget being impacted?



Based on 539 responses. "Don't know" excluded from calculations.

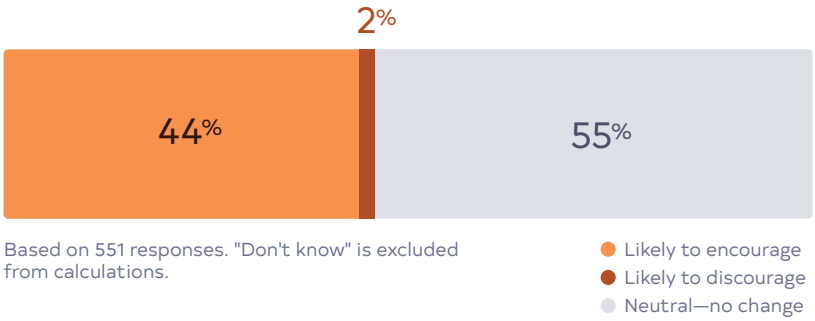
- Budget has decreased
- Budget has increased
- Neutral—no change

The pandemic hit Asia first, so it is unsurprising that 61% of respondents from the region—representing 15% of the study—reported budget cuts caused by the economic downturn. Economically sensitive industries like consumer, retail, and manufacturing were most likely to cite a negative budget impact. On the flip side, governments, non-profits, and large companies—often slow-moving ships that don't make dramatic mid-year budget changes—reported a minimal impact to their budgets.

When faced with reduced budgets, open source is viewed more favorably by organizations as a way to reduce costs, as it helps them innovate faster without having to write all of the code themselves or purchase expensive proprietary software tools. In fact, 44% of respondents believe their organization's leaders are likely to encourage the use of more open source for application development during the downturn, while only 2% expect its use to be discouraged.

44% of organizations likely to use more open source during the downturn

> During the current economic downturn, how likely are leaders in your organization to encourage or discourage the use of more open source for application development?

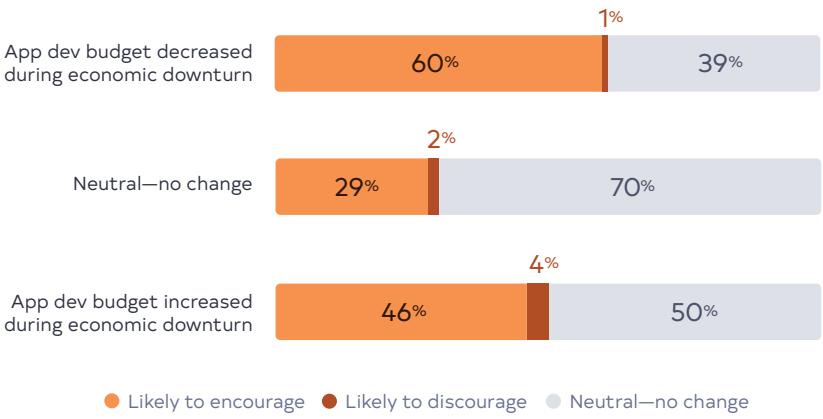


Based on 551 responses. "Don't know" is excluded from calculations.

Encouragement of open source is even more likely (60%) among organizations cutting budgets due to the economy. Interestingly, use of more open source is also being encouraged at organizations with rising app development spending.

Open source is encouraged most often at organizations facing budget cuts

> During the current economic downturn, how likely are leaders in your organization to encourage or discourage the use of more open source for application development?



App dev budget increased during economic downturn, n=54; Neutral—no change, n=248; App dev budget decreased during economic downturn, n=219. "Don't know" is excluded from calculations.

Finally, 49% of leaders at the largest companies (above 10,000 employees) are likely to encourage the use of more open source.

As the long-term move towards open source continues, our data shows that the recent economic downturn may be an accelerant. And as companies turn to open source for its cost savings benefits, we hope—as has occurred in previous open source migrations—they stay with open source for some of the other larger transformational benefits.

FINDING #2

Organizations look to open source to save time and money, while increasing efficiency

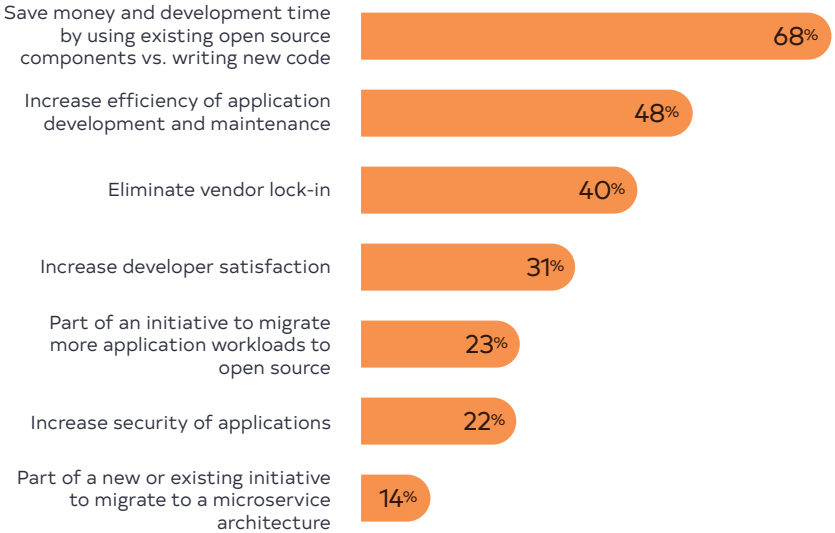
If organizations are increasing their use of open source during the current COVID-19-induced recession, we wanted to know why. Was it simply a cost savings measure or were there additional benefits that using more open source might bring?

Out of the full survey sample, 44% of respondents reported that their organizations would increase their use of open source for application development. So we followed up by asking about the key reasons for this increase in open source usage.

Unsurprisingly the top answer, chosen by over two-thirds of respondents (68%), was that open source helps them save money and development time by using existing open source components versus writing new code. This has always been one of the enticing benefits of open source—that developers move more quickly because they can choose from billions of lines of code that have already been written versus having to start from scratch.

Why do organizations use more open source during tough times?

> What are the key reasons why leaders in your organization are more likely to encourage the use of more open source for application development during the current downturn? (Select all that apply.)



Based on 239 respondents with leaders encouraging use of more open source during the economic downturn.

Forty-eight percent of respondents identified increased efficiency of application development and maintenance as a key reason why open source usage was being encouraged. This, too, is no surprise. When developers choose open source components versus writing code from scratch, they benefit from the fact the code is being used and maintained by a larger community beyond their own organization.

While there are few guarantees about quality and continued maintenance with open source application components (well, at least until the Tidelift Subscription came along), developers can often expect that, if they are making good component choices, the open source they use will continue to improve as long as the community that nurtures it stays healthy.

