Anatomizing the Developer Shortage

February 07, 2023

Research Analyst – Digital Experience and Automation

Coming from a background in conducting original ethnographic research, Mary-Kate brings a humanities lens to the technology she writes about. She’s passionate about using her background in primary and secondary research to bring innovative solutions to clients in both the digital experience and automation spaces. Outside of work, Mary-Kate enjoys both traveling and hiking.

Submitted by Mary-Kate Sloper on 7, Feb 2023
Anatomizing the Developer Shortage

Organizations looking for citizen developers to fill the talent gap of professional developers are barking up the wrong tree. In this article, we unpack the key reasons behind the talent shortage and share best practices for organizations to attract and retain developers.

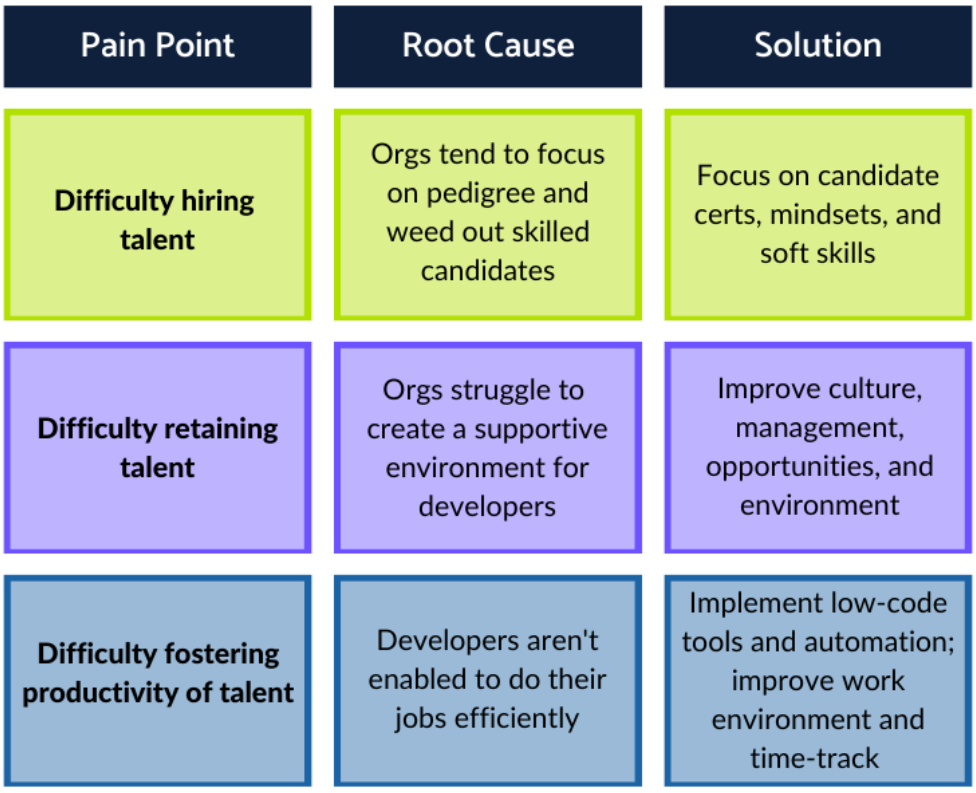
Many organizations have been looking to citizen developers as a solution to the talent shortage of professional developers. But they'd be better off focusing on improving their culture to attract and enable top talent.

Industry leaders have positioned citizen developers as a solution to the shortage of professional developers, but while citizen development has its benefits, it also has its risks. Ultimately, citizen development could simply be a Band-Aid solution that doesn't adequately address cultural issues.

IDC predicts that "The global shortage of full-time developers will increase from 1.4 million in 2021 to 4.0 million in 2025." CodeSubmit says of the long-term predicament, "The US Labor Department estimates that the global shortage of software engineers may reach 85.2 million by 2030."

Organizations are increasingly looking to digital transformation to drive competitive advantage, and IT has become a key differentiator. Teams of smart, productive developers are invaluable to the organizations they create custom applications for. As such, organizations need to truly understand the talent shortage problem and change to a more humanistic approach to hiring, retaining, and enabling developers.

Tom DeMarco and Timothy Lister published their classic book Peopleware in 1987—in it they write, “The major problems of systems work are not so much technological as sociological in nature.” (xv) Organizations should focus on sociological solutions for attracting and retaining developers and maximizing their productivity.



