

## FIND WHAT YOUR TESTS MISS

### ISSUES ARE NOT CAUGHT IN PRE-PROD



Many issues are missed in testing and only get detected by APMs in production, leading to business interruptions and revenue loss.

### INVESTIGATING PROD ISSUES IS MANUAL



In complex environments, performance and scaling issues could take days and weeks until engineering teams can identify their root cause.

### ENGINEERING TEAMS WASTE DEV CYCLES



Engineering teams spend an average of 20% of their time on troubleshooting production issues rather than developing new ones.

## PREEMPTIVE OBSERVABILITY ANALYSIS (POA)

Digma's POA engine is designed to identify issues by their PATTERNS (rather than by thresholds), alerting engineering teams to issues in code level as it executes in pre-prod environments.

### AVOID BREAKING CHANGES

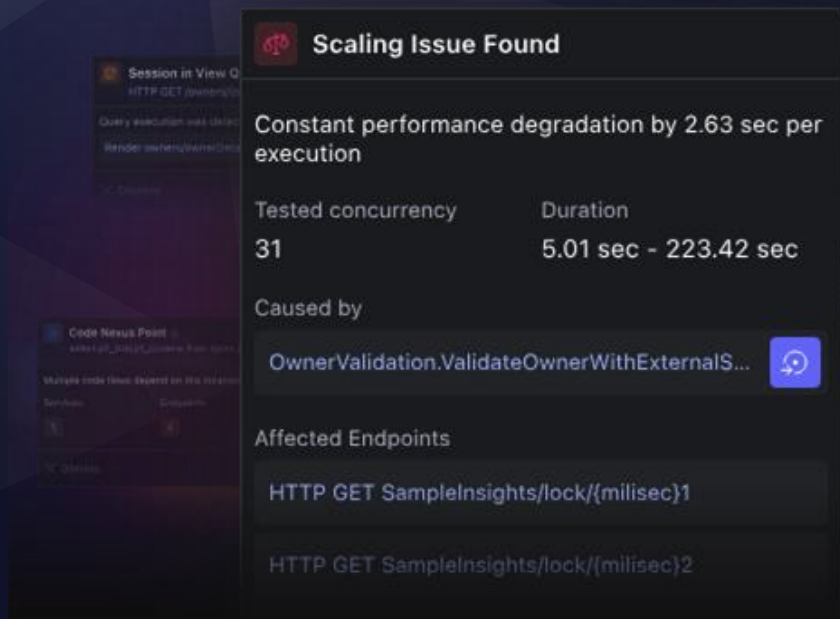
Digma POA engine highlights the affected areas for every code change and Pull Request, eliminating the risk of breaking changes and consequent loss of engineering time.

### AUTOMATICALLY IDENTIFY PERFORMANCE ISSUES

APMs alert about performance issues when they have impact on the organization's SLOs. Digma's, identifies scalability and performance issues in pre-prod, down to a line of code. Allowing faster resolution time with no customer impact.

### DRIVE DOWN INFRA COSTS

Most of the cost saving tools focus on cloud infrastructure inefficiencies. Digma's POA engine identifies inefficient code areas that impact your application cost and areas for optimization.



**Scaling Issue Found**

Constant performance degradation by 2.63 sec per execution

Tested concurrency	Duration
31	5.01 sec - 223.42 sec

Caused by

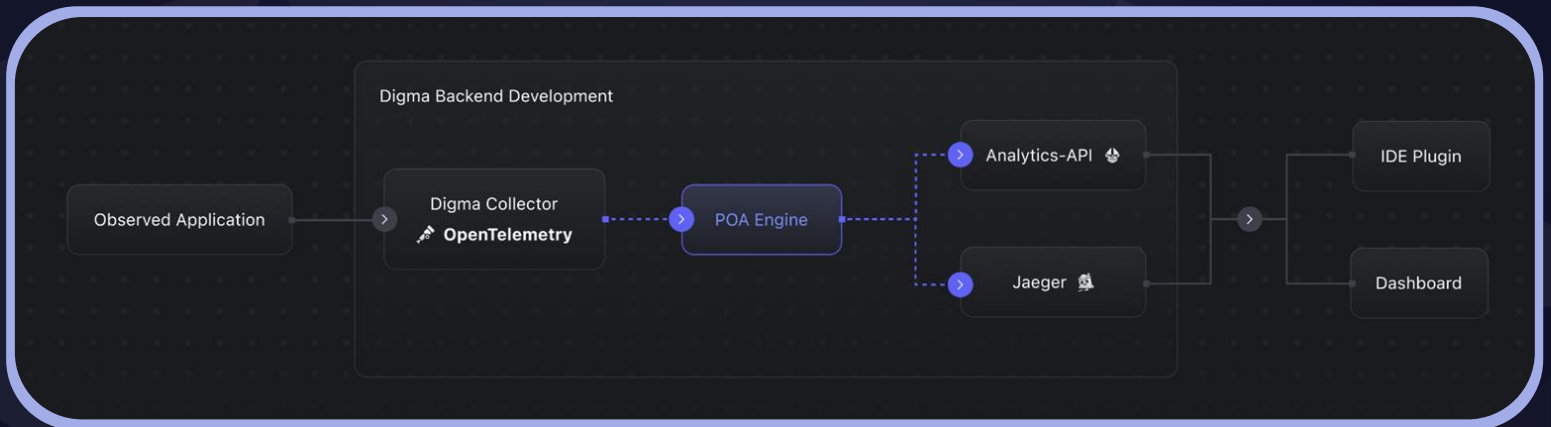
`OwnerValidation.ValidateOwnerWithExternalS...`

Affected Endpoints

- HTTP GET SampleInsights/lock/{milisec}1
- HTTP GET SampleInsights/lock/{milisec}2

# TRANSFORM OBSERVABILITY DATA INTO INSIGHTS

Digma uses OpenTelemetry (Otel) behind the scenes to collect data (traces, logs, and metrics) about the code when you execute it in pre-production or production environments. After collecting the data, Digma POA engine analyzes it to detect meaningful insights about the code. It looks for regressions, anomalies, bottlenecks and other code smells, as well as useful patterns to benefit from during development.



ALL DATA  
STAYS LOCAL



CODE CHANGES  
NOT REQUIRED



NO PUBLIC AI  
IS USED

## HOW IS DIGMA DIFFERENT

### APMS

Digma is complementary to APMS. It leverages the same observability data but goes beyond to identify code issues during development. Unlike APMS that focus on post-issue application monitoring, Digma targets the code level, from pre-prod to production.

### PROFILERS

Profilers are helpful once an issue has already been detected in production, and the engineering team identified the scenario in which it reproduces. Digma takes a proactive approach in continuously and automatically detecting scanning observability data for issues, and finding the root cause in the code. Usually, Digma finds these issues before they even manifest in production.

### STATIC CODE ANALYSIS TOOLS

Static code analysis tools can only relate to the code structure and have no access to the runtime data. As such, they are not able to identify real-world issues, examine the code performance and scaling, or catch errors and exceptions.