swimm

2024

The state of developer knowledge sharing

Impact, challenges, and emerging AI trends

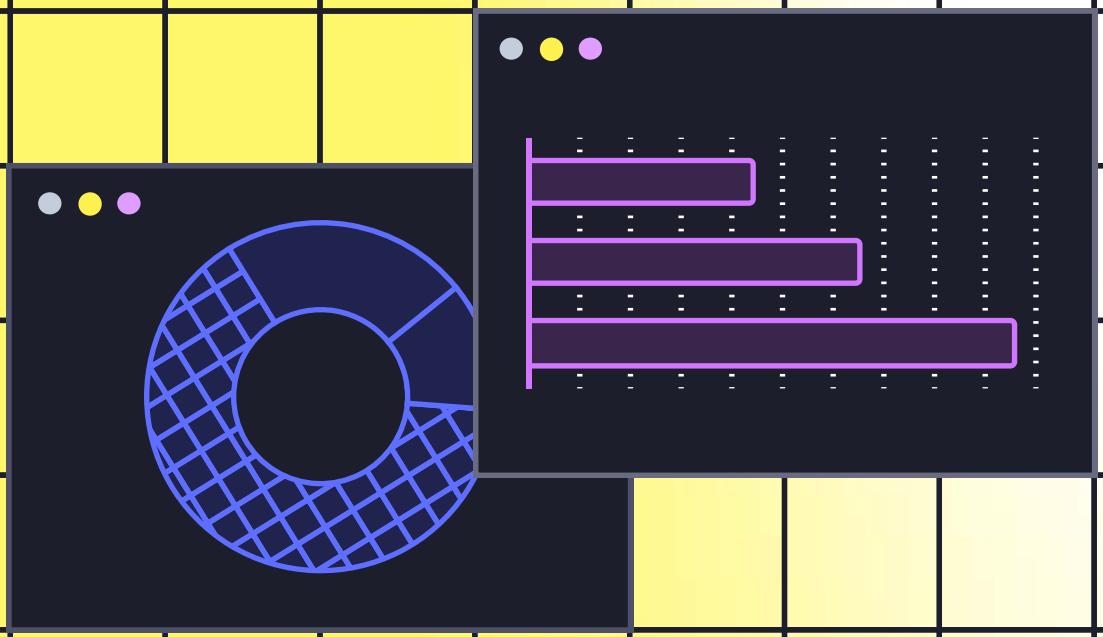


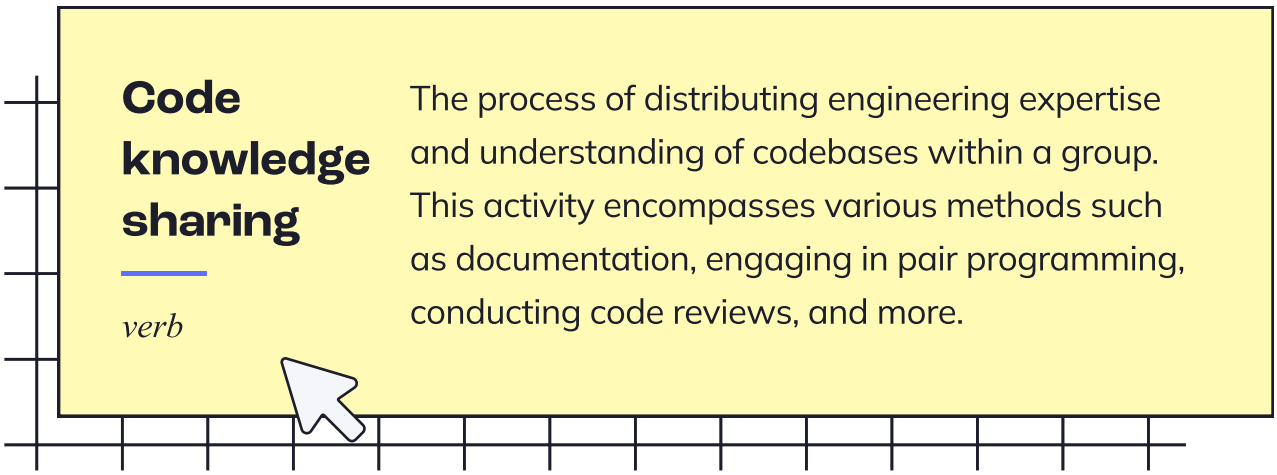
Table of Contents

Chapter 1

Introduction & key findings

Introduction:

With almost all developers (92%) using or experimenting with AI coding tools, AI continues to transform nearly every aspect of the software development lifecycle. Despite rapid technological advancements, the fundamental requirement to understand code remains unchanged, for developers and managers alike. Whether code is manually written by a developer or generated by AI, eventually, someone else will need to understand and interpret that code.



Investments to improve the ways developers understand code are usually about sustaining or improving developer velocity. The argument here is that with accessible technical knowledge, developers can quickly find the information they need to complete tasks or make better decisions.

This is a great starting point, but there's more to it:

- Onboarding new developers to the team
- Working with an unfamiliar area of the codebase
- Enforcing standards during development
- Reducing the burden of code reviews
- Responding to incidents faster
- Improving the developer experience

As codebases become increasingly complex due to factors like advanced infrastructure, the use of multiple programming languages, distributed architectures, and a significant rise in AI tool usage, the need to understand code becomes even more mission critical for teams.

This survey highlights the challenges and benefits of code knowledge sharing, revealing differing perspectives between managers and developers on this issue. It emphasizes the urgent need to provide developers with the best tools and solutions for navigating and understanding complex codebases.

Key findings:

01. Understanding code accelerates development velocity by 50%

73% of developers surveyed recognize the importance of code understanding in boosting their productivity. They believe that sharing precise, comprehensive, and deep insights about code can, on average, enhance their project delivery speed by 50%. These findings highlight just how crucial the exchange of technical knowledge is for accelerating development cycles, simplifying processes, and enhancing developer experience. As engineering organizations adjust their strategies to keep up with AI advancements, prioritizing strong code knowledge transfer practices is crucial for maintaining high levels of velocity and fostering innovation.