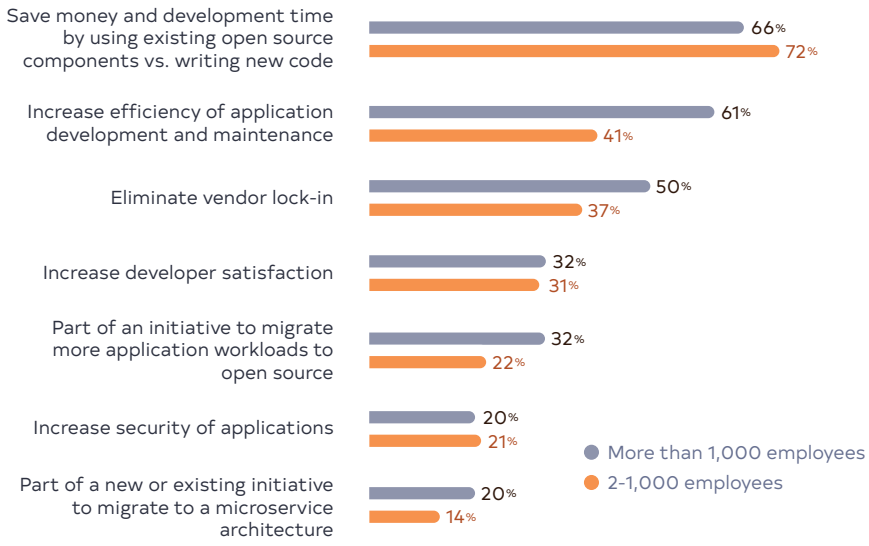
Interestingly, organizations with more than 1,000 employees were much more likely to cite efficiency (61% vs. 41% for organizations under 1,000 employees) as a reason for encouraging the use of more open source.

The third most important benefit of using more open source, chosen by 40% of respondents, was eliminating vendor lock-in. Clearly, more organizations are turning to open source as a way to replace expensive proprietary software and gain more control over future expenditures. Of note, this too was a more popular answer with larger organizations (50% for organizations with over 1,000 employees vs. 37% for organizations under 1,000 employees).

While only 20% of respondents cited having an internal initiative to move more workloads to open source, one-third of larger organizations over 1,000 employees (33%) identified this as a key reason, which tied for the third most popular choice among this group.

Key reasons for using more open source: small vs. large organizations

> What are the key reasons why leaders in your organization are more likely to encourage the use of more open source for application development during the current downturn? (Select all that apply.)



Respondents with leaders encouraging open source during the economic downturn: 22-1,000 employees, n=140; More than 1,000 employees, n=56.

Organizations of all sizes did not prioritize increasing the security of applications as a primary reason for moving to more open source. Only 22% of organizations identified this as a key benefit, and this was the most consistent response across different organization sizes.

While security may not be a primary reason to choose open source, it is a subject that receives intense attention in the open source world. This provides a nice segue into our next set of findings, where we explore some of the key concerns organizations have about using open source and how they can address them.

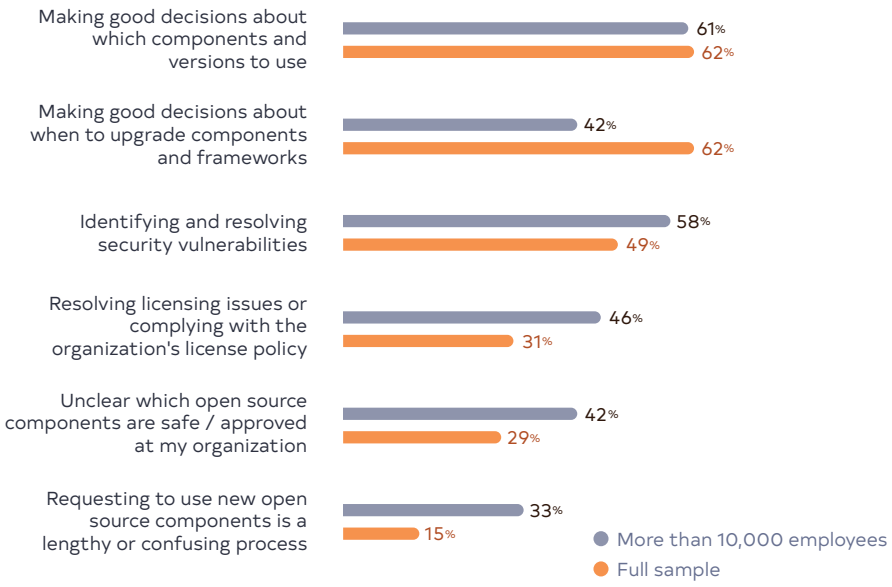
FINDING #3

Large companies are burdened by cumbersome open source approval processes

Organizations are accelerating their usage of open source because of how it helps them save time and money, while increasing efficiency. But increasing usage of open source also comes with its own set of challenges. In this year's survey, we asked technologists to indicate the most critical challenges their teams face when using open source software.

The most common challenges when using open source

> Which of the following challenges does your team face when using open source? (Select all that apply.)



Based on 562 respondents, 57 of which work at an organization with more than 10,000 employees.

The two most common challenges, faced by 62% of our respondents, are how to make good decisions about which components and versions to use and how to make good decisions about when to upgrade components and frameworks. These challenges happen at different times in the development lifecycle. Component selection happens when an application is initially written. In contrast, upgrading decisions usually occur as part of regular software maintenance.

While identifying and resolving security vulnerabilities is a challenge for 49% of respondents, it is even more critical in organizations with over 10,000 employees, with 58% of these respondents citing it as a key issue.

Two other challenges rated higher for organizations with over 10,000 employees than with the full sample.

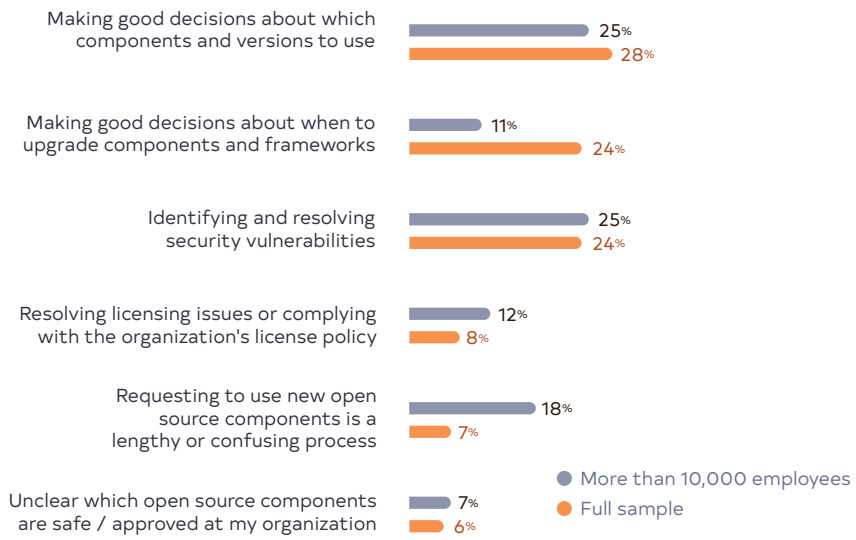
Where resolving licensing issues or complying with the organization's license policy was only chosen by 31% of the respondents overall, in organizations with over 10,000 employees, almost half (46%) face this challenge. This lines up with data from previous surveys, where licensing issues tend to be of more concern to larger organizations where the risks of non-compliance can be higher.

