opentext™

eBOOK

ADM Market Insight: The Future of Dev in a Disconnected World

Content

The Future of Dev in a Disconnected World

The nature of software development has changed over recent years. It has achieved higher delivery speed with DevOps, Agile development, multithreading application development, and remote teams.

Useful applications require high-quality software that solves business problems. Modules that are delivered quickly but require rework defeat the purpose of DevOps. Not only do organizations need new tools and techniques to maximize value but also to ensure everyone accesses the same information.

How to shift from "job done" to "value realized"

- Not everyone will work on the same thing at the same time—or even with the same tools. Align all team members with a clear vision of what needs to be accomplished.
- Keep cross-functional teams on the same page with a way to collaborate and communicate effectively. This collaboration must be visible, accessible, and consistently available to all team members.
- Emphasize quality from end-to-end with clear quality metrics and determine how you'll test to fulfill these measures during the development lifecycle. Not only do organizations need new tools and techniques to maximize value but also to ensure everyone accesses the same information.

The Shift from "Job Done" to "Value Realized"

Strategic alignment is more important than ever. Because teams of developers work together, often in separate locations, keeping everyone on the same page is essential.

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 other team members, their activities may not align with business objectives.

They may develop software that, while functionally correct, only solves a subset of the business needs it was supposed to satisfy.

Consider a payment application. The intent is to obtain payment for the product and have customers receive what they ordered. Suppose the developer chooses to create a simple payment screen without allowing customers to review their order. In this case, the code may satisfy the functional objective of allowing payment while not meeting the customers' needs 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 development teams up to date on the whys of each project requires facilitating visibility into 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 delivering 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.

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.

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. Often, important information and updates that everyone must see are buried in a blizzard of emails, text messages, or meeting notes.

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

However, development teams use a variety of tools to accomplish their goals. This is especially true for geographically separated teams that collaborate on a project. As a result, developers must integrate tools, goals, progress, and project dependencies to eliminate bottlenecks and see the whole picture.

Value stream delivery solutions, such as OpenText™ ALM Octane, coupled with high-level portfolio management tools, such as OpenText™ Project and Portfolio Management, facilitate a centralized hub to align work. The capabilities integrate together and across the DevOps and Agile tooling ecosystem 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 reentering information to keep multiple systems in sync.

The key is integrating the information flow between development and business without slowing down either one.

opentext™

Incorporate 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 includes the functional aspects of the software. Does it meet the 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 customers' needs and translate them into measures teams apply 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 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 to 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 the 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 users receive the software. These issues include:

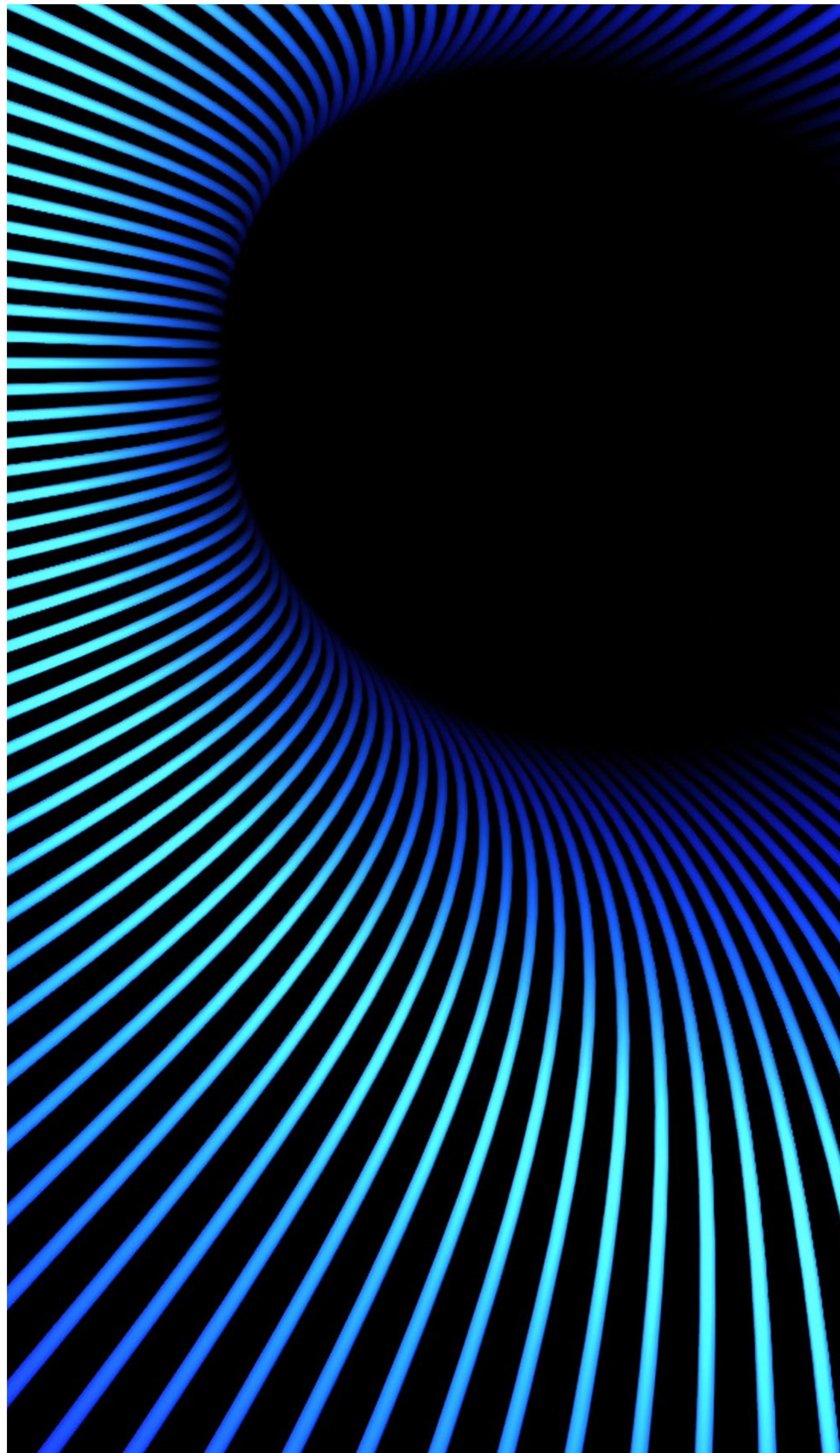
- Will the user be locked out of their home if their Wi-Fi or power goes out?
- Does a device's software have security vulnerabilities when connected to other devices on the network?