02. Only 1% of developers believe their company excels in sharing code knowledge

Less than half of the participants (46%) feel confident in their company's ability to share code knowledge effectively, with only 1% considering their organization very successful in this area. This lack of confidence is alarming, given the significant benefits that effective code knowledge sharing has on speeding up project delivery. Moreover, as AI-generated code becomes more common, the need for efficient knowledge transfer among engineers is increasingly critical.

03. It's time to implement your AI strategy (if you haven't already started)

An overwhelming 99% of organizations surveyed are either currently using or intend to implement AI tools for sharing code knowledge. Of these, 30% are already employing AI tools, and another 47% plan to do so within the coming year. Developers everywhere are making AI a part of their development flow and it's clear that engineering organizations that are not adopting these tools risk falling behind.

04. Developers want AI tools that integrate with their existing workflows & prioritize automation

Respondents emphasized several crucial features for code knowledge sharing tools, notably the ability to automatically keep sources of technical knowledge up to date (internal documentation, for example) (45%), offer dashboards with usage metrics (42%), and facilitate the creation and discovery of technical codebase specific knowledge within the IDE (41%).

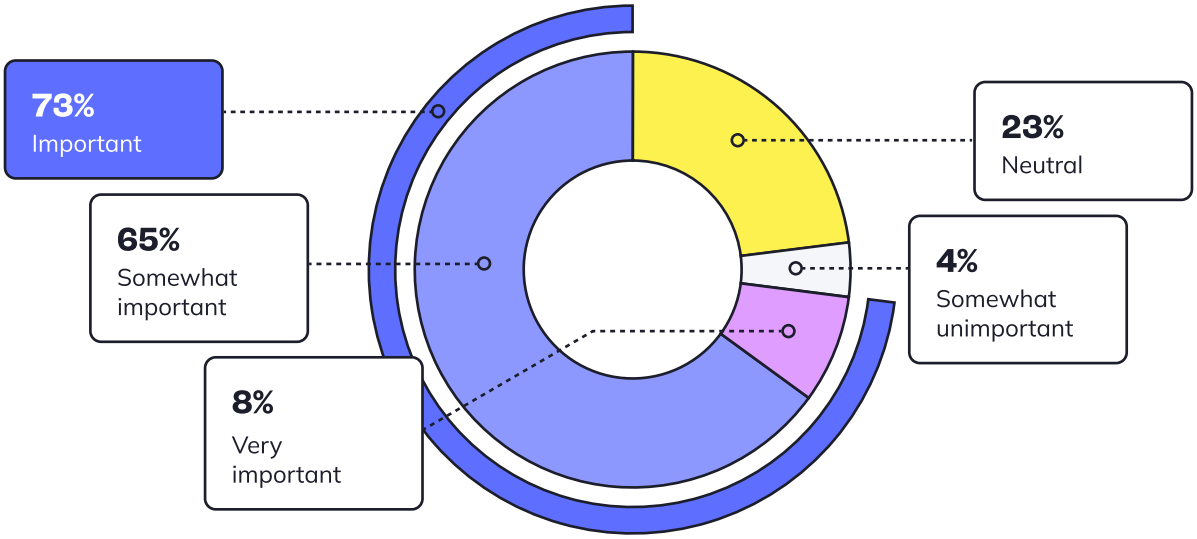
05. Effective knowledge management requires implementing tailored solutions

Developers responded almost equally to the challenges they face in sharing technical knowledge, recognizing the impact that effective knowledge sharing has on productivity. Developers struggle to efficiently capture technical knowledge and managers fear losing valuable insights when key team members move on. These findings highlight the need for the implementation of comprehensive solutions that simplify capturing and preserving technical knowledge and expertise.

Chapter 2

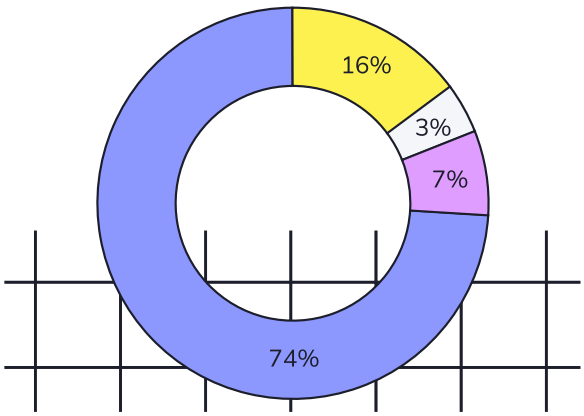
Developer productivity: Expectation vs. reality

A clear majority of respondents (73%) confirm the critical role code knowledge sharing has in boosting developer productivity.



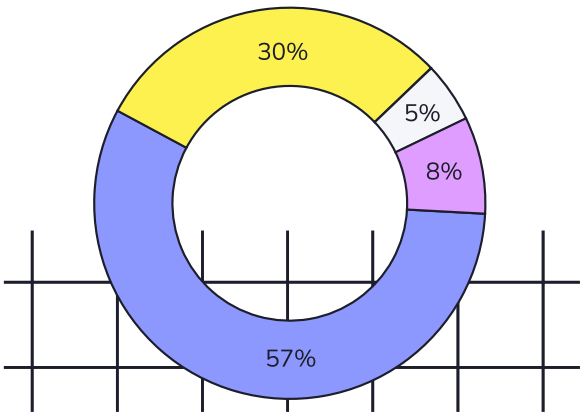
Developers' perspective

81% of team members acknowledge the importance of sharing code knowledge for developer productivity.



Management perspective

A smaller proportion (65%) of senior staff, including managers and directors, recognize its importance.