The other challenge that was more commonly selected by respondents in larger organizations was lack of clarity about which open source components are safe and approved to use. Forty-two percent of large organization respondents cited this challenge, versus only 29% in the full sample.

We also asked which of these challenges was most urgent. Respondents at the largest organizations were more than twice as likely than the average (18% vs. 7%) to say that requesting to use new open source components is a lengthy or confusing process. This matches what we hear from developers working in large enterprises—that the process for introducing new open source dependencies can be slow, bureaucratic, and frustrating.

The most urgent challenges when using open source

> Of those same challenges, which of the following is the MOST urgent to address?



Based on 548 respondents, 55 of which work at an organization with more than 10,000 employees. Not shown are the 3% that answered "other."

Clearly the challenge of bringing in new open source components is a sore spot at the largest organizations, with almost half (43%) reporting that their most pressing issue is either confusion about how to request a new component or making good decisions about which components or versions to use.

The more applications that use open source components, the more urgent this becomes. Among those with no more than half of their organization's projects containing open source, only 18% believe this the most urgent challenge. In contrast, choosing which components to use is the top challenge for 32% of respondents with 100% of projects containing open source.

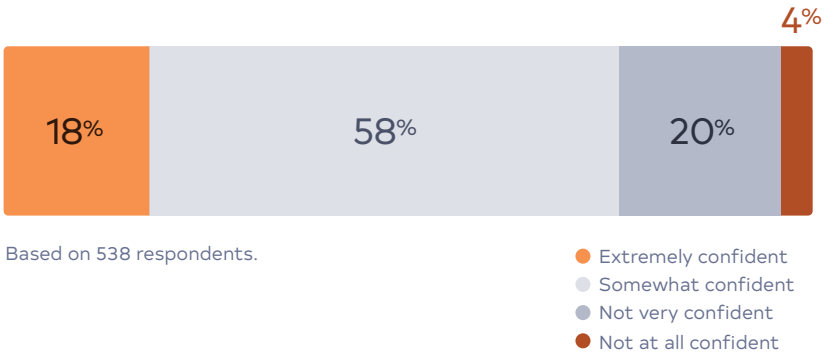
FINDING #4

Confidence in open source practices declines as size of company grows

As part of this year's study, we wanted to understand how confident technologists are about whether their open source components are secure, up-to-date, and well maintained.

Only 18% of organizations are extremely confident in their open source

How confident are you today that the open source components your organization is using are up to date, secure, and well maintained?



Based on 538 respondents.

- Extremely confident
- Somewhat confident
- Not very confident
- Not at all confident

Looking at the entire survey sample, a majority of respondents describe themselves as somewhat confident (58%). Almost one-quarter of respondents (24%) describe themselves as not very or not at all confident. And only 18% describe themselves as extremely confident.

There are two ways to view these high-level findings. On one hand, you could look at this data as proof that many organizations are at least somewhat confident that their open source components are up-to-date, secure, well maintained, with three-fourths of respondents (76%) describing themselves as somewhat or extremely confident.

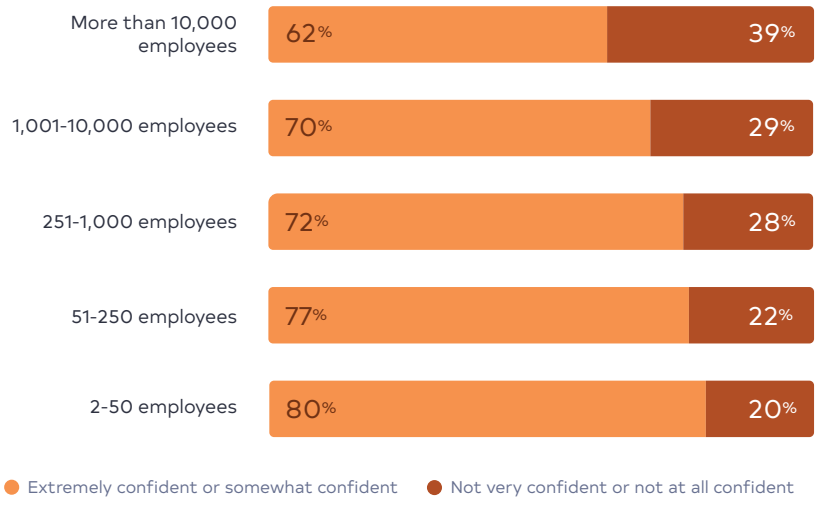
On the other hand, you might look at the fact that only 18% are extremely confident as a fairly disturbing finding. Is it good enough to be "somewhat confident" in the health of your organization's open source dependencies? Or could any one of these 82% less confident or not confident organizations become the next Equifax?

The data gets even more bleak once we split out the largest organizations of over 10,000 employees. Notably, only 61% are confident that their open source components are up-to-date, secure, and properly maintained (compared to 76% for the full sample).

And even more interesting, confidence in an organization's open source practices declines as the size of the organization grows. Meaning, the smallest organizations are the most confident while the largest organizations are the least confident.

Confidence in open source declines as organization size grows

> How confident are you today that the open source components your organization is using are up to date, secure, and well maintained?



2-50 employees, n=193; 51-250 employees n=98; 251-1,000 employees; n=64; 1,001-10,000 employees, n=65; More than 10,000 employees, n=57.

Our theory is that the largest organizations are the least confident for two key reasons:

1. **Awareness of open source-related risk is higher at larger companies.** Many larger organizations have entire departments related to security, risk, and compliance. Some of them even have open source program offices (OSPOs) solely dedicated to open source-related policies. This consequently creates heightened awareness of potential problems.

2. **The consequences of taking open source-related risks are higher in large organizations.** In a multi-billion dollar business storing immense troves of customer data or processing financial transactions at scale, one security issue caused by a poorly maintained open source dependency can cause irreparable harm (again, case in point, Equifax). So larger organizations may set a higher bar for how they measure their comfort with the security and health of their open source dependencies.

To put these findings in another light, imagine if you asked technologists this very same question for any proprietary software product they buy today. If only 18% of respondents said they were extremely confident that this product was up-to-date, secure, and well maintained, how successful do you think that product would be?

When it comes to the open source building blocks at the heart of our organizations' applications, we should expect no less. We need to ensure that more organizations are extremely confident in the health and security of these critical open source components.