Source: Stratascale 2023

1. Difficulty hiring talent

a. Overlooking talent — technical skills

A key component of the talent shortage is an avoidable problem: viable candidates are often overlooked. Forbes says of developers, “The talent is out there. We just need to find it.”

Recruiting industry leader Hung Lee is quoted: “Prestige hiring, or credentialism, as it’s sometimes called, drastically reduces the total addressable market of candidates, and is likely one of the primary reasons for the ‘tech talent shortage’. We’re making false-positive decisions by hiring developers with name-brand employers on the CV, and we’re making false-negative decisions by rejecting developers who don’t.”

Stratascale’s Director of Technology Development, Marc Cantelmo, explains: “It’s no different than if you go to the market and you see something on the shelf. There's a generic bottle of something, and then a brand-name bottle. Nine times out of 10, the generic is just as good. Money doesn’t determine value — you have to examine what’s in the bottle to understand the true value.”

Cantelmo says of his approach to hiring developers: “I don’t even look at resumes anymore. To be completely blunt, they're worthless, and with college degrees it’s pretty much the same thing. They really don't have any merit.” Instead, Cantelmo looks for “someone who has open-source projects on GitHub or has their own Raspberry Pi at home where they’re working on things.”

Cantelmo also mentions there’s a lot of noise around various certifications, but that candidates with the Cisco Certified DevNet Professional certification stand out: “It’s brutal. They literally give you a spec and you have to build it. You go away and then come back the next day and they broke it. You have to find out what they broke within a certain amount of time systematically — the way they taught. You have to show your work a certain way.”

Organizations should therefore be looking at developers who see software development as a passion and hobby, and who have proven their skills through a challenging certification, rather than judging them based on “name brands” (or lack thereof) on their resume.

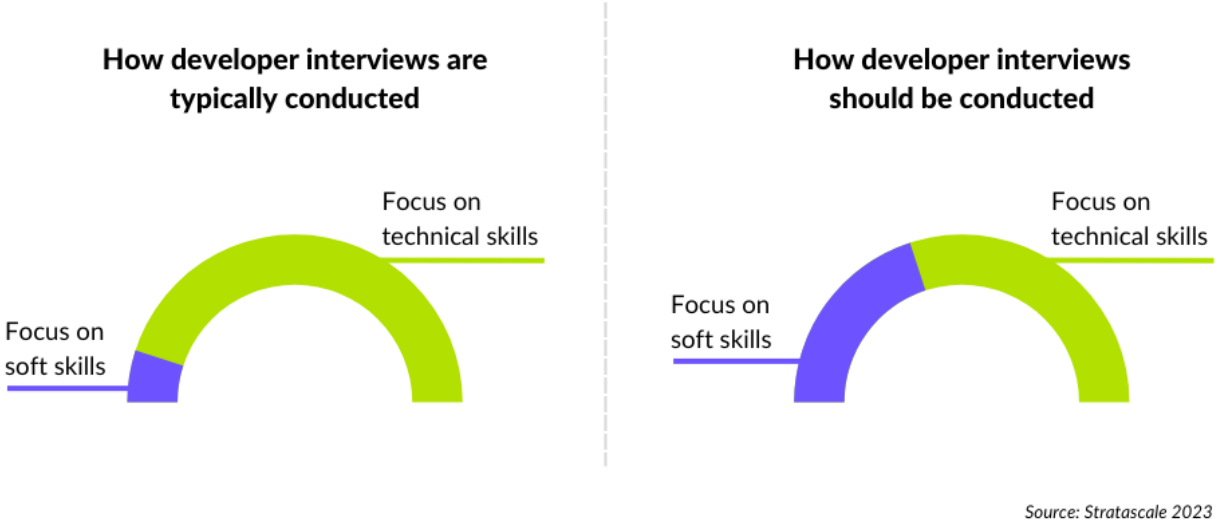
b. Overlooking talent — soft skills

Beyond any hard or technical skills, being a successful developer and team member also requires softer skills which are much harder to assess.

In Peopleware, DeMarco and Lister describe a project staff member whose manager didn't "quite see what she adds to a project; she's not a great developer or tester or much of anything." (10) But on closer investigation, this staff member had a stellar track record: "During her 12 years at the company, the woman in question had never worked on a project that had been anything other than a huge success. It wasn't obvious what she was adding, but projects always succeeded when she was around." (10)

DeMarco and Lister go on to define this type of worker as a catalyst: "Teams naturally jelled better when she was there. She helped people communicate with each other and get along. Projects were more fun when she was part of them… The catalyst is important because the project is always in a state of flux. Someone who can help a project jell is worth two people who just do work." (10-11)

Organizations might overlook skilled developers like Cantelmo described by myopically only looking at resumes, but they also might needlessly weed out catalysts by conducting interviews focused solely on technical skills.