Bridging the knowledge gap

The gap in views on sharing code knowledge between developers and senior management emphasizes the importance of managers engaging with their teams when evaluating methods for knowledge transfer. The fact that 30% of management is neutral about the impact of knowledge sharing on productivity suggests they might not be using metrics that clearly link knowledge sharing to deployment frequency.

Alternatively, it could be that as management becomes more distanced from the day-to-day realities of coding, they may not immediately see the direct correlation between sharing code knowledge and improvements in project delivery speeds and overall productivity. By narrowing this divide, organizations not only enhance efficiency but also safeguard the business side of things against system outages and vendor disruptions.

Chapter 3

Developer questions are a major time usage

✳

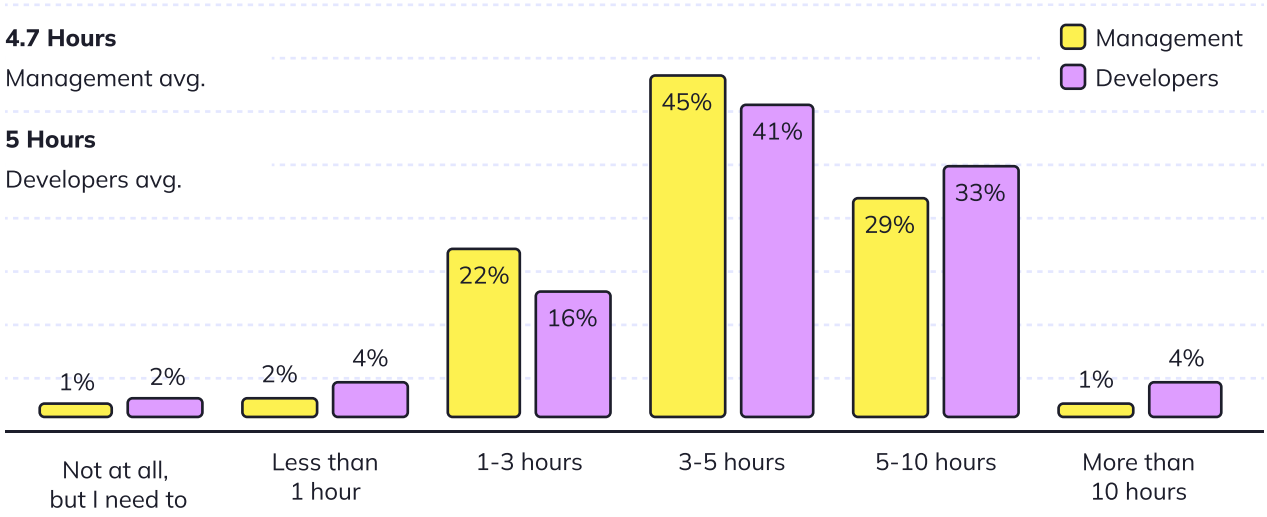
& only increase as companies scale

Almost all respondents, 94% of developers and 97% of managers, spend significant time (4.9 hours/week, on average) searching for answers to questions about code.

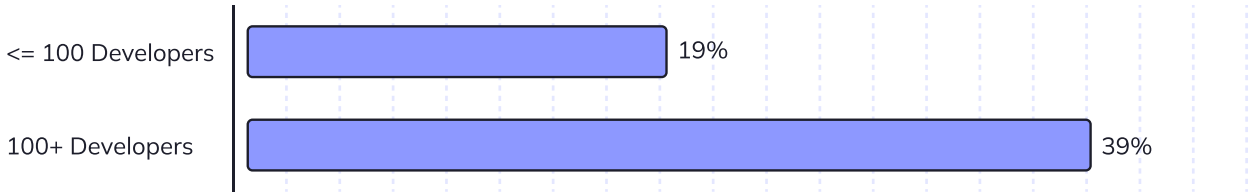
However, the data presents an interesting trend: the weekly time spent looking for information to understand code or answering code questions grows as companies scale and their codebase becomes increasingly complex. Developers from companies with 100+ engineers allocate 5-10 hours per week to these tasks. That's 10% of their workweek.

Additionally, developers' self-reported time usage skews higher (5 hours), while management (4.7 hours) tends to underestimate the amount of time their team spends searching for answers to code-related questions.

Weekly time spent on code understanding and answering questions:



5-10 hours spent by number of developers in the company:



The distance to knowledge

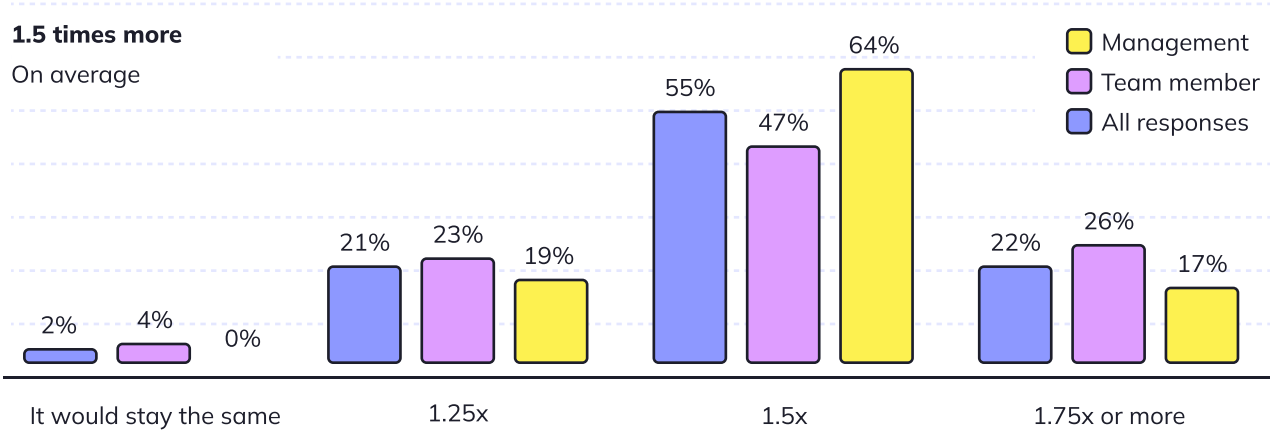
The relationship between a company's size and time dedicated to managing knowledge highlights the scaling challenges engineering organizations face. As companies grow, accessing relevant knowledge becomes harder, further complicated by increasingly complex codebases. However, when information about a company's codebase is made widely available and accessible, it greatly reduces the gap to necessary knowledge, allowing developers to focus more on building and fully utilizing their creativity.

