GROWTH ACCELERATION PARTNERS

Survey Reveals GitHub Copilot Accelerates Coding Power, But Still Needs a Human Touch

Learn How Growth Acceleration Partners Boosts Coding Speed While Maintaining Architectural Integrity

Executive Summary:

A recent survey conducted by Growth Acceleration Partners (GAP) reveals GitHub Copilot has become a valuable tool for software engineers, offering productivity benefits while presenting some challenges.

The AI-powered coding assistant has demonstrated its ability to increase coding productivity, even for expert developers. The survey aimed to assess GitHub Copilot's impact on daily engineering activities, focusing on coding efficiency, code quality, innovation, knowledge sharing and employee satisfaction.

⊕ Positive Impact

In the survey, 80% of GAP engineers indicate GitHub Copilot has been beneficial in various ways. While it primarily improves coding speed it also suggests alternative, version-specific libraries, including MySQL, PDF parsing libraries and SignalR. Teams have used Copilot to learn new coding patterns, implement specific libraries and work with unfamiliar APIs. Other positive use cases include getting accurate code suggestions and explanations, autocompleting boilerplate code, and maintaining coding flow. Engineers called out how much smoother coding is when they didn't have to sift through Google, StackOverflow and Reddit results.

⊖ Limited Impacts

The survey responses indicate GitHub Copilot has had a limited impact on teams' approaches to code design and architecture at GAP. The majority of respondents (about 70%) report Copilot has not influenced their design or architectural decisions, and they use Copilot primarily as a tool to assist with specific coding tasks, provide suggestions, and offer detailed information within the context of the code they are writing. However, Copilot is generally not seen as a tool for high-level design or architecture. Copilot is not able to help design an entire project or system.

Report

GitHub Copilot has emerged as an essential tool for software engineers, offering substantial benefits while also presenting some challenges. The AI-powered coding assistant offers strategies to improve code security and consistency, helping even expert developers to more efficiently produce secure, high-quality software. Proponents say GitHub Copilot has the potential to foster better teamwork and peer-coding practices among engineering teams.

In 2024, Growth Acceleration Partners (GAP) conducted a survey to assess its impact on daily engineering activities. Specifically, the goal of the survey was to identify the tool adoption and contributions to daily engineering activities and its impact in specific areas, including:

- **Coding efficiency and productivity**
- **Better code quality**
- **Exploration of innovative technologies and solutions**
- **Knowledge sharing**
- **Employee satisfaction**

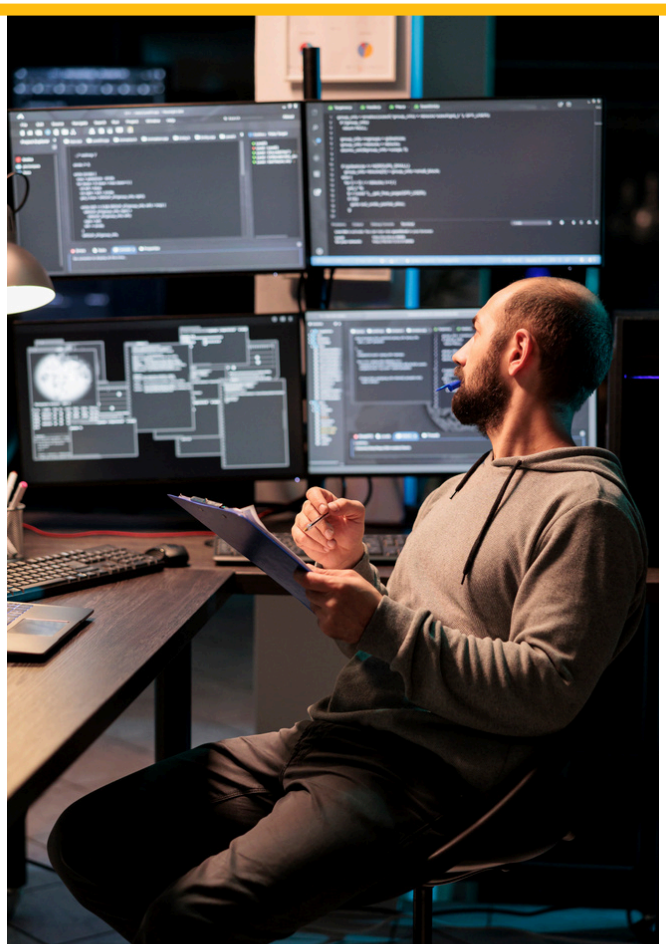
GAP's survey results — which focused on GAP teams working with clients that include Amadeus, Elekta, Saatva, nZero and YouDecide — revealed several key insights.

