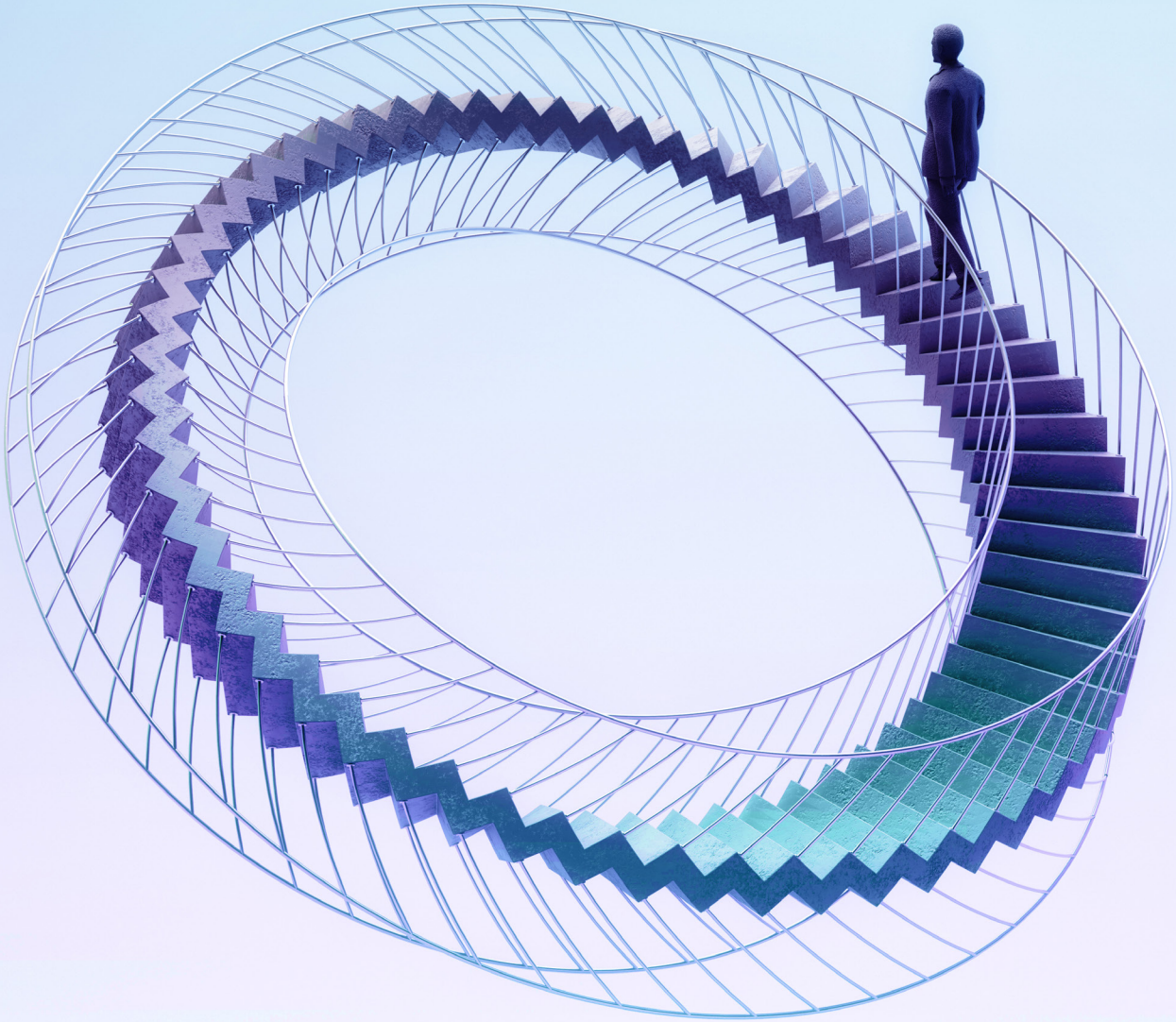
opentext™



ADM Market Insight: Achieving Agile software quality

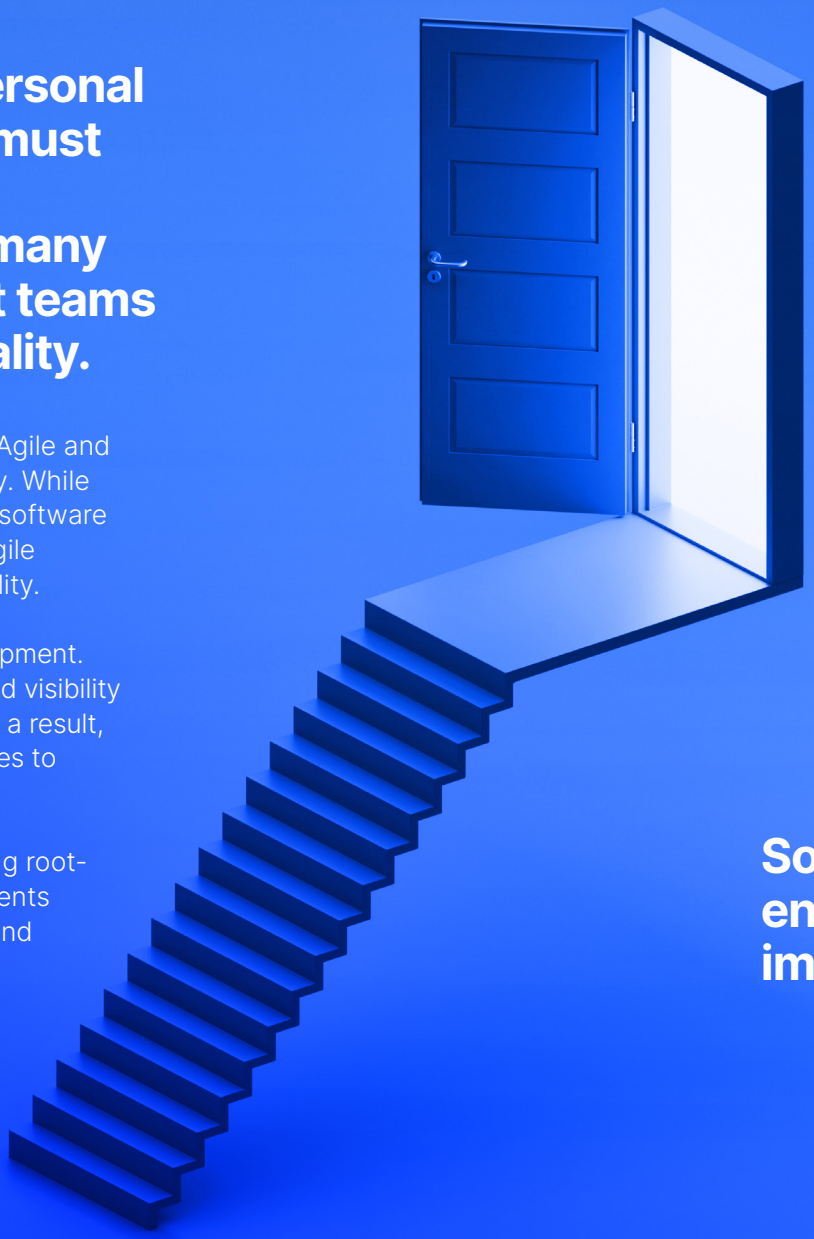
As software use increases in our personal and professional lives, enterprises must deliver quality applications to meet consumer expectations. However, many enterprises with large development teams struggle to ensure high product quality.

Part of this issue is due to a focus on agility. Companies adopt Agile and DevOps practices to speed up software production and delivery. While these processes improve both, they don't necessarily focus on software quality. But with the right tools and techniques, every part of Agile lifecycles and DevOps pipelines can contribute to software quality.