Chapter 4

Developer productivity can be improved by >50%

When knowledge about code is shared effectively, the impact is clear: developers report a more than 50% increase in delivery velocity.

Impact on delivery velocity by seniority:



Once again there's disparity in the perspectives between developers and managers:

Developers' perspective

73% of individual contributors believe that effective knowledge sharing increases delivery velocity by at least 50%.

Management perspective

That number increases to 81% when surveying management.

The difference in views between individual contributors and managers indicates a varying awareness of how knowledge sharing affects delivery speed. Managers, with their broader perspective, might better understand the impact of effective knowledge sharing and documentation on the team as a whole.

Chapter 5

Developers struggle with documentation

✳

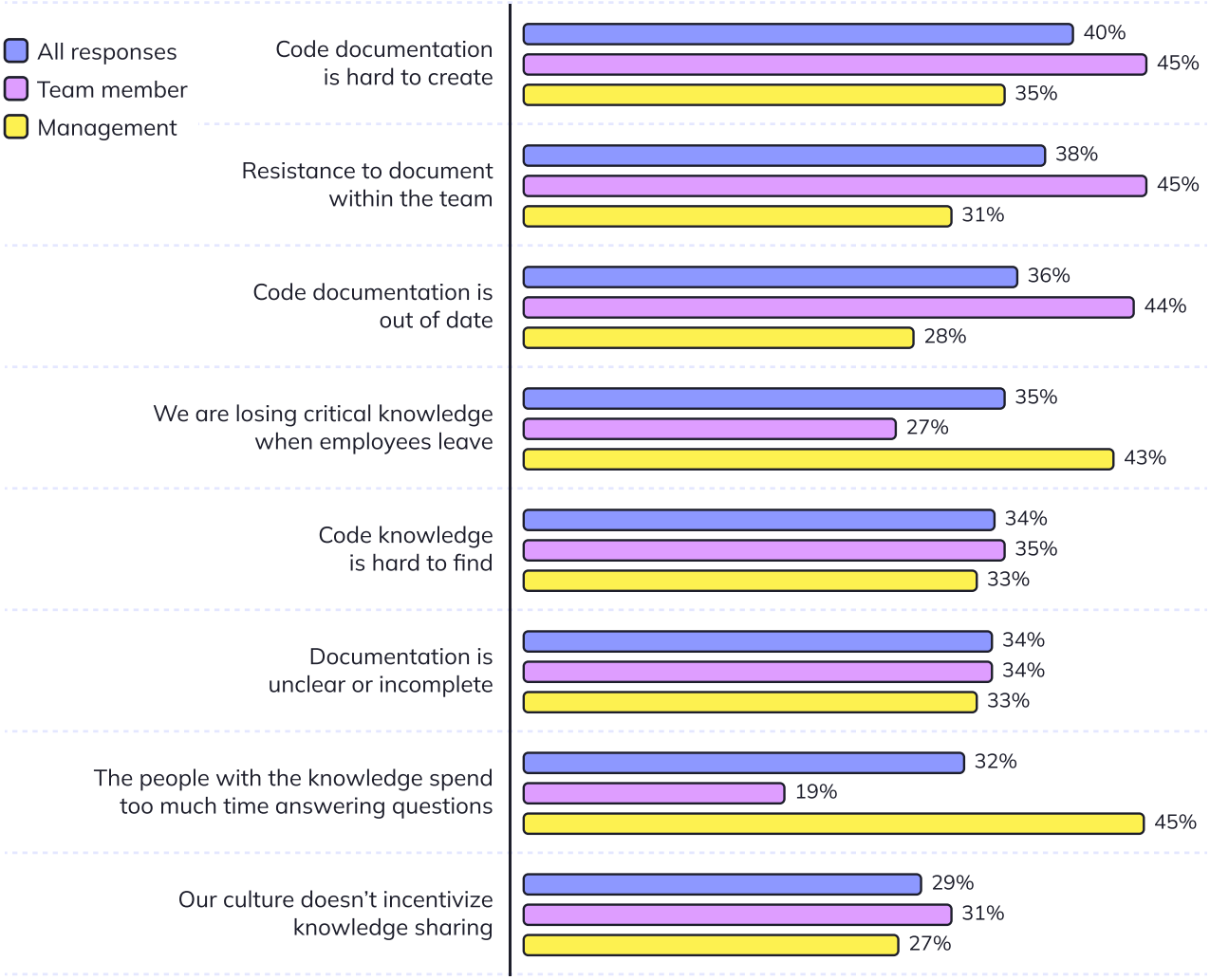
Managers worry about knowledge loss

Developers identified three main challenges they encounter daily regarding the sharing of code knowledge:

01. **Code documentation is hard to create (45%)**

02. **Resistance to documentation within the team (45%)**

03. **Code documentation is out of date (44%)**



Once again, there's contrasting viewpoints between developers and managers:

Developers' perspective

Developers find it hard to capture and use sources of technical knowledge.

Management perspective

Managers worry about losing knowledge when an employee leaves and how long it takes for developers to find answers to questions about code

Management should focus on the system that facilitates knowledge sharing. This includes addressing 'bus factor' and the amount of time senior engineers devote to answering questions. Addressing these issues, knowledge loss and reduced engineering capacity, is more critical than the commonly cited challenge of documentation.