FINDING #5

Upgrading versions and adapting to bugs are the two most common open source maintenance activities

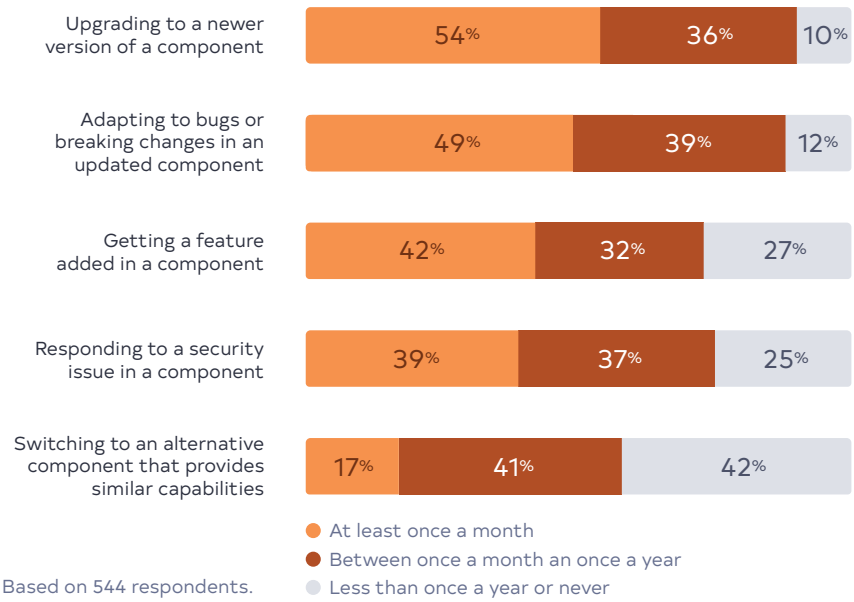
Open source has sometimes been referred to as free, as in a free puppy. Like a free puppy, while the initial acquisition cost may be nothing, there are many hidden costs,v including keeping the code secure and well maintained.

We wanted to understand which of these time-stealing maintenance activities around open source were most common amongst the organizations in our survey. In particular we wanted to know about the prevalence of five common maintenance activities we hear technologists regularly mention.

According to our results, these activities are quite common—three-fourths (74%) of organizations perform at least one of these activities monthly or more frequently.

The most common open source maintenance activities

How often do you or someone on your team perform the following open source maintenance-related activities?



Based on 544 respondents.

Upgrading to a newer version of a component is the most common maintenance-related activity, with 54% doing so at least monthly. Almost as many (49%) adapt to a bug or other issue in an updated component at least monthly.

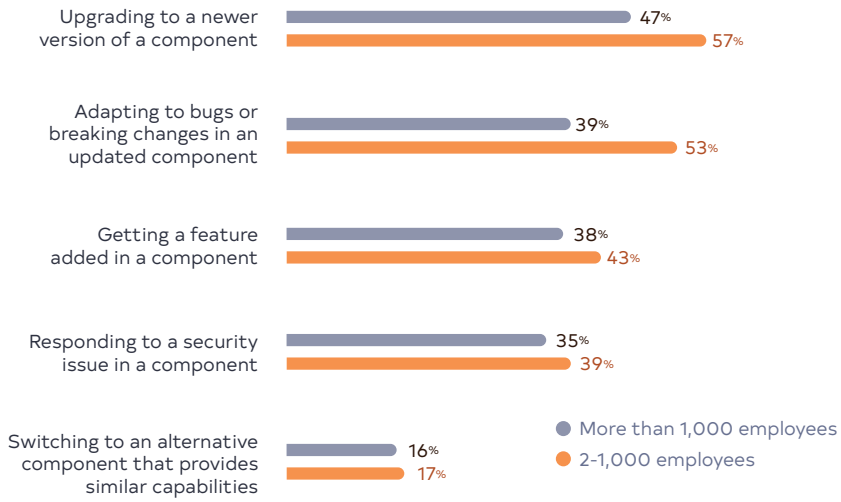
Switching to an alternative component is less common, but still occurs at least once a year at 58% of the companies surveyed. One reason swapping components does not happen as often is because of the added time needed to troubleshoot possible issues with all the application's dependencies. Case in point: fear of being swamped by unexpected problems is why moving to a new major version of a framework or library was the most commonly cited open source maintenance challenge in our 2019 study.

As we learned in an earlier finding, decisions about making a major version change continue to be painful, which is why 62% of our 2020 respondents said it is a challenge their team faces when using open source. One of the best ways to address this challenge is actually spending a little more time making regular, incremental updates to the components a company uses.

Overall, these sorts of open source maintenance activities were more commonly reported at organizations with 1,000 or fewer employees. For example, among this group, 53% of their teams adapt to bugs or breaking changes at least once a month, while that only happens at 39% of larger organizations.

Impact of organization size on open source maintenance activities

Percentage of respondents performing these tasks at least once per month



2-1,000 employees, n=349; More than 1,000 employees, n=122.

While open source maintenance issues will always be there—the free puppy will never be free—we can improve the ways we address these issues. This is part of what we are tackling at Tidelift: ensuring that developers can minimize the time they spend on open source-related maintenance activities so they can maximize the amount of time they spend working on the unique code that truly differentiates their application.

FINDING #6

The top programming languages organizations rely on are JavaScript, Python, and Java

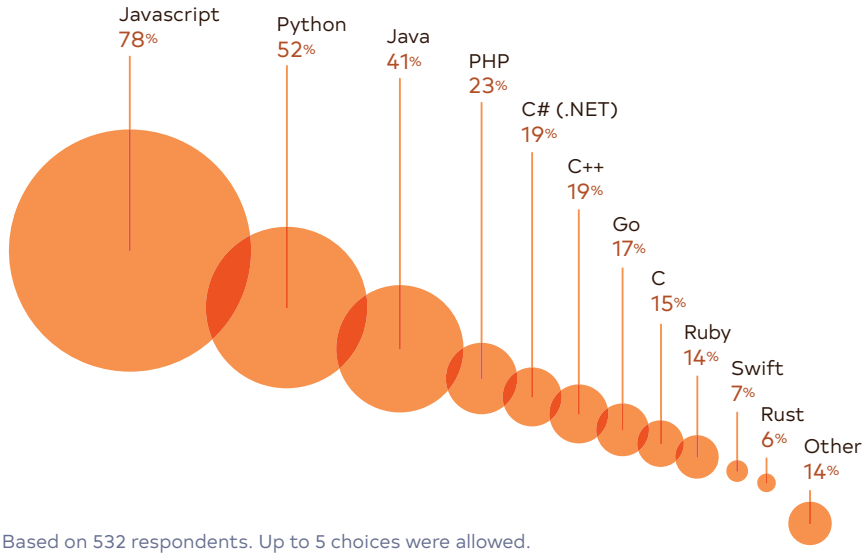
Seeing your favorite language gaining popularity is professionally affirming. The RedMonk Programming Language Rankings, TIOBE Index, IEEE Spectrum Interactive Rankings, The State of the Octoverse, Stack Overflow Developer Survey—all use different methodologies to measure things like attention, job opportunities, and the prevalence of new code being written.

None will tell you which language is functionally better, but they provide actionable insight into 1) which languages you should be learning and 2) which languages you should be using for certain types of projects.