In a DevOps environment, IT operations support software development. It's challenging to plan and track work across many teams. Limited visibility into the DevOps toolchain is often the source of these issues. As a result, you might run into insufficient test planning and limited capabilities to execute tests.

DevOps pipeline management becomes complex and performing root-cause analysis becomes time-consuming. Measuring achievements then becomes challenging, and organizations cannot quantify and track quality metrics.

When an organization adopts Agile and DevOps practices, failing to take quality into account actually slows velocity, creates technical debt, and results in lost productivity.



So, what can enterprises do to improve quality?

Achieve software quality at the pace

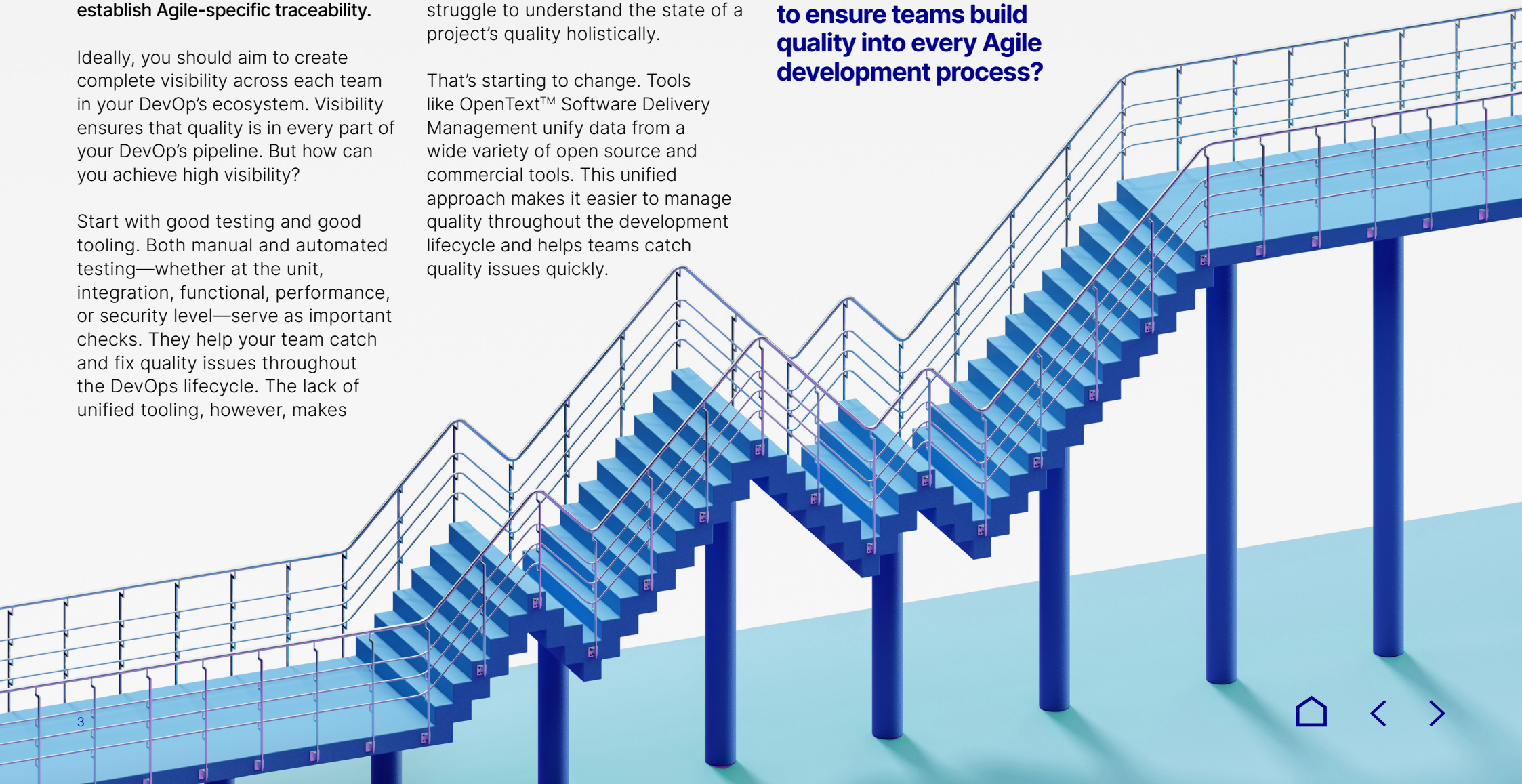
DevOps plays a vital role in the software development lifecycle, both for timely releases and for quality. It makes it easier to place quality at the core of application delivery and establish Agile-specific traceability.