Overall, GitHub Copilot Business is well-received, with positive impact on coding efficiency and new feature discovery. However, there are areas for improvement, particularly in security, platform consistency and integration with various development environments.

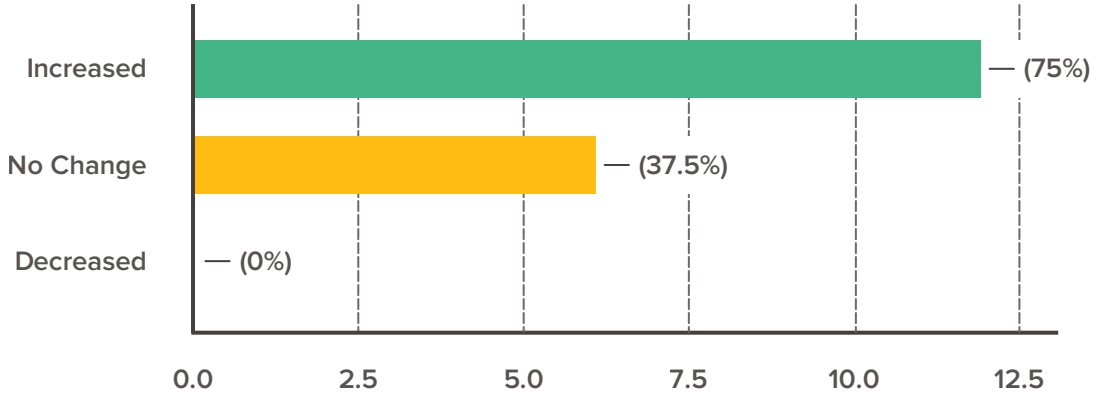
Coding Efficiency and Speed

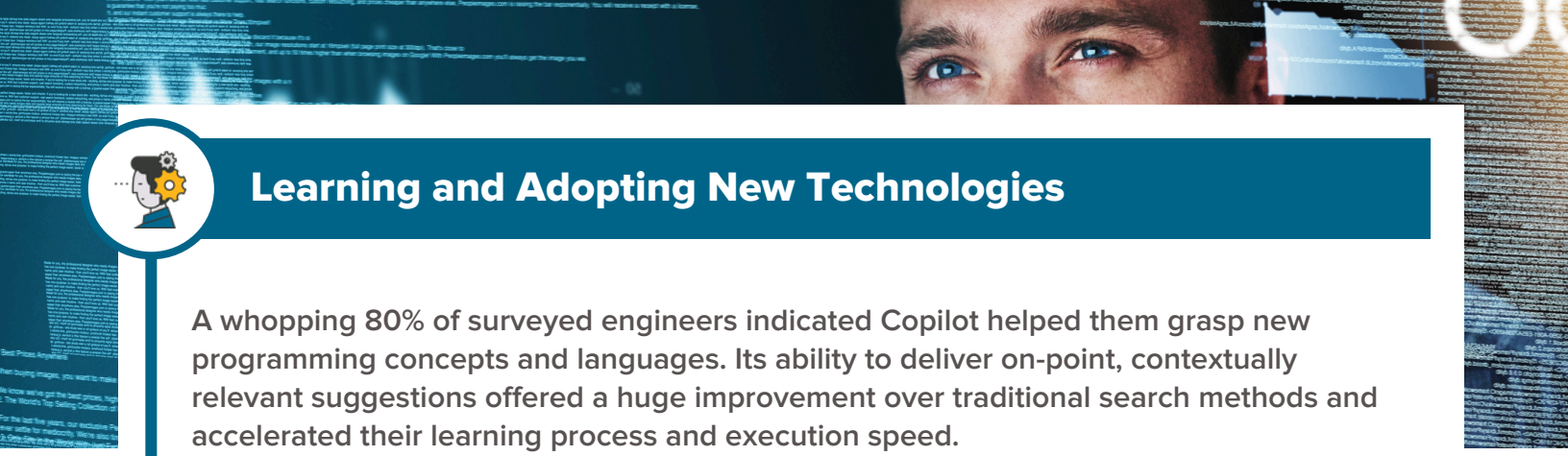
In GAP's survey, 75% of respondents reported increased development speed and efficiency. Engineers found GitHub Copilot valuable for inline code completion and new feature discovery. Additionally, they praised Copilot chat for providing useful context-specific answers, boosting overall productivity.