We wanted to add some meaning of our own to this crowded pool of data with a few questions about the programing languages technologists rely on most. We started by asking respondents to select the top open source languages their organization relies on, allowing them to choose up to five languages.

The three top languages: JavaScript, Python, and Java

> What are the top open source languages your organization relies upon? (Select up to five.)



Based on 532 respondents. Up to 5 choices were allowed.

Not surprisingly, the top three languages by far were JavaScript (78%), Python (52%), and Java (41%). In fact, the top five languages reported by our respondents exactly match the most recent RedMonk language ranking from June 2020, with PHP and C# (.NET) ranking fourth and fifth.

Ranking	TIDELIFT	RedMonk
JavaScript	1	1
Python	2	2
Java	3	3
PHP	4	4
C# (.NET)	5	5
C++	6	5
Go	7	15
C	8	8
Ruby	9	7
Swift	10	11
Rust	11	20

RedMonk's ratings were published as a Top 20 that also included CSS, TypeScript, R, Objective-C, Scala, Shell, PowerShell, Perl, and Kotlin.

Go ranked higher in the Tidelift survey than in RedMonk's analysis (7 vs. 15), while Ruby ranked lower in the Tidelift survey (9 vs. 7). At the bottom of the Tidelift list were up-and-coming languages Rust (6%) and Swift (7%), which are often used for mobile app development.

RedMonk's analysis looked at over 50 languages, of which eight ranked higher than Rust, which by some measures has never been hotter, with TypeScript showing up at #9. The next time Tidelift asks this question, we expect to include TypeScript and Kotlin in the list of languages, as both received many write-in responses in the 2020 version of the study.

Fans of Java often complain that it is underrated in language studies because its use is concentrated in larger enterprises. The survey supports this conventional wisdom, as 66% of organizations with more than 1,000 employees rely on Java, which moves it ahead of Python among this cohort.

Although Python is popular among students and hobbyists, our data shows that larger organizations are more likely than smaller organizations to rely on it (61% of organizations with more than 1,000 employees vs. 49% of organizations with less than 1,000 employees).

It is important to note that larger organizations selected more languages. Organizations with more than 1,000 employees on average identified 3.5 languages, while those in organizations with 1,000 employees or fewer only chose 2.9 languages. Since larger organizations have more applications, it is not surprising that they rely on more languages.

Java and Python used in more apps at larger organizations

> What are the top open source languages your organization relies upon? (Select up to five.)

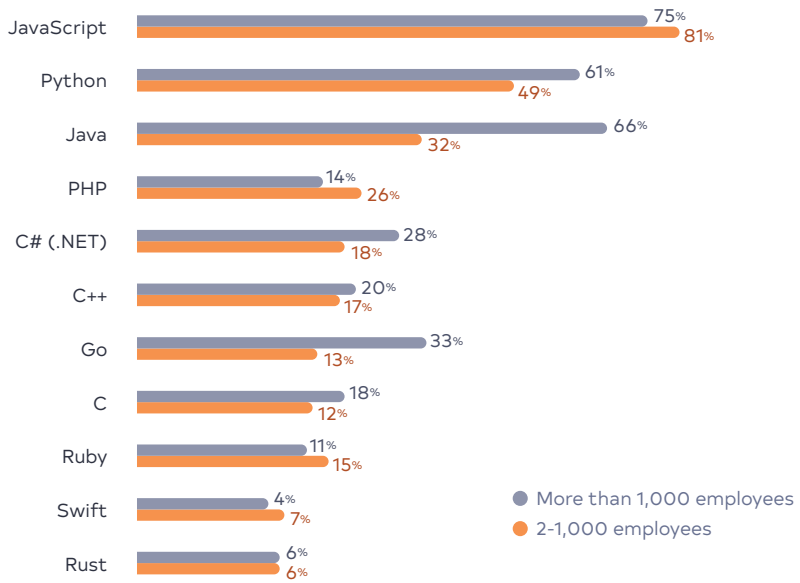
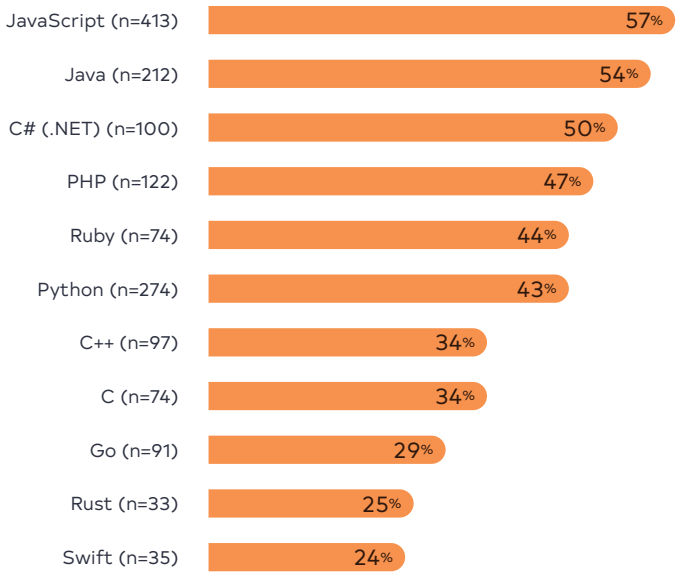


Chart shows the mean figures. All data is statistically significant, with at least 50 respondents represented in any single bar.

We also wanted to understand how critical each of these languages is across an organization's applications. Respondents were shown the languages they had chosen in the previous question, and asked what percentage of their organization's applications relied on each language.

Most popular languages, by percentage of apps that use them

> What percentage of applications in your organization use each of these languages?



JavaScript stayed in first place, with the average organization using it in 57% of their applications. Java jumped to second place at 54% since it is often an integral part of larger enterprises' internal applications.

Most notably, C# (.NET) and Ruby improved significantly by this metric at 50% and 41% respectively. Even though the sample sizes are smaller (100 respondents for C# (.NET) and 74 for Ruby), this tells us that the organizations that rely on these languages do so quite heavily.

The other outlier here is Python, which was the second most relied upon language overall, but dropped to sixth in terms of the percentage of applications that rely on it with an average percentage of 43%. In keeping with its dual role as a primary language and a glue-layer for smaller projects, Python is used by a lower percentage of applications than JavaScript and Java.

Older, entrenched languages battle for developers' attention against new languages with new approaches. The two metrics we are tracking—top languages being used and percentage of applications using each of those top choices—provide valuable data technologists can use to make decisions on which languages are most established, which are gaining momentum, and which are losing momentum.

FINDING #7

Formal processes around open source management are more common, but it is still basically a free for all

More and more organizations are taking a top-down approach to promoting open source, but for many developers, their first open source experiences take place outside of the workplace. They learn a new language by going through online tutorials or pursuing a hobby. Then, at some point they get an epiphany and realize they could apply this language to the programming they are doing at their day job.

What happens next is the subject of another question we asked in this year's survey. We wanted to learn whether it's common for organizations to have formal processes for introducing new open source components into their codebase. We had previously asked a question about this in our 2018 survey and were interested to learn whether organizations were maturing in their practices around managing open source.