Ideally, you should aim to create complete visibility across each team in your DevOp's ecosystem. Visibility ensures that quality is in every part of your DevOp's pipeline. But how can you achieve high visibility?

Start with good testing and good tooling. Both manual and automated testing—whether at the unit, integration, functional, performance, or security level—serve as important checks. They help your team catch and fix quality issues throughout the DevOps lifecycle. The lack of unified tooling, however, makes maximizing ROI on testing difficult. Different testing types done at different points in DevOps pipelines often use different reporting tools. With all these differences, managers struggle to understand the state of a project's quality holistically.

That's starting to change. Tools like OpenText™ Software Delivery Management unify data from a wide variety of open source and commercial tools. This unified approach makes it easier to manage quality throughout the development lifecycle and helps teams catch quality issues quickly.

DevOps is a good start. But what else can we do to ensure teams build quality into every Agile development process?

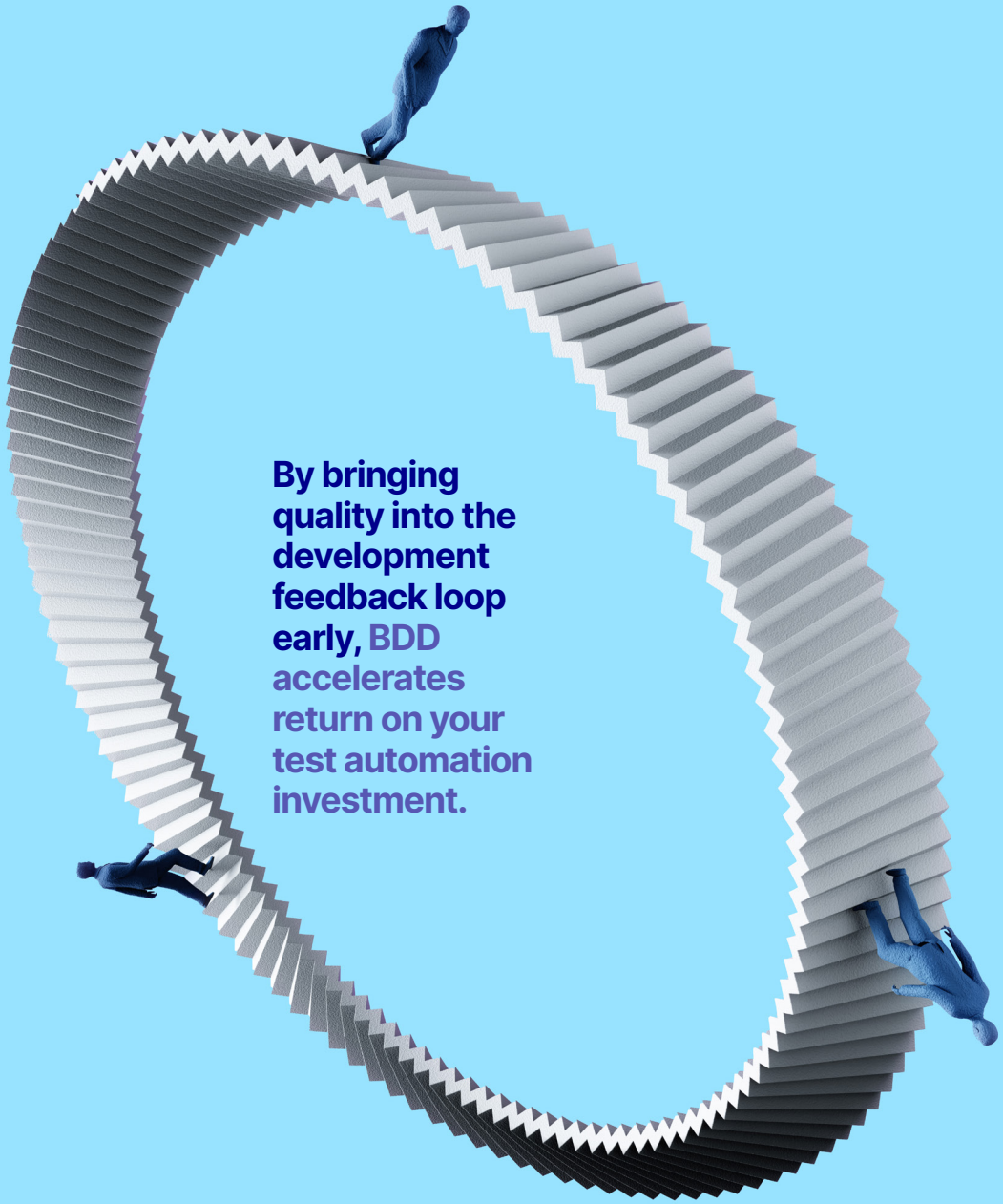


Create cross-functional alignment

Cross-functional teams must plan and track related work to produce high-quality releases. Behavior-driven development (BDD) can be a key driver of this process.

BDD uses real business-driven scenarios and use cases to drive the development and testing of new software features. It's not just a new facade over old tools and techniques. BDD uses new tools and processes to improve collaboration. Development, quality assurance (QA), and management pass requirements to one another, reducing translation costs.

By bringing quality into the development feedback loop early, BDD accelerates return on your test automation investment. It shifts testing left in the delivery lifecycle and provides a flexible framework for defining what to automate. BDD helps enterprises focus on quality throughout the DevOps cycle—from Agile planning and test execution to deployment and production.



Enable a high-performing team of teams

High-quality applications require complete visibility across teams and Agile release trains for release planning and progress tracking. "Team of teams" is an organization where the relationships among teams are similar to the relationships among members of a single team.

In enterprise software development, the team-of-teams approach can manifest in several ways. You might have discrete teams organized around functions, such as product management, software development, and QA. Or you might have cross-functional teams that each contain a product manager, software developers, and QA testers. Either approach to team building can work. And when you're building, testing, and deploying software at enterprise scale, a team-of-teams alignment is critical. Even the strongest development teams suffer quality setbacks as inter-team dependencies are lost, and timelines suffer.