Copilot strengthens problem-solving by providing coding suggestions, fixing errors, and speeding up a user's understanding of new technologies and frameworks. Respondents said it is better for starting tasks and refactoring code. It guides in reasoning and tackling problems, proposes better approaches, and allows more focus on core problem-solving.



How has Copilot impacted your team's overall development speed?



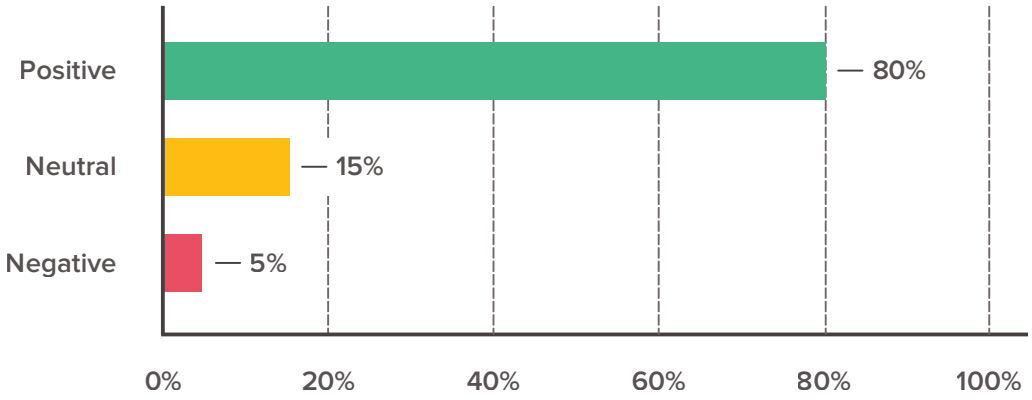


Learning and Adopting New Technologies

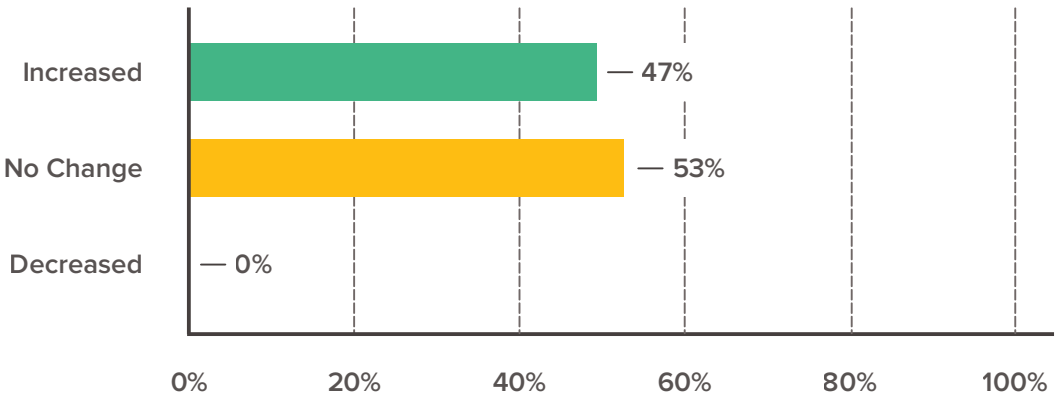
A whopping 80% of surveyed engineers indicated Copilot helped them grasp new programming concepts and languages. Its ability to deliver on-point, contextually relevant suggestions offered a huge improvement over traditional search methods and accelerated their learning process and execution speed.