We saw a jump from 2018 to 2020 in terms of the percentage of organizations that have a formal evaluation and approval process for introducing new open source dependencies. Only 10% of respondents in 2018 had a formal process, and the percentage nearly doubled to 17% in this year's survey. In addition, the percentage of organizations with an informal process also increased slightly, going from 22% in 2018 to 26% this year.

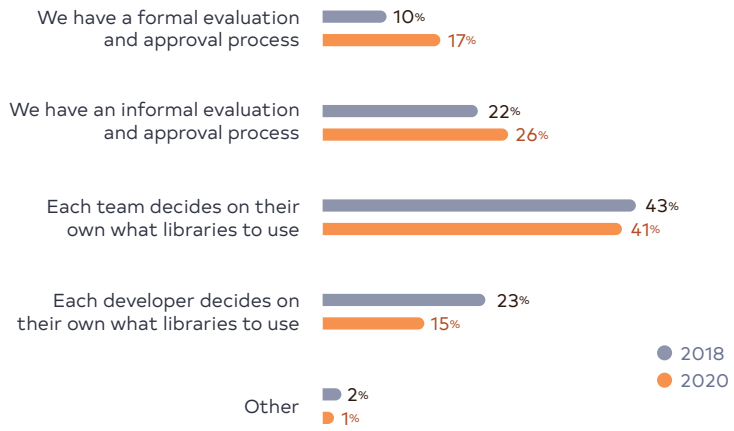
On the other end of the spectrum, a smaller percentage of organizations (15%) allow each developer to decide on their own which components to use as compared to 23% in the 2018 survey.

So at the extremes, more organizations are introducing formal or informal processes for managing open source dependencies, while fewer organizations are allowing each developer to make their own decisions about bringing in new components.

Yet the most common process for introducing components is to make decisions at the team level. This response stayed consistent between the two surveys, with 43% in 2018 and 41% in this year's study.

Only 17% of organizations have a formal process for evaluating open source

> Which of the following best describes the evaluation or approval process you use for introducing new open source components?



Based on 526 responses in 2020 and 704 responses in 2018. The 2018 question used the words "dependencies" and "libraries" instead of "components."

As one might expect, the largest companies are more likely to have some sort of organization-wide process—formal or informal—for managing their open source components. Sixty percent of organizations with 10,000 or more employees selected one of these two responses compared with 41% for organizations with less than 10,000 employees.

But zooming out for a second, it is still important to note that less than half of organizations (43%) have any organization-level evaluation and approval process—formal or informal. And over half (56%) of organizations are still making decisions at a team or individual developer level.

We are making progress, but it is slow. Why?

As we wrote about in a previous finding, one possible reason is that approval processes at larger organizations are often viewed as burdensome or bureaucratic. There is a strong link between these two findings as respondents who reported that their organizations required a formal approval process were much more likely to say that the process is unclear, confusing, or takes too long (50% for those with a formal process vs. 37% for the full sample).

Perhaps organizations fear putting more processes in place because they worry these processes might reduce risk and improve reliability—but force development velocity to take a hit. Which is where there is a huge opportunity for organizations to implement strategies that don't force them to make a choice between these two extremes.

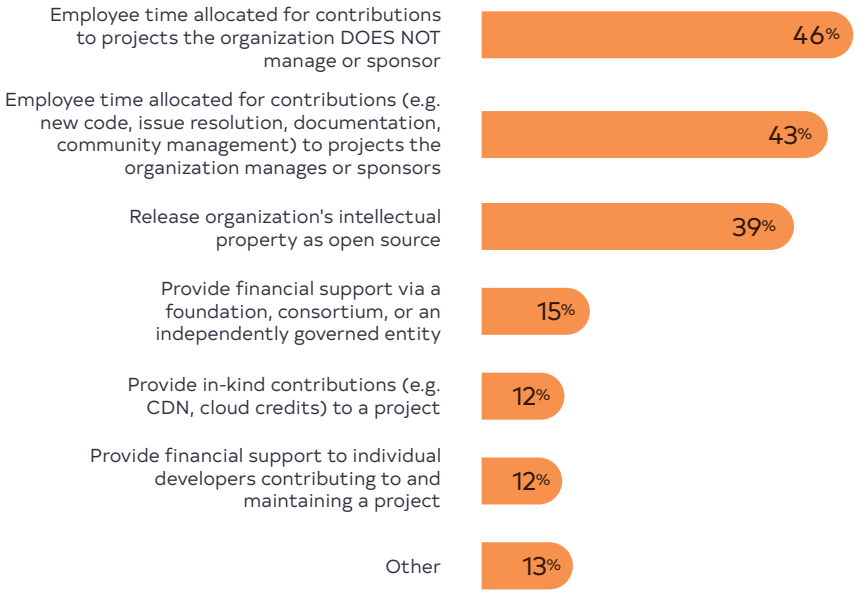
FINDING #8

The top ways organizations contribute to open source—and where they see ROI

If your organization uses open source to build applications, it is probably also contributing to open source in some way. More than four-fifths (83%) of respondents claim their organization contributes using at least one of six different methods we asked about in this year's survey.

Top ways organizations contribute to open source

> In which of the following ways does your organization contribute to open source? (Select all that apply.)



Based on 441 respondents.

Almost half (46%) of respondents said their organization contributes employee time to projects the organization does not manage or sponsor, and almost as many (43%) contribute employee time to projects the company does manage or sponsor. Since respondents could choose as many options as applicable, many selected both, so overall two-thirds said their organization allocates employee time for open source contributions. That is encouraging news, but there is still room for improvement as only 28% of respondents allocate employee time to both sponsored and unsponsored projects.

Almost two-fifths (39%) of respondents reported that their organization contributes by releasing code as open source, which was the third most commonly cited option.

None of the three remaining options were chosen by more than 15% of respondents—and, notably, each cost more than just employee time. They were: providing financial support via a foundation, consortium, or independently governed entity; providing in-kind contributions to a project; and providing financial support to individual developers contributing to or maintaining a project.

Sadly, only 22% of organizations currently provide financial support to the projects themselves, whether via a foundation, consortium, or independent entity (15%), or via support to individual project maintainers (12%).