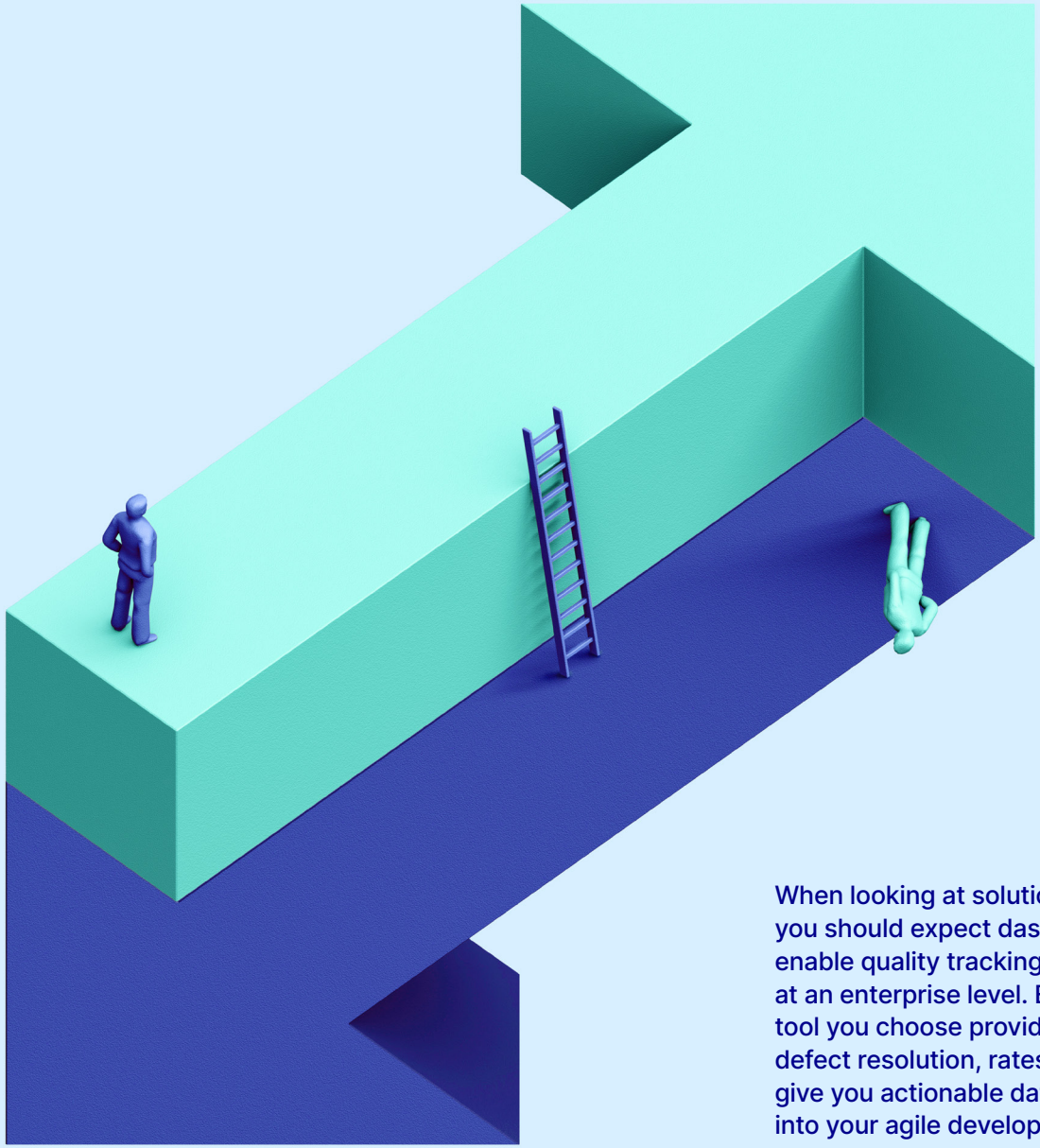
The team-of-teams approach comes with a cost, however: coordination overhead. Fortunately, modern DevOps tooling solves this issue by streamlining handoff between teams and providing management with a bird's eye view of new features and fixes.

Attain product-level visibility of quality and velocity

IT and business leaders must maintain a holistic view of velocity and quality progress. To maximize quality and velocity, they must understand quality across all teams, rather than at the project or team level. Product-level visibility allows you to measure progress and quality against critical key performance indicators (KPIs) by connecting the dots across parallel work streams.

The necessity for visibility ties back to our discussion of team of teams. It's great if multiple teams work on different parts of a project—perhaps even in parallel. But without coordination, a multi-team effort can quickly devolve into a free-for-all that makes software quality worse instead of better. It's difficult to focus on end-to-end, DevOps-driven quality without understanding what work you're doing across the entire development pipeline.

High-level tools like OpenText Software Delivery Management enable Agile release trains and solution trains to easily measure product quality by associating features, defects, and tests with application modules representing the application's functional areas.