GAP engineers indicate Copilot's live coding assistance, and detailed explanations and documentation links, make complex concepts easier to understand. By summarizing language features, explaining code patterns and offering practical examples, Copilot supports learning across multiple languages and tools. And this improves the overall coding experience and productivity for teams.

Usefulness of Copilot's Suggestions in Learning New Concepts



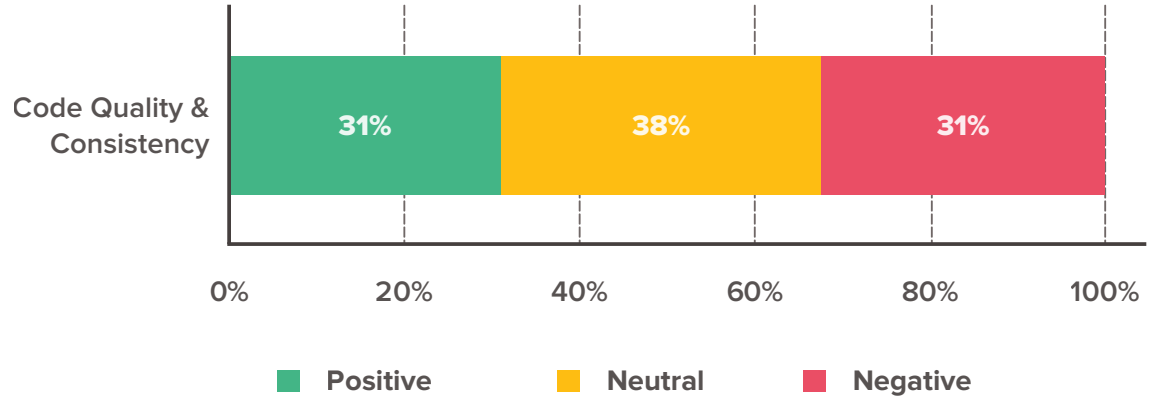
Impact of Copilot on Independent Problem-Solving Ability



Code Quality and Consistency

Opinions on code quality and consistency were mixed. Overall, while Copilot is appreciated for its speed and support in repetitive coding tasks, its impact on code quality is seen as minimal, with room for improvement in its suggestions. For example, Copilot sometimes suggests code that, although technically correct, can introduce issues with error handling, or reduce readability or even introduce performance bottlenecks..

Code Quality and Consistency



In the survey, 31% of respondents noted positive impacts, citing better standardization and reuse of existing code. They also referenced benefits in code optimization, autocompletion and learning new coding strategies. However, 38% reported no significant impact.

Internal processes like code reviews are still critical for ensuring code quality. However, Copilot still contributes by accelerating coding and handling mundane tasks, freeing up developers to focus on more strategic tasks such as design and architecture.

Another 31% expressed concerns about the quality and consistency of suggestions. This underscores the need for careful review and validation of Copilot-generated code. The negative responses implies Copilot does not always improve code quality, but serves as a resource for speeding up development. Some users feel the generated code is not always optimal and requires further refinement, and specific issues were mentioned in the context of using interpreted languages like Dart. We assume other languages might have similar challenges.

Security Awareness

GitHub Copilot had a limited impact on security awareness, with only 15% of respondents noting increased awareness in specific areas like secret handling. The majority (85%) continued to rely more on existing security tools, indicating a need for further integration of security best practices within Copilot's suggestions.