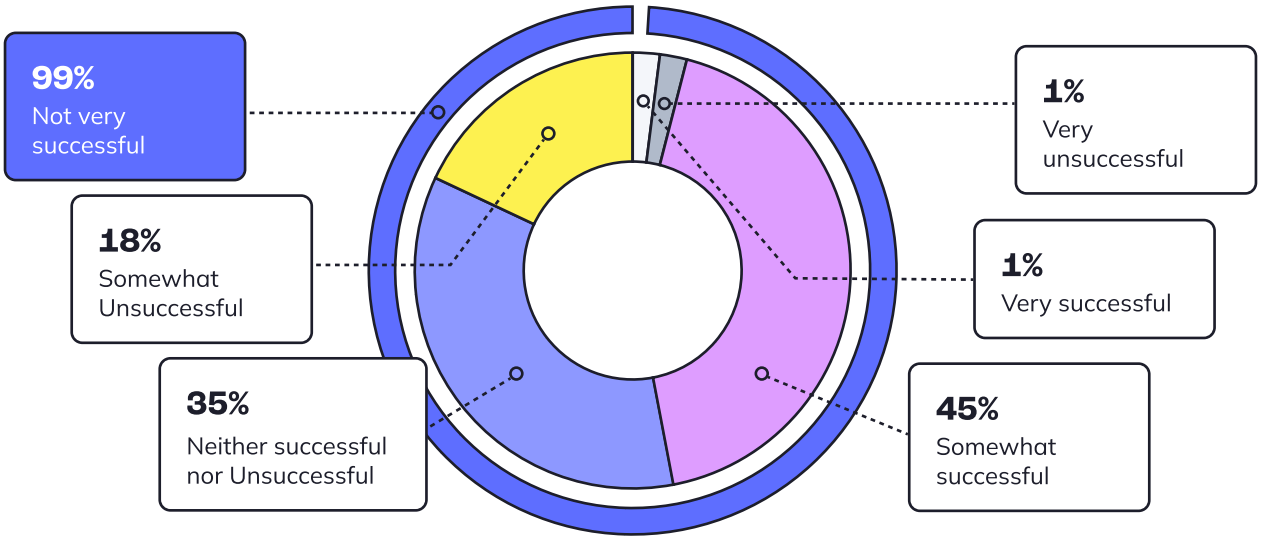
Top 3 management challenges:

01. **Losing critical knowledge when employees leave (45%)**
02. **The people with the knowledge spend too much time answering questions (43%)**
03. **Code documentation is hard to create (35%)**

Chapter 6

Devs doubt the efficiency of your knowledge sharing practices

Less than half (46%) of developers expressed confidence in their company's efficiency in sharing code knowledge - only 1% (!!!) said they are very successful.



This indicates that the majority of organizations face difficulties in efficiently sharing and maintaining vital code knowledge among their engineering teams. As AI takes on more coding tasks, the necessity for effective knowledge transfer becomes even more critical. Poor knowledge sharing not only slows down productivityok but also risks confusion over code ownership and understanding.

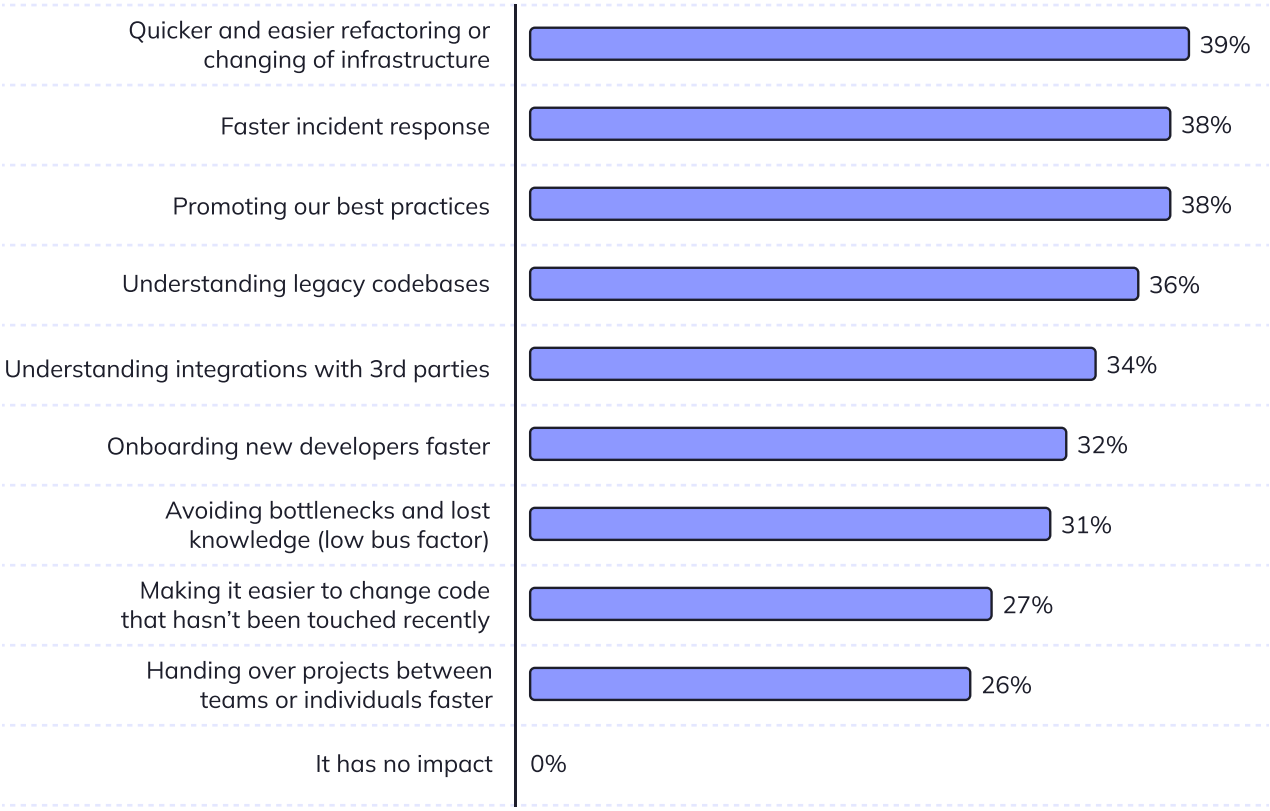
Chapter 7

Investment in knowledge sharing pays dividends

Engineering organizations identified the key advantages of success technical knowledge sharing, with the main benefits being:

01. **Quicker & easier refactoring or changing of infrastructure (39%)**
02. **Faster incident response (38%)**
03. **Promotion of best practices (38%)**

These data show an interesting trend: a margin of just 13% divides the most and least recognized benefits, highlighting the broad and diverse positive effects that effective code sharing can bring to an organization. Since each benefit was selected by at least 25% of respondents, it's evident that the strengths of solid knowledge sharing practices span several vital areas of organizational performance.



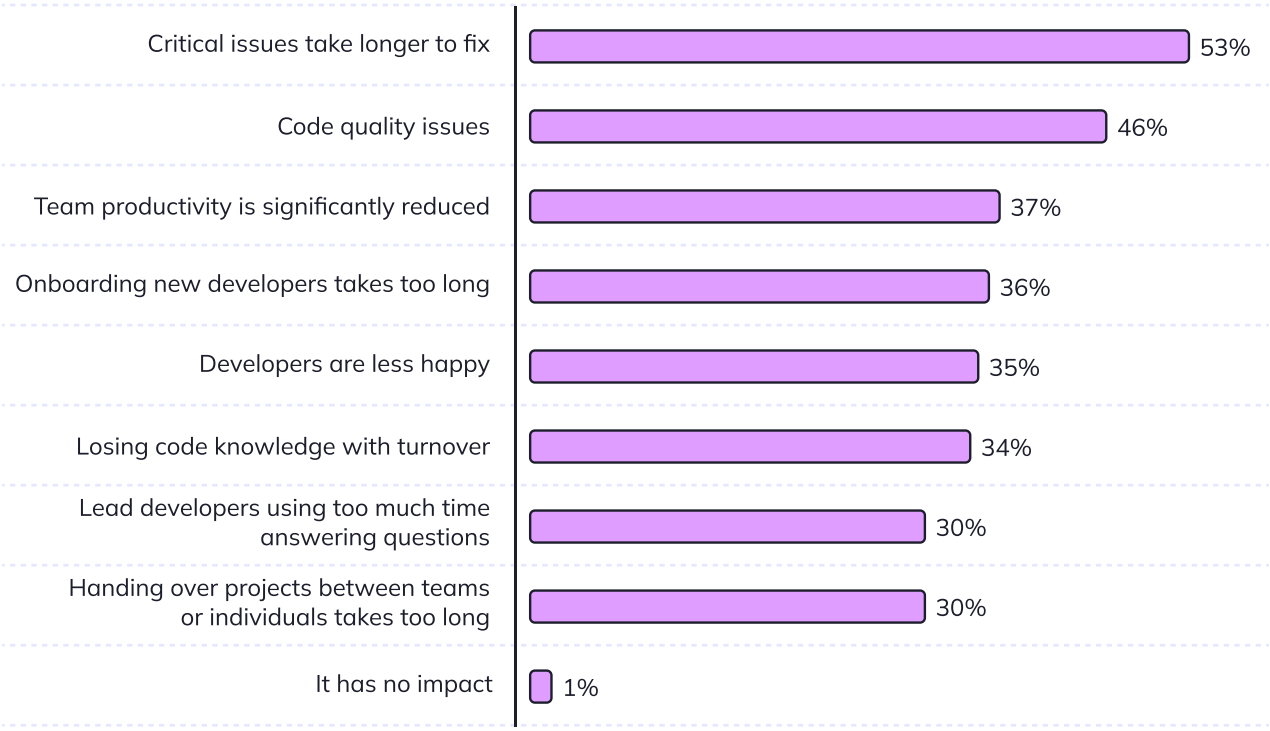
Chapter 8

The organizational toll of ineffective knowledge sharing

Nearly all (99%) of developers reported feeling the negative impact of insufficient or poor technical knowledge sharing within their organization. The most common issues cited were:

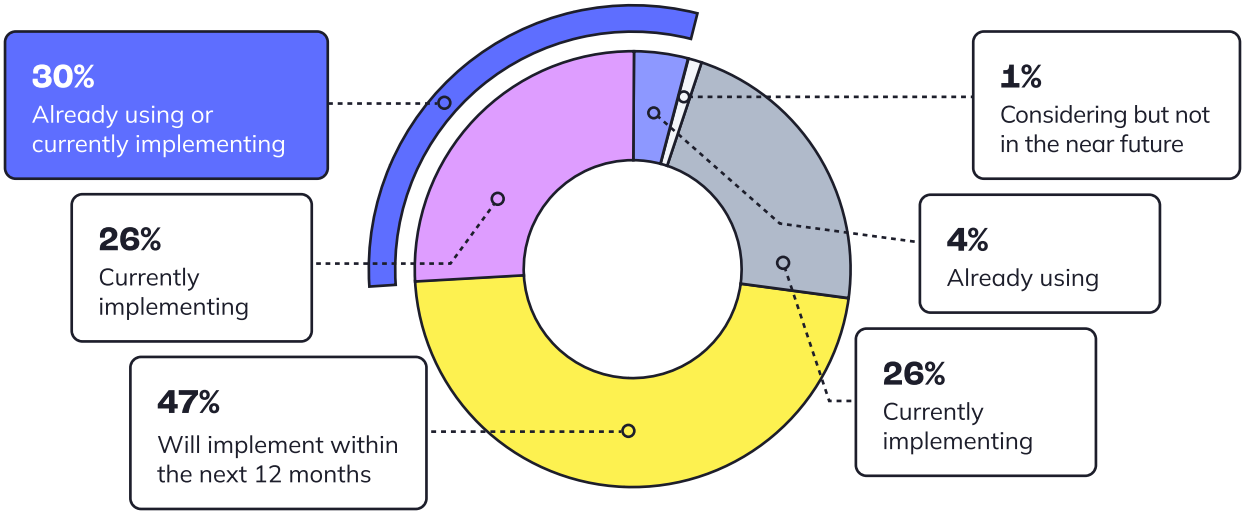
01. **Critical issues taking longer to fix (53%)**
02. **Issues in code quality (46%)**
03. **Significant reduction in team productivity (37%)**