When looking at solutions in this space, you should expect dashboards that enable quality tracking and governance at an enterprise level. Ensure that any tool you choose provides metrics like defect resolution, rates, and aging to give you actionable data to feed back into your agile development process.

Implement a comprehensive test strategy

Tests aren't all created equally. Different scenarios require different types of tests. Running manual tests early in the development cycle ensures functionality meets proposed acceptance criteria. Later, you can automate tests for future regression and end-to-end testing.

From manual testing to robust automated test suites and CI, you should measure software quality against acceptance criteria at every step. Aim to efficiently quantify, track, and manage Agile delivery quality throughout the entire application development lifecycle.

Increase test automation ROI

Test automation should be a key part of your testing strategy. While it can't catch everything, automate as much testing as you can. Think of it as raising the quality floor of your application. Every automated test you add now represents a regression that won't happen in the future.

Test automation is at the foundation of any Agile or DevOps transformation. But, like other development activities, it's not free. Automated tests take time and money to develop. It's important to get a return on any time your team invests in automated testing. Aim to answer questions like:

+ **As automation increases, is cycle time decreasing? Are we able to ship more quickly?**
+ **If automation is slowing us down, does it make up that time with increased quality?**
+ **As automation increases, am I finding defects earlier?**
+ **As automation increases, are fewer defects making it to production?**

Yet again, holistic agile tooling helps find answers to these critical questions. Embedded analytics and context simplify critical decisions, such as what to automate and what not to. After all, it's impossible to determine whether you're benefiting from test automation without a way to get a detailed look at how the quality and velocity of your software projects have changed over time.

Enabling Agile quality with OpenText Software Delivery Management

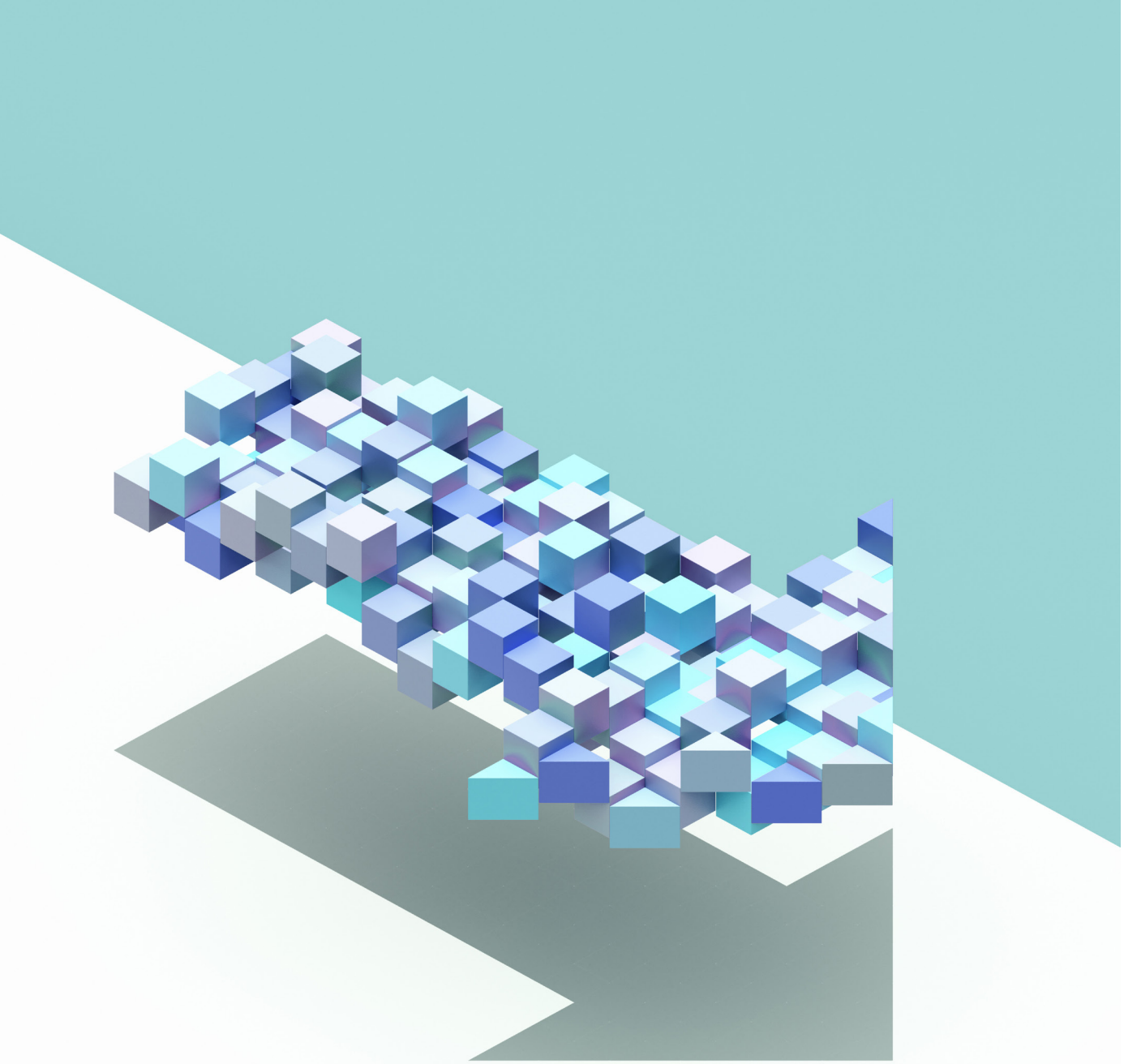
We've touched on the need for end-to-end tooling that drives agile software quality. But this kind of holistic agile tooling isn't easy to find. Ideally, you need a solution that talks to all your existing tools. This approach allows product managers, software developers, and testers to keep using the tools they're already productive with.

