



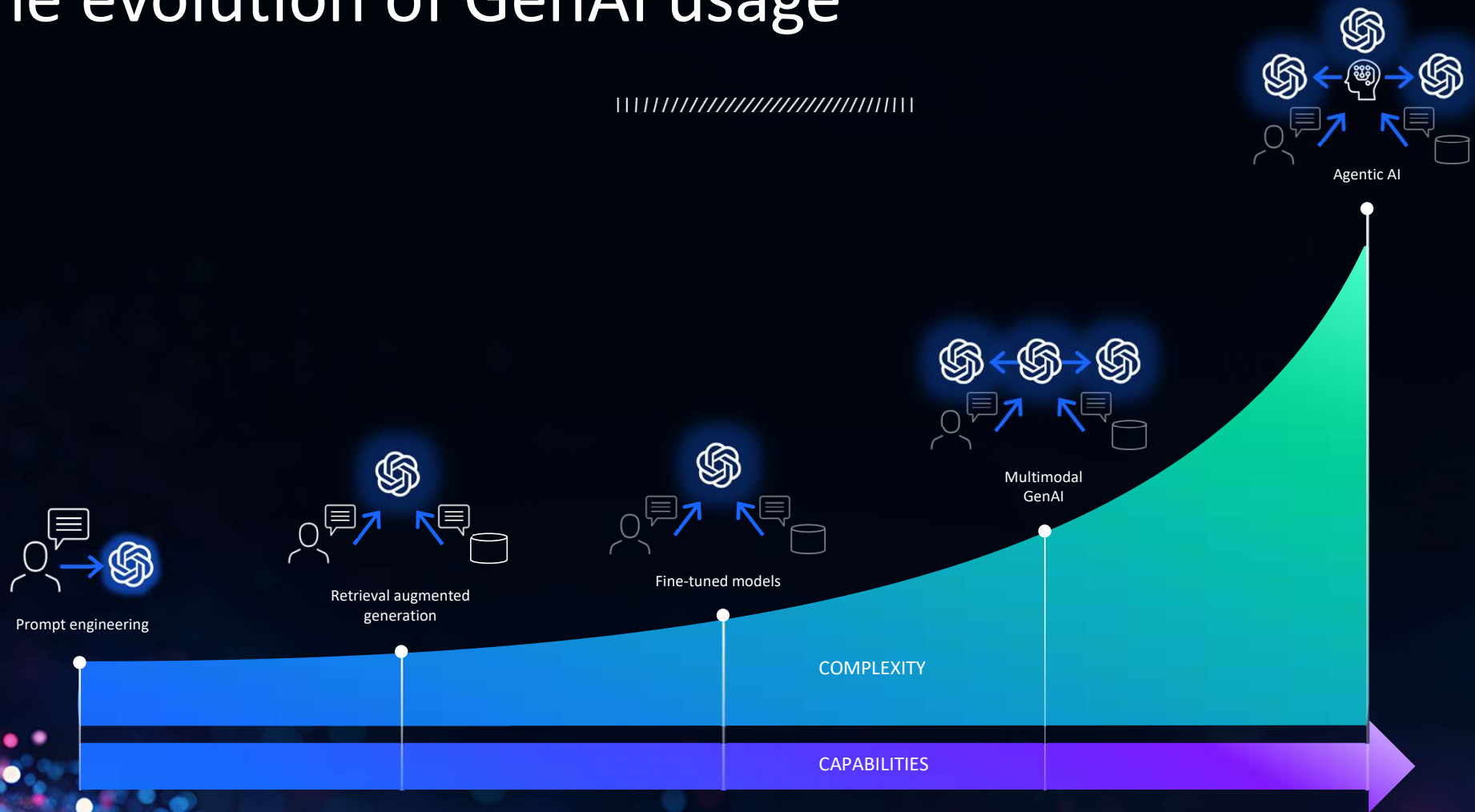
AI and Observability



Roman Spitzbart

VP, Solutions Engineering EMEA

The evolution of GenAI usage



40%

Of enterprise applications will be integrated with task-specific AI agents by end of 2026, up from less than 5% in 2025.

[Gartner](#)

57%

Of organization now deploy agents for multi-stage workflows

[Anthropic](#)

80%

Of organization believe AI **agents are the new enterprise apps**, trigger reconsiderations of investments in packaged apps

[IDC](#)

-15%

Expected productivity loss for companies failing to establish high-quality, AI-ready data foundations.

[IDC](#)

Executives expect productivity

Learn from past incidents to prevent future problems

AI automations will free up 30% of engineering time

Software will fix itself

Shift focus away from maintenance is 70% of team's effort

Operations expect automation

Incidents handled autonomously or prevented entirely

Offload fixes, define quality goals, and guardrails

Architect the use of AI

Developers expect convenience

Get full context and suggest error repair

Suggest tunings based on production insights

Create tests automatically

Auto-collect info to fix vulnerabilities

Agent adoption is no longer limited by model capability — the limiting factors have shifted to integration with existing systems, security, and operational scalability.

AI-Driven Development & Autonomous Operations require reliable agents, but how?

Hallucination of LLMs cannot be eliminated in the real world