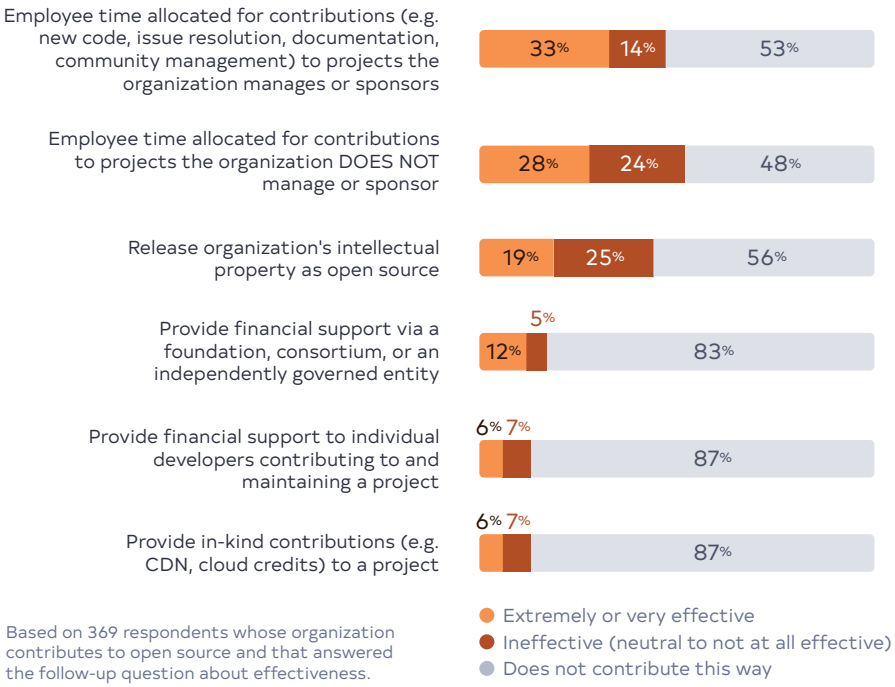
Not surprisingly, respondents at organizations with more than 10,000 employees were twice as likely as the average (30% vs. 15%) to financially support a foundation, consortium, or another type of independently governed open source organization. Although it does not require cash, the largest organizations are also almost twice as likely (19% vs. 10%) to provide in-kind contributions like cloud credits.

Many organizations question the ROI of traditional contributions to open source

We next asked respondents to share their views about the effectiveness of their organizations' contributions, and sorted by both effectiveness and popularity—since some of these methods are employed much less often than others.

The most effective ways for organizations to contribute to open source

> Please rate the effectiveness of each of your organization's contributions to open source.



Based on 369 respondents whose organization contributes to open source and that answered the follow-up question about effectiveness.

The amount of control an organization has over a contribution appears to influence opinions about effectiveness. Thus, the overall most effective and popular method of contribution was allocating employee time for contributions to projects the organization manages or sponsors, with 33% of all respondents rating this as extremely or very effective.

The most popular response—allocating employee time for contributions to projects the organization does not manage or sponsor—was only second on the list of most effective, with 29% rating this as effective and 24% rating it as ineffective. So while more popular, respondents don't find this approach as impactful.

Even though it is a popular contribution method, less than half of organizations that release their intellectual property into the open source commons believe this is an effective approach (19% effective vs. 25% ineffective).

While the other three options we studied were markedly less popular, the 17% of respondents who reported their organization contributes to an open source foundation, consortium, or independently governed entity feel like these investments are effective (12% effective vs. 5% ineffective).

These data points lead us to the conclusion that there is a lot of room for new approaches when it comes to improving the effectiveness of how organizations contribute to open source.

There are some proven effective methods, like contributing time, code, and other resources like documentation to open source projects. And contributing to foundations can be an effective approach for the deep-pocketed organizations that can afford to do it at the level required to have a strategic impact on the highest-profile open source projects.

For organizations that lack these multi-million-dollar budgets, we believe there are better ways to contribute more effectively to the future of open source. Ways that level the playing field and further the success of the tens of thousands of open source projects that are equally critical to enterprise application developers, regardless of their size, influence, or ability to contribute code or other resources. Ways that directly create a financial impact for open source creators and maintainers.

This is one of the key reasons we created the Tidelift Subscription—to make it easier for more organizations to pay the maintainers of the projects they rely on, while getting a clearly defined SLA and set of promises about the future of those projects in return.

FINDING #9

Almost half of organizations have open source contribution policies. We learned what they cover.

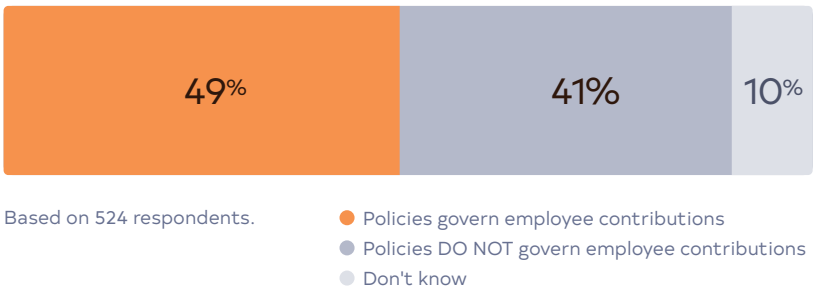
Should organizations contribute to open source? Old question. Today, the benefits of contributing to open source are better understood than ever. And as we found in last year's survey, the vast majority of respondents (84%) already contribute to open source projects more than once a year.

Today, the more interesting question is *how* should organizations contribute to open source. When it comes to managing the contributions of their employees, is it better to take a more formal or informal approach? Should they allow contributions to projects that don't directly impact the organization's interests? And where does responsibility for and ownership of this work begin and end?

In this year's survey, we touched on some of these questions. In particular, we wanted respondents to share more about how their organizations manage open source contributions today.

We learned that almost half (49%) have at least one policy in place to govern open source contributions (respondents could select more than one option), while 41% do not have a formal policy and 10% don't know.

49% of organizations have policies governing employee contributions to open source



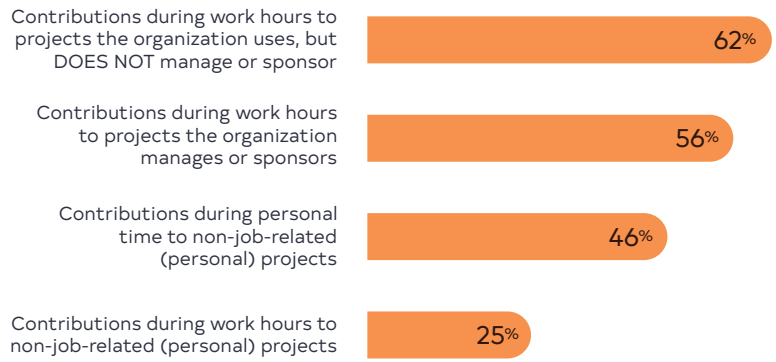
Worth noting here—as Heather Meeker has pointed out—organizations without an open source contribution policy actually just don't have a written policy.

> *"You have a policy, whether it is written down or not. It could range from 'no open source at all' to 'anything goes.' The question is: does anyone follow it? Is it sensible for your business? Written policies are useful to communicate your expectations about use of open source in your organization, particularly to outsourced developers or engineers in subsidiaries and affiliates, whom you may not see every day."*

For those that do have written policies, what do the policies cover? Are they intended to restrict open source contributions? Are they designed to minimize organizational risk? Or are they designed to encourage open source contributions?

What do organizations' open source contribution policies cover?

> Does your organization have policies that permit employees to make the following types of open source contributions? (Select all that apply.)



Based on 259 respondents at organizations with a policy governing employee contributions.