Tooling fragmentation is one obstacle to embracing a true end-to-end approach to enterprise software quality management. Fortunately, OpenText Software Delivery Management solves this by integrating existing Agile and DevOps tooling and providing a sound foundation for a team-of-teams alignment. OpenText Software Delivery Management facilitates:

+ **A common release cadence.**
+ **Advanced dependency mapping.**
+ **Complete visibility into every team's progress.**

OpenText Software Delivery Management's comprehensive feature set enables an end-to-end focus on enterprise software quality and easily scales to thousands of users. You also benefit from real-time continuous integration and continuous delivery status, and you can easily view changes and identify the root cause of failures.

Better yet, OpenText Software Delivery Management helps accelerate delivery while maintaining governance by offering support for enterprise agile frameworks like SAFe 4 .0 and DaD. OpenText built OpenText Software Delivery Management with the understanding that quality should be everywhere in the software delivery. Its capabilities highlight that quality is prioritized beyond just testing .



Who benefits from OpenText Software Delivery Management?

A wide variety of enterprise roles benefit:

+ The research and development (R&D) manager can prioritize, manage, govern, track, and report on the lifecycle and quality of business-critical applications.
+ The QA manager can ensure quality across software releases and act as a bridge between development and QA teams—and OpenText Software Delivery Management helps them handle the tradeoff between a quick release and comprehensive testing .
+ Project managers ensure business requirements and goals are met throughout the development process with OpenText Software Delivery Management.
+ The QA engineer uses OpenText Software Delivery Management while creating detailed, comprehensive, and well-structured test plans and test cases .
+ The DevOps team responsible for guiding code releases from development to release can use OpenText Software Delivery Management to streamline and automate workflows.

How to get started

Agile and DevOps can help improve quality, as long as you use the right tools and make a conscious effort to focus on quality at each stage of your development lifecycle. OpenText Software Delivery Management helps you implement these best practices so that you can move fast while maintaining quality. Contact us today to learn about the benefits of using OpenText Software Delivery Management.

Contact us today

opentext™