Organizations should look for developers who can work well on a team, can both listen and communicate skillfully, can take the lead when needed, and problem-solve, whether the issue is people- or technology-related. Even the technical portion of the interview shouldn't be too focused on hard skills — coding languages are always evolving, so a true testament to a developer's skills is their ability to adapt and think critically in their approach to coding, regardless of the language.

2. Difficulty retaining talent

In addition to the barriers professional developers face in the hiring process, they're also electing to leave their positions. The Great Resignation has not exempted developers from looking for work elsewhere or leaving IT altogether. CodeSubmit reports that "72% of US IT workers say they might quit their jobs in the next 12 months."

The direct cost of replacing an individual employee “can range from one-half to two times the employee's annual salary — and that’s a conservative estimate,” according to Gallup. But turnover’s impact extends beyond the direct costs of replacing each worker, as Demarco and Lister point out: “In companies with high turnover, people tend toward a destructively short-term viewpoint, because they know they just aren’t going to be there very long.” (118)

Employees with a short-term viewpoint might exhibit less motivation (knowing they won’t stay long enough to see a promotion), which leads to lower productivity and lower quality output. They might also be less likely to form deep connections with coworkers and to come up with proactive, novel ideas that benefit the organization they’re in if they don’t see a future there. These employees might also be more likely to cause technical debt; although they might get their projects over the finish line during their tenure, they’re only leaving a mess for operations and product managers to clean up later.

a. Culture of enjoyment

Organizations need to foster a culture of enjoyment by giving developers the opportunity to have genuine fun with their teammates.

In Peopleware’s final section, entitled “It’s Supposed to Be Fun to Work Here,” DeMarco and Lister give some suggestions for promoting a culture that makes work more exciting for developers. One suggestion they give is called “Coding War Games,” where teams compete in a 24-hour Project Tournament. DeMarco and Lister describe this as a “sometimes raucous, competitive, no-lose experience” that, when pulled off successfully, developers describe as “the most exciting and enjoyable experience of their entire careers.” (226-227)

To actualize a culture of enjoyment, organizations need to identify a point person to take charge in improving developers’ experiences. Organizations also need to identify a program manager for the events they decide to host.

Whether through holding a tournament like Peopleware’s Coding War Games or another creative avenue (like hackathons) for developers to hone their skills together, organizations need to make an effort to make work more enjoyable for developers. Even the most passionate developers can face burnout if they’re not given the opportunity to have fun on the job.

b. Managing to motivate

A high salary, generous employee benefits, and access to helpful tools alone cannot solve what is inherently a sociological issue. Cantelmo summarized the reason developers are leaving in droves: “Their management doesn't do their job.”

In the spirit of football season, Cantelmo gives the following analogy: “If you look at the teams talent-wise, there is a microscopic difference between the very best and the very worst team in the NFL. So, what’s the factor? Well, it’s the coach. The coach has to come up with a plan and the value of it. But most importantly, the coach has to be the one that’s the motivator.” Cantelmo explains that developers look to managers to be effective motivators, to “make them feel good about what they’re doing, how they’re doing it, what their value is to the company.”

DeMarco and Lister say of managers in Peopleware: “The best success is one in which there is no evident management, in which the team works as a genial aggregation of peers. The best boss is the one who can manage this over and over again without the team members knowing they’ve been ‘managed.’ These bosses are viewed by their peers as just lucky. Everything seems to break right for them. They get a fired-up team of people, the project comes together quickly, and everyone stays enthusiastic through the end. These managers never break into a sweat. It looks so easy that no one can believe they are managing at all.” (160)

In addition to motivating their developers and aligning them towards a common goal, managers need to allow their employees to “jell” as a team, and a key driver of this is trust. Managers must grant their employees autonomy and responsibility in their areas of strength; in turn, the team is “making sure that the trust that’s been placed in them is rewarded.” (163)

Organizations need to look beyond just talented developers when considering someone for management; they might also look at project managers, product managers, and enterprise architects to lead their dev teams. In fact, some of the best dev managers may have no direct dev experience. While dev managers should possess an understanding of application and solution architecture, they also need to demonstrate other equally important qualities such as leadership, listening, communication, delegation, and organization skills.

c. Developing managers within

Organizations save money, save time, and drive retention when they promote within.

