# The future of dev in a disconnected world


### How to shift from “job done” to “value realized”:

- Not everyone will be working on the same thing at the same time, or even within the same tools. Make sure everyone is aligned with a clear vision on what you’re aiming to accomplish.
- Keep cross-functional teams on the same page with a way collaborate and communicate effectively. This collaboration must be visible, accessible, and consistently available to all members of the team.
- Emphasize quality from end-to-end with clear quality metrics and determine how you’ll test to ensure these measures are met during the development lifecycle.

## The future of dev in a disconnected world

The nature of software development has changed over recent years. With DevOps, CI/CD, Agile development, multithreading application development, and remote teams, software development has achieved greater delivery velocity. But quick delivery is only one part of the puzzle.

Useful applications require high-quality software that solves business problems. Modules that are delivered quickly but require rework defeat the purpose of DevOps. Organizations need to not only implement new tools and techniques to reach maximum value, but also to ensure everyone has access to the same information.

## The shift from “job done” to “value realized”

Strategic alignment is more important than ever. Since work is accomplished by teams of developers, often in separate locations, the challenge of keeping everyone on the same page is even more critical. Everyone needs to row in the same direction and to the same beat.

Most DevOps tooling focuses on what work needs to be done but not on why it is being done. When software teams don’t understand why they are doing a task, or how their work integrates with that of other team members, they may not focus on the right thing at the right time. Even more importantly, they may develop software that, while functionally correct, only solves a subset of the business needs it was supposed to satisfy.

**Change is a constant in today's business environment. Development tools that keep pace with that change and keep teams in the loop are essential.**

Consider a payment application. The intent is to obtain payment for the product and have the customer receive what they intended to order. Suppose the developer chooses to create a simple payment screen without allowing the customer to review the order. In that case, the code may satisfy the functional objective of allowing payment while not satisfying the customer's actual need to verify their order and make a payment on a single screen. When customers don't have a final opportunity to catch and correct potential mistakes in their orders, it can lead to returns and increased costs.

Keeping dev teams up to date on the whys of each project requires creating and maintaining visibility to objectives and key performance indicators (KPIs), including how they align with the overall strategy. Dashboards that show progress on module goals and business goals help teams focus on what is essential and ensure they deliver business value.

Agility is also key to DevOp's effectiveness. Change is a constant in today's business environment. Development tools that keep pace with that change and keep teams in the loop are essential. When team members communicate for developers to incorporate changes as early as possible in the development lifecycle, it minimizes the cost of making these changes and improves the overall quality of the finished product. Development alone does not incur the cost of changes. The cost also increases with delay in achieving the software's business value.

## Focus on integrating and automating

While it has always been essential to keep IT and business teams on the same page, the modern development environment has increased its importance. More detail on what the business requires and more coordination with and across teams is necessary. Cross-functional teams go a long way to creating higher quality software products, but only when they collaborate and communicate effectively.

This collaboration must be visible, accessible, and consistently available to all members of the team. Important information and updates must be seen by everyone, rather than buried in a blizzard of email, text messages, or meeting notes.

Collaboration demands a single source of the truth that brings information together without consuming excess time, manual duplication, and data entry. It must be easy for team members to find the information they need.

Development teams use a variety of tools to accomplish their goals. This is especially true in teams that are geographically separated and brought together for a particular project. Development teams can bring their own toolsets as long as they eventually integrate. This situation means that developers must integrate goals, progress, and project dependencies in order to eliminate bottlenecks and see the whole picture.

High-level portfolio management tools such as OpenText™ Project and Portfolio Management can integrate out-of-the-box with Agile tools, including OpenText™ Software Delivery Management, CA Rally, Jira, VersionOne, and other popular tools. These features integrate without disrupting what the development teams are already doing. The integration automatically provides visibility into all business goals, resource allocations, value drivers, and project portfolios across the entire organization, without the overhead of manually re-entering information to keep multiple systems in sync. The key is integrating the information flow between development and business without slowing down either one.

**Providing flexibility is more than just allowing work from home. It's about allowing your team to use the tools they want without sacrificing visibility or having disconnected and outdated information.**

In the wake of the COVID-19 pandemic, more teams are discovering how to work together without being in the same physical space. Messaging and teleconferences have replaced walking across the office to chat. Teams each find their own ways to use technology to work together, determining their own processes and tools for collaboration. Automating the integration of those tools is key to flexible intra-team collaboration. Providing this flexibility is more than just allowing work from home. It's about allowing your team to use the tools they want without sacrificing visibility or having disconnected and outdated information.

## Incorporating testing and quality throughout the application development lifecycle

Testing and quality are important throughout the application development lifecycle. Ensuring that the product meets its requirements is not just testing whether the software conforms to technical specifications. Quality brings in the functional aspect of the software. Does it conform to expected measures of business functionality?

Measuring quality starts with developing quality metrics and testing whether these measures are met during the development lifecycle. Quality metrics are a necessary part of quality management. They take the customers' needs and translate them into measures to be applied to the software function and performance. Applying these measures as checkpoints during development, not merely at the end, can improve software quality and avoid costly rework.

To help understand how to emphasize quality throughout the development process, consider:

- Shift-left quality testing: building quality into the software from the initial design stage.
- Shift-right quality testing: monitoring software in production to catch quality issues impacting customers.

Shifting left refers to bringing quality as far back as possible in the initial design phase. Software development shouldn't begin with the statement, "You start coding. I'll go find out what the business wants." Quality begins with collaboration of all key teams and stakeholders in the development effort. The shift-left approach should apply to the development process as a whole, and not occur as a signal event. It shouldn't be the sole responsibility of quality assurance (QA) testers.

Shifting right refers to improving the software after release to the customer. While shift-left testing fixes problems detected by quality reviews in the development process, shift-right testing looks to remediate unknown or unknowable issues once the software is placed in the users' hands. These are issues like:

- Will the user be locked out of their home if their Wi-Fi or power goes out?
- Does a device's software have security vulnerabilities when connected to other devices on the network?

Simply put, shift-right testing determines how well software performs when exposed to realworld combinations of environmental factors.

Collaboration and communication between all key participants in the software lifecycle is essential. Different perspectives reduce the number of unknowns and make the software more robust, effectively catering to a variety of situations.

## Learn more at

opentext.com/products/devops-cloud ›

opentext.com ›

Standardizing tools and processes is a necessary part of the development process. This includes being able to capture, analyze, and preserve information during the product lifecycle. Developers, especially those working remotely, must be able to learn about issues impacting development. Tools need to channel information to the appropriate people to avoid losing actionable information in the communications stream.

Automating development and quality processes prevents mistakes that would cause rework and remediation expenses. Tools that automate information capture and use keep everyone focused on the right goals and help eliminate the “I didn’t see that email” problem.

## Conclusion

Dev teams are increasingly disconnected physically. The COVID-19 pandemic didn’t create the trend toward work from home and distributed teams but it certainly accelerated this trend. Good tooling and processes have always been necessary to keep teams aligned with the business and its goals, but they have become more critical with development and business teams working remotely.

To make sure products not only meet their technical specifications but also their real-world functional goals, teams must have strong communication processes and robust quality testing. Teams and their tools should never be isolated or disconnected. Everyone working on a product must be kept up to date on its functionalities and the value it’s meant to provide—not to mention inevitable changes along the way. Then, the right product can be delivered quickly and successfully.

Learn more about PPM and Software Delivery Management and and how they work together to enable businesses to automate and manage their development process from top to bottom and end to end.



opentext™