These significant impacts highlight the critical need for effective knowledge sharing within organizations. Addressing these challenges by improving knowledge transfer can enhance efficiency, improve code quality, and boost team productivity, ultimately fostering a more productive and dynamic development environment.



Chapter 9

Everyone's doing it (AI, that is 😉)



Nearly ⅓ of companies surveyed are already using or starting to use AI tools to share knowledge about their code throughout their organization. While almost half (47%) aim to adopt these tools in the company year. A mere 1% has no plans to use AI soon.

This data is hardly surprising, especially considering the broader trend of AI tool adoption across the entire software development lifecycle. If you haven't started planning your AI strategy, there is no time like the present. As more and more code is generated with AI, the need for tools that help developers understand and interpret the nuances and particularities of your codebase becomes even more mission critical.

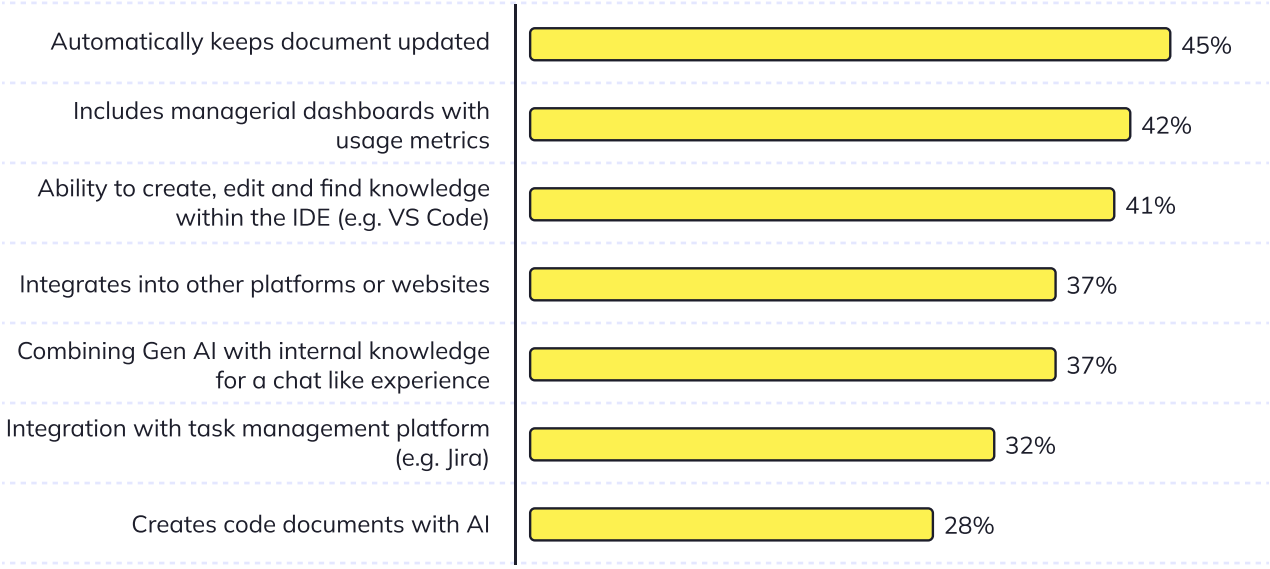
Evaluating an AI tool

The AI tool landscape continues to grow, with new advancements emerging daily. The integration of AI into the SDLC increasingly allows for higher levels of customization, making it essential to choose tools that can adapt to specific contexts and incorporate complex business and product logic seamlessly. A one size fits all approach doesn't work when it comes to AI – businesses must look for AI solutions that offer a comprehensive package, combining innovation, flexibility, and the ability to meet the unique needs of their operations for maximum effectiveness.

Chapter 10

What devs are looking for in a knowledge sharing tool

The results are clear: developers want tools that keep sources of technical knowledge up to date (45%), managerial dashboards with usage metrics (42%), and capabilities for creating, editing, and discovering technical knowledge within the IDE (41%).