While a majority of GAP engineers feel reassured about code ownership and intellectual property with GitHub Copilot, there are concerns among a subset of users. There was apprehension about whether Copilot suggestions might expose previously written sensitive information, but initial concerns were mitigated by client approval and understanding that code wouldn't be shared publicly. Some team members also raised concerns about handling sensitive information, such as secrets in non-committed files.

Additional Positive Impacts

Helper for New Technologies: Copilot boosts confidence when working with new languages or libraries, easing the learning curve and assisting the development process.

Focused Coding: Copilot keeps developers “in the flow” by providing answers and suggestions within the editor, eliminating the need to interrupt their work to search external sources.

Suggesting Alternative Libraries and Approaches: Copilot keeps developers “in the flow” by providing answers and suggestions within the editor, eliminating the need to interrupt their work to search external sources.

Additional Areas of Limited Impact

Granular Assistance: The tool is mainly used for solving specific scenarios and providing granular code suggestions, rather than guiding the overall design or architecture of projects.

Outdated Elements: In some cases, Copilot's outdated suggestions limit its usefulness in influencing architectural decisions. For example, while Copilot suggestions for Spring Boot code may look fine, some proposed annotations are commonly used with older Spring Boot versions known for their lack of readability and specificity.

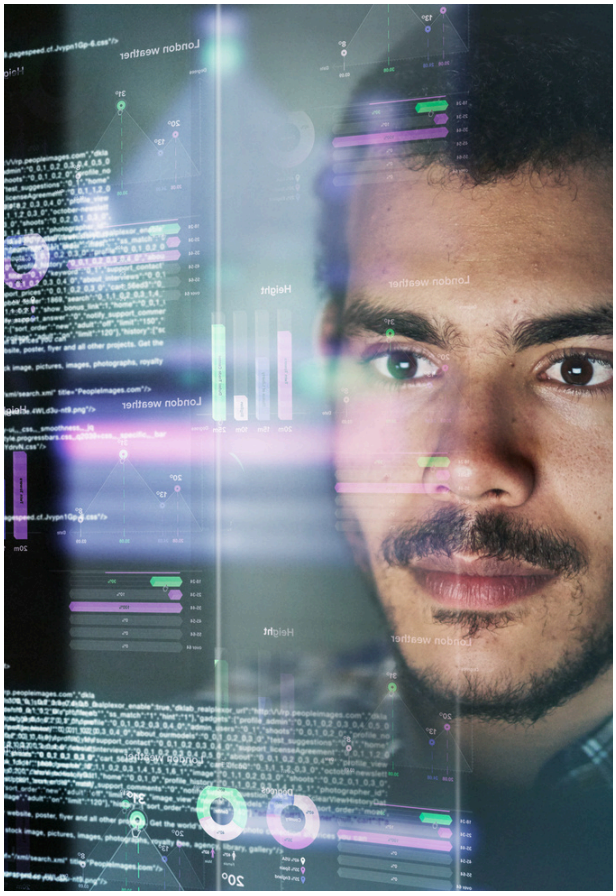
Best Practices: Only 7% of teams actively discussed best practices related to Copilot usage, which suggests an opportunity for organizations to foster more conversations around Copilot's usage.

Significant Human Oversight Required

It's important to note GAP's survey responses suggest that, while GitHub Copilot is appreciated for its ability to speed up coding, assist with routine tasks, generate initial options and optimize small functions, it requires A LOT of human oversight and refinement to ensure high-quality output.

Interpreted Languages

Copilot hiccuped when asked for suggestions when using interpreted languages like Python and Dart. The code suggestions often required major rewriting.



Contextual Testing

Copilot provides flaky and unreliable unit tests when more context is needed, sometimes deviating from official documentation. In general, the team has so far not been able to use Copilot to write test cases but this is an area we hope will improve.

Outdated Frameworks

For several frameworks, including Springboot and Flutter, Copilot gave outdated responses, and impeded workflow when using newer versions.

Broader Scope Tasks

Copilot struggles with tasks beyond the current file's scope, such as project-wide or architectural questions, and hypothetical planning requests.

What's Next for GAP Engineers Using GitHub Copilot?

GAP is prioritizing education and clear guidelines to help build trust in Copilot. In response to the survey feedback, the company is implementing some recommendations to maximize Copilot's effectiveness:

1. Provide GAPsters with additional training materials, including comprehensive guides on Copilot usage, tips, tricks and best practices to boost productivity.

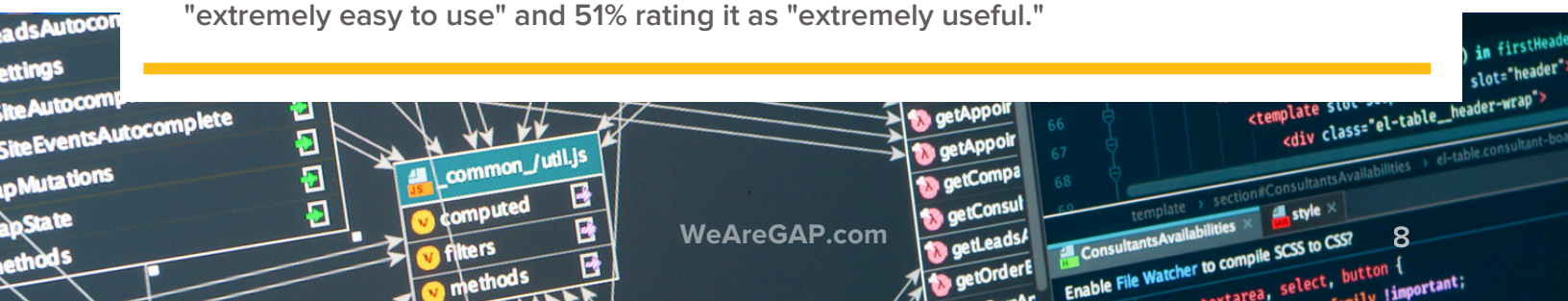
2. Focus on improving the accuracy of Copilot's suggestions and reducing rework, encouraging practices that ensure reliable suggestions while maintaining code quality.

3. Explore broader use cases for Copilot, such as generating test scripts, improving team communication, knowledge sharing and collaboration, and identifying code vulnerabilities.

While formal discussions on using Copilot to improve security are currently limited, GAP engineers recognize the potential in this area. For example, Copilot could be trained to identify common security vulnerabilities in existing code, such as SQL injection or cross site scripting flaws, hardcoded credentials, cross-site scripting, insecure storage of sensitive data, and path traversal attacks. Also, Copilot could help prevent new vulnerabilities by suggesting secure coding patterns and flagging potential issues in real-time as code is written. By establishing best practices and providing training on security use cases, GAP could improve its code quality.

These findings align with other recent studies on GitHub Copilot's impact. For instance, a collaboration between GitHub and Accenture found that 67% of developers used Copilot at least 5 days per week, with an average usage frequency of 3.4 days per week. The study also revealed high adoption rates, with 81.4% of developers installing the Copilot IDE extension on the same day they received a license.

Furthermore, 96% of those who installed the extension started receiving and accepting suggestions on the same day, indicating ease of use and quick integration into workflows. This is supported by survey results showing 43% of users finding Copilot "extremely easy to use" and 51% rating it as "extremely useful."



By implementing the recommended actions and addressing the challenges identified in the survey, Growth Acceleration Partners believes organizations can enhance the overall effectiveness and satisfaction of using GitHub Copilot within their development teams, making it an indispensable tool in their engineering toolbox.

TECHNICAL SPECIFICATIONS:

GAP has implemented GitHub Copilot Enterprise across its engineering teams. The tool functions as an IDE extension, meaning it operates within the user's familiar coding environment and uses the team's existing security and compliance infrastructure. GAP used this approach to simplify adoption and ensure teams could easily follow organizational protocols.

To find out more, please visit WeAreGAP.com

company/growth-acceleration-partners/ @GrowthAccelerationPartners @GAPapps