Forbes references a study on external versus internal hires: “The external hires made 18% more than the internal promotes in the same jobs. In addition to scoring worse on performance reviews, external hires were 61% more likely to be fired from their new jobs than were those who had been promoted from within the firm.”

DeMarco and Lister mention that the ramp-up time for an external candidate can be six months but is more realistically “two years to bring a new worker up to speed” for more complex work. (Peopleware, 129) Internal candidates are already familiar with the ins and outs of their organization’s culture and will take far less time acclimating themselves to a new role.

Opportunities for promotion alone encourage the promoted employee to stay, but internal promotion also encourages the developers that the employee manages to stay. Lasting social relationships are critical to retention — people want to work with and for people they trust and respect.

d. Professional development

In addition to promoting developers into management positions to encourage retention, organizations should enable their developers to upskill through conferences, certifications, and other educational opportunities.

DeMarco and Lister especially encourage travel opportunities for employees: “Perhaps this is a sad comment on the dismal corporate workplace, but everybody relishes at a chance to get out of the office.” (Peopleware, 228)

Offering the chance to travel to a conference or seminar can combat the monotony that comes with development work. But investing in your employees this way also sends a clear message that you consider them and their future at your organization worthy of investment.

e. Remote work

Many employees, software developers included, are now using the ability to work remotely as a key factor in their decision to work for an organization. Turing states: “66% of software developers now want to work from home full-time,” and “84% of

employees believe that working in remote software engineer jobs long-term will make them happier, with many prepared to take a wage reduction to do so.”

Remote work can not only drive happiness and retention in developers, but it can also be more conducive to productivity. ZDNet refers to a recent study on developers: “68% said they are able to get more meaningful work done while working remotely or from home, compared to 32% who said they are more productive in an office environment.”

Openness to remote developers also expands the candidate pool exponentially from just local developers or developers looking to move to the area. If asked, most managers of the highest performing dev teams would take a remote, high-performing employee over a mediocre employee with lots of face time. Therefore, flexibility for remote work should be part of the equation when looking at retaining talented developers.

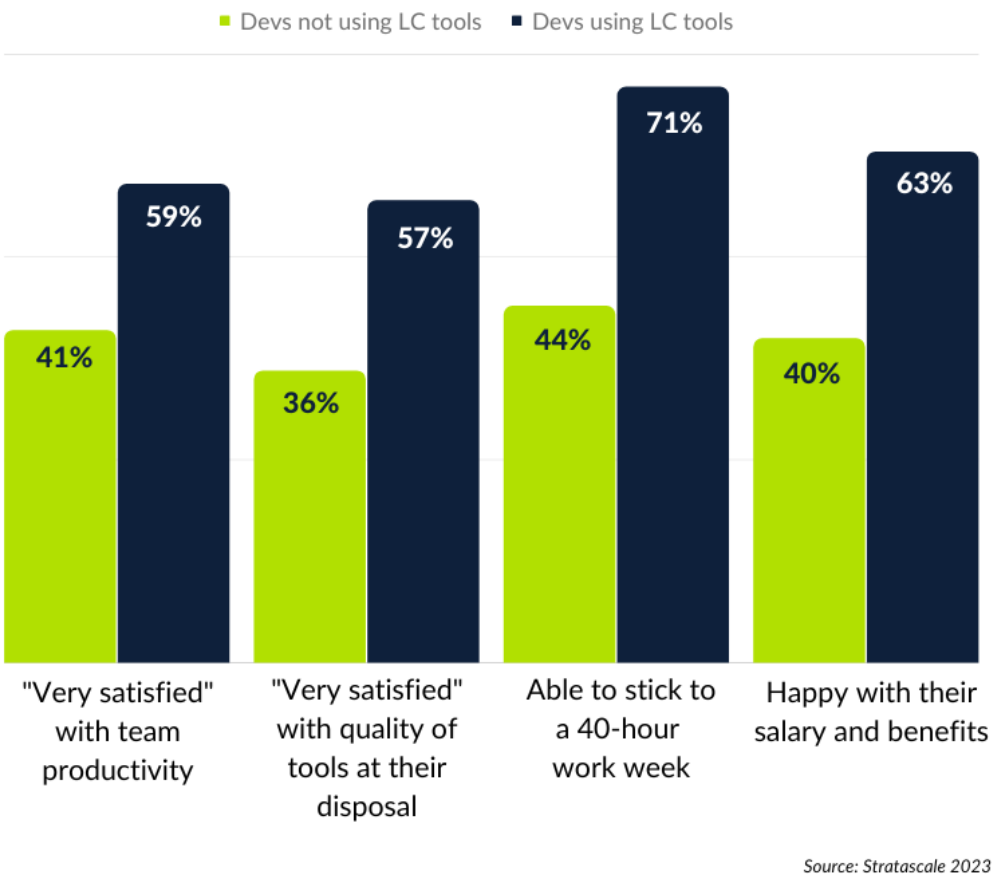
3. Difficulty fostering productivity of existing talent