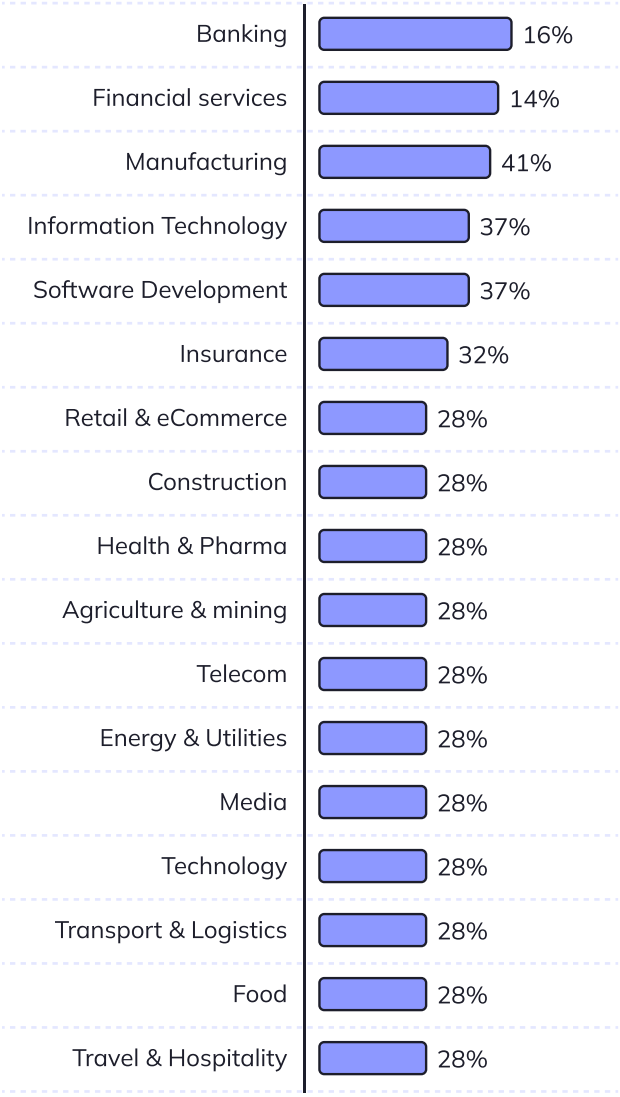


The emphasis on automatic updates highlights the need for tools that adapt and update in real-time, ensuring content remains current and accurate. Managerial dashboards with usage metrics reflect the need for better ways to measure the impact of knowledge sharing efforts. Finally, integrating these features within the IDE is seen as essential, making it easier for developers to access information without disrupting their workflow by switching tools.

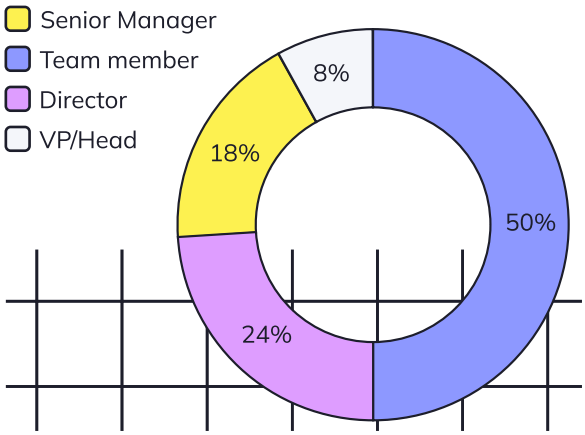
Chapter 11

Demographics

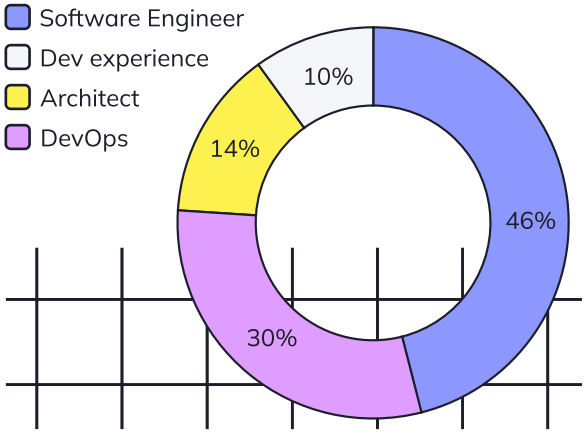
Industry:



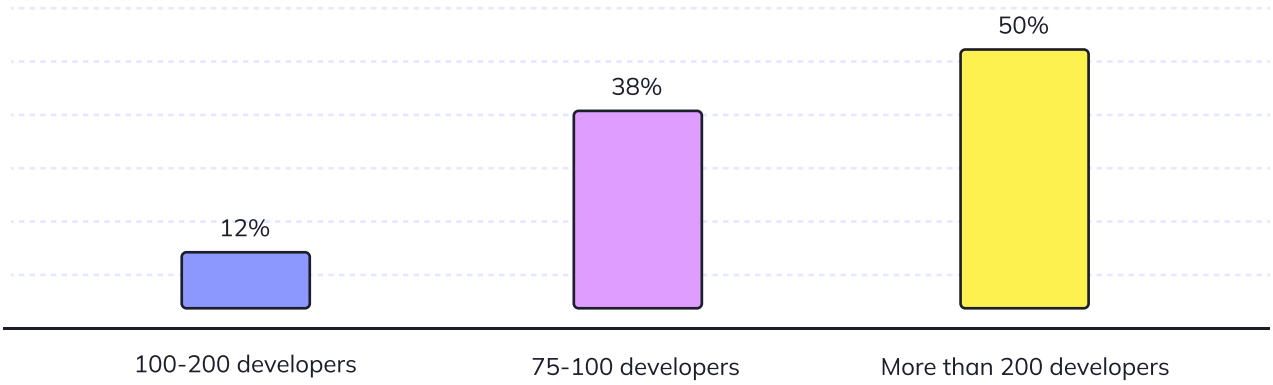
Seniority:



Title:



Number of developers in the company:



Chapter 12

Conclusion

The report highlights a critical but often overlooked aspect of software development: effective knowledge sharing. Regardless of whether a developer wrote the code or if it was generated using AI, managing and sharing knowledge is challenging yet essential, and many teams struggle to do it well – if they're doing it at all.

The solution lies in adopting AI-powered tools that integrate smoothly into the SDLC, catering specifically to an organization's unique technical needs. Engaging with developers to understand their challenges is vital in finding the right tool. The bottom line is clear: addressing knowledge sharing is urgent, and with AI, organizations have a powerful way to enhance their practices.

Chapter 13

Methodology

This report is based on a survey conducted online by Global Surveyz on behalf of Swimm throughout November 2023. It involved 200 full-time software engineers, as well as senior managers to VP-level personnel in engineering departments who work at companies with more than 75 developers across various industries. All respondents were residents of the United States at the time of the survey.

swimm

Swimm is an AI coding assistant that helps developers quickly understand big, complex codebases—and seamlessly captures knowledge to fill in any documentation gaps

Learn more about Swimm