OverOps

EXECUTIVE OVERVIEW

Continuous Reliability

THE PROBLEM

Innovating at speed and scale during uncertain times

In today's climate, applications are becoming the lifeblood of your organization. The world's new reality dictates doing more with less, while possibly facing a hiring freeze or slowdown. Companies have prepared for this by adopting CI/CD and increasing development velocity, but as uncertainty grows, stressed software quality is becoming the bottleneck. The faster companies innovate and release new features, the harder it is to ensure quality and reliability of software. Releasing a GA version has effectively become the new beta and customers are now the new QA. To make things worse, there's never enough time for testing and even when it seems sufficient, there is no guarantee that the code is absolutely safe to promote.

Stressed code quality is exposing the limitations of existing tools

This shift has exposed the limitations of existing DevOps processes and tools. Today, organizations typically use 3 types of solutions to address code quality across the software delivery lifecycle: testing & static analysis, log management, and performance management tools. However, these tools have inherent limitations:

1. **Testing & Static Analysis:** *requires foresight into what may break.*
   Inability to cover all errors conditions — not even with 100% code coverage.

2. **Log Management:** *requires foresight, noisy, manual and reactive.*
   No way to prioritize errors and separate the signal from the noise.

3. **Application Performance Management (APM):** *focuses on performance, not errors.*
   No relevant content & context beyond what's already in the logs.

The bottom line is that even though existing DevOps solutions find *many* errors, the errors they miss are having a major impact on customer experience. And the errors that are

identified - are hard to prioritize and fix with the limited context that's provided by existing solutions.

THE SOLUTION

Introducing Continuous Reliability (CR) with OverOps

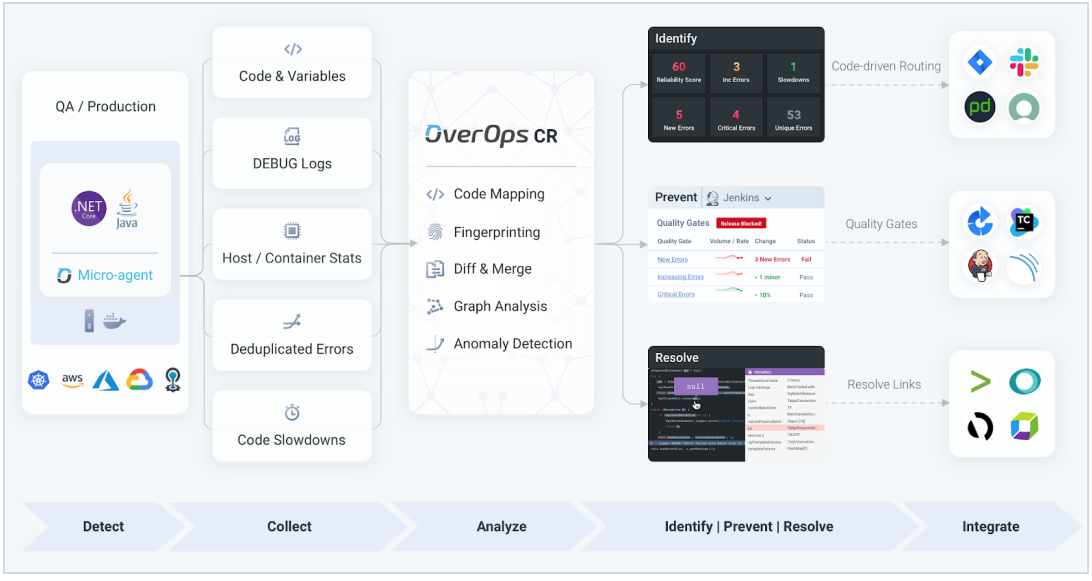
To overcome the limitations of existing tools, a new automated approach was developed to help organizations overcome the agility/stability paradox. Continuous Reliability helps prevent code changes from impacting the customer, and when an issue does reach production - it helps identify and resolve it before customers are severely impacted. Here's how OverOps implements this approach with the following building blocks:

1. **Runtime Code Analysis:** *specifically built for mission-critical Java & .NET applications.* Proactively identifying signals, analyzing code as it executes — not the logs.

2. **Code Quality Gates:** *integrated into existing testing, CI/CD and pipeline solutions.* Report on unknown errors, informing go/no-go decisions for new releases.

3. **Automated Feedback Loops:** *integrated with logs, APM, ticketing, and alerting.* Source code, variable state, and operational context for every critical error.

Furthermore, the specialized data that is collected and analyzed by OverOps is enabling a "shift left" use case to prevent issues that were missed by testing in pre-production, and "shift right" to identify new errors in production. And In both cases, provides a feedback loop back to development with the complete code-level context that's required to resolve.



Using OverOps, companies like Comcast, BT, Visa, TripAdvisor, Intuit, Cox Automotive, Aflac and many more have introduced Continuous Reliability into their workflows, saving millions of dollars in churned customers and lost revenue without impacting the customer experience.