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

Collaboration and communication among key participants in the software lifecycle are essential. Different perspectives reduce the number of unknowns and make the software more robust, effectively catering to various situations.

Standardizing tools and processes is a necessary part of the development process. Standardizing 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.



opentext™

Conclusion

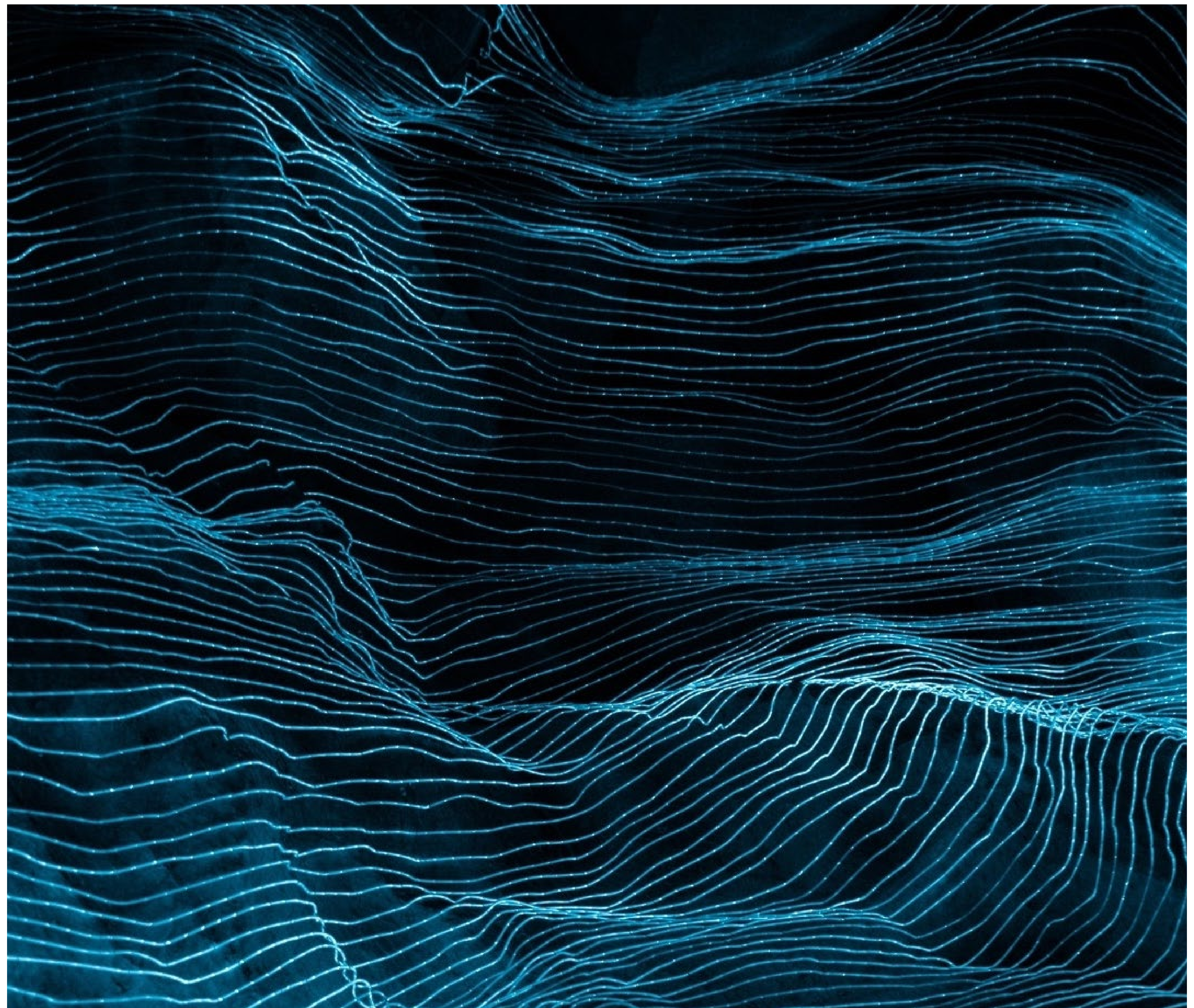
Good tooling and processes have always been necessary to keep teams aligned with the business and its goals. They are now more critical than ever, with development and business teams working remotely.

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

To ensure products meet their technical specifications and 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 up to date on its functionalities and the value it's meant to provide—not to mention inevitable changes along the way. Then, they can quickly and successfully deliver the right product.

Learn more about OpenText™ ALM Octane and OpenText™ Project and Portfolio Management solutions and how they work together to enable businesses to automate and manage their development process.

Learn More



About OpenText

OpenText, The Information Company, enables organizations to gain insight through market leading information management solutions, on premises or in the cloud. For more information about OpenText (NASDAQ: OTEX, TSX: OTEX) visit opentext.com.

opentext.com

X (formerly Twitter) | LinkedIn | CEO Blog