Sixty-two percent of survey respondents report that their organizations' policies permit contributions to projects the organization uses, but doesn't manage or sponsor; 56% of policies permit contributions to projects the organization manages or sponsors; and 37% permit both.

Many organizations' policies also cover contributions to personal projects. When it comes to contributing to non-job-related (personal) projects, 24% of organizations have policies about making these contributions on personal time, while 13% have policies covering contributions during work hours.

For a deeper understanding of the motivations behind these different sorts of policies, and the characteristics of the best policies, see Tobie Langel's excellent presentation Open Source Contribution Policies That Don't Suck.

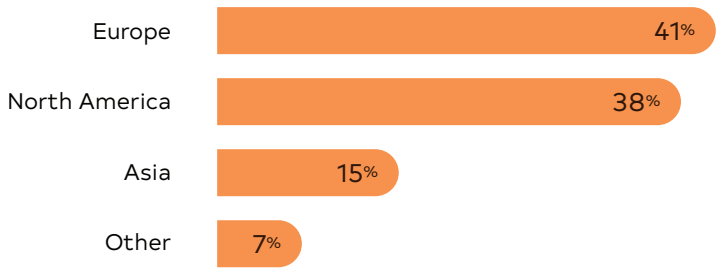
One thing this data made clear to us: we have only scratched the surface of this line of questioning. In future surveys, we'll plan to explore contribution policies in more detail. Do technologists view their organizations' policies positively or negatively? Are the policies seen as restrictive or permissive? And most importantly—are they helping organizations contribute to open source more effectively?

This marks the third year Tidelift has conducted a survey to answer our most pressing questions for technologists using open source to develop applications (here's where you can download the results from our 2019 and 2018 surveys and a third shorter survey we did in between).

The 2020 Tidelift managed open source survey was conducted from May 28 through July 4, 2020. Participants were contacted via Tidelift and Changelog email lists and social media. We screened respondents to make sure they use open source to build applications at work, and the full survey sample was 638 respondents, including software developers, engineering executives and managers, architects, and devops pros.

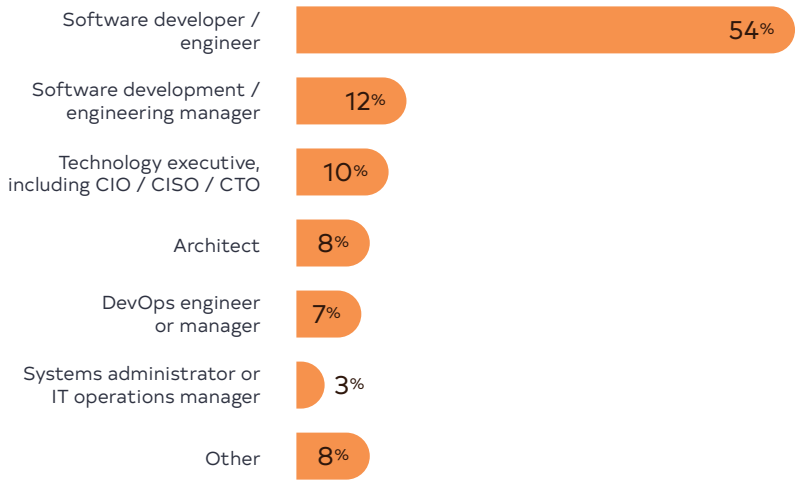
The next few charts give you a bit more detail about the survey demographics:

Which geographic region are you located in?



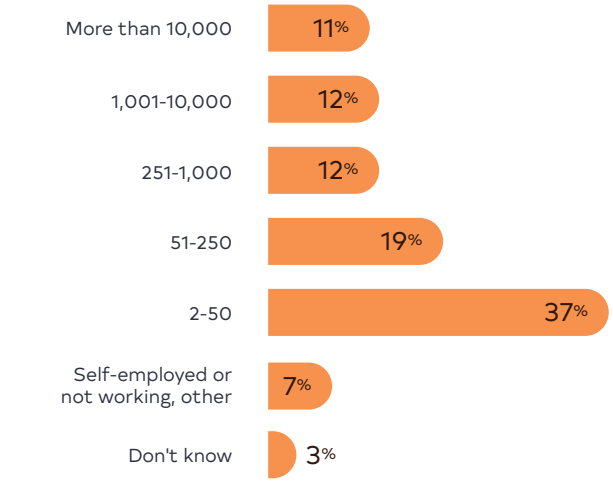
Based on 523 respondents.

Which job category most closely matches your role?



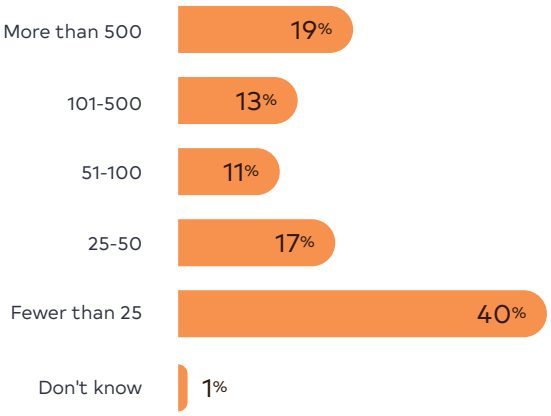
Based on 521 respondents.

How many employees work for your organization?



Based on 523 respondents.

How many people work in and around software development at your company?



Based on 476 respondents.

WHAT IS MANAGED OPEN SOURCE?

Why do we call this survey the "managed open source" survey? What exactly is managed open source?

In short, managed open source provides a way for organizations to offload the complexity of managing the open source components they use to build applications. It takes precious developer time to keep track of all of the components an organization is using and to keep them well-maintained, properly integrated, and secure.

A managed open source approach helps organizations give their development team a "paved path" where the components they are using are actively maintained to continually meet corporate standards around security, maintenance, and licensing. This allows development teams to focus less time and attention on the open source pieces of their application, and more on the code that is unique to their organization.

Because of this, a managed open source approach helps organizations cut costs, accelerate development, and reduce risk when using open source to build applications.

In Tidelift's approach to managed open source, we partner directly with the independent creators of open source to create customizable catalogs of components that just work. We cover thousands of packages across JavaScript, Java, Python, PHP, Ruby, and .NET. We partner with the open source maintainers to provide ongoing security updates, active maintenance, and accurate licensing information for these components.

Through the Tidelift Subscription, organizations can then create their own customizable catalogs of known-good open source components that are safe for developers to use. For more information or a free demo, visit Tidelift.com.

ABOUT TIDELIFT

Tidelift is the largest provider of commercial support and maintenance for the community-led open source behind modern applications.

Tidelift partners with independent project maintainers to make it safer, easier, and more cost-effective for application development teams to build with open source, so they can create even more incredible software, even faster.

The Tidelift managed open source solution delivers customizable catalogs of components that are actively maintained, secure, and accurately licensed, enabling development teams to build and deploy with confidence. *Tidelift makes open source work better—for everyone.*

For more information:
Tidelift.com

TIDELIFT