It’s no secret that software development is challenging, demanding work, but there are some steps that organizations can take to make developers’ lives easier.

a. Low-code platforms

Even though technology alone cannot solve the sociological problem of the developer shortage, it can alleviate some of the pressure. Cantelmo explains how low-code tools can enable professional developers to execute their work more efficiently: “What you really want is to be able to expand the productivity of the resources that are good in those areas exponentially.”

ZDNet refers to a recent study conducted by OutSystems, who surveyed 860 developers to compare those who use low-code tools versus those who don’t. As summarized in the chart below, the study found that developers using low-code tools were more satisfied with both productivity and compensation:



The most prominent difference lies in the “able to stick to a 40-hour work week” category — 71% of the surveyed developers using low-code tools were able to do this versus only 44% of the surveyed developers not using low-code tools. Employees are increasingly using work-life balance as a metric of job satisfaction, and low-code tools can help drive satisfaction and reduce turnover.

b. Traditional development efficiencies

Low-code platforms are not the only thing that can make developers’ lives easier — other strategies can make traditional development more efficient as well. Developers can now access things like a rich set of components from the open-source community and cloud platforms with self-service at the click of a button.

Organizations should also keep their own repositories of reusable components so developers can maximize their productivity with every project.

c. Automation

Several stages of the software development lifecycle can be automated. From gathering requirements, writing the code itself, and deploying, to testing and optimizing, automation can help make professional developers more efficient.

Organizations should consistently make up-front investments to automate frequently performed, time-consuming tasks. On the other hand, they must strike a balance: not everything that can be automated should be, as the cost of building and maintaining the automation may not always justify the benefits.

d. Work environment

A work environment fraught with distractions and interruptions can impede even the most efficient employees.

In their chapter “You Never Get Anything Done around Here between 9 and 5,” DeMarco and Lister give several statements like the following: “The office is a zoo all day, but by about 6 P.M., things have quieted down and you can really accomplish something.” (Peopleware, 41)

The authors also give an example where one woman called out sick just to have a day to finish her work, distraction-free at home. However, working from home does not always guarantee a distraction-free environment.

Whether in-person or remote, a “quick” drop-in, five-minute call, or IM exchange can easily cost a developer 30 to 60 minutes in productivity when they then must return to their work, refocus, and re-enter flow. Organizations should create a work environment that allows developers and other knowledge workers the necessary hours-long focus time to enter flow and do “deep work.” In an in-person office setting that might look like doors on offices or cubicles instead of open-concept layouts, or in a remote work setting that might look like instilling a culture where team members respect a “do not disturb” presence from their peers.

e. Time management

Parkinson’s Law states that work expands to fill the time allocated for its completion — DeMarco and Lister offer their own twist on the adage: “Organizational busy work tends to expand to fill the working day.” (Peopleware, 29)

“Organizational busy work” can not only interfere with developers completing the tasks that bring their organization the most value, but it can also cause frustration and impact retention as well. For example, meetings with talking points that could’ve been communicated via email both interfere with productivity and can be unnecessarily draining.

In The Effective Executive, Peter Drucker encourages readers to track their time to see how much time actually goes into productive, value-added activities: “The effective executive knows that to manage [their] time, [they] first [have] to know where it actually goes.” (27) Organizations should evaluate how much time their developers spend doing organizational busywork and cut it back accordingly.

Takeaways

Anyone in software development would benefit from reading Peopleware, or at the very least remembering its premise, which is worth restating: “The major problems of systems work are not so much technological as sociological in nature.” (xv)

Difficulty attracting and retaining IT professionals cannot be solved by throwing citizen developers into the mix. Organizations must do everything in their power to attract talented individuals and keep them happy, because professional developers are vital to driving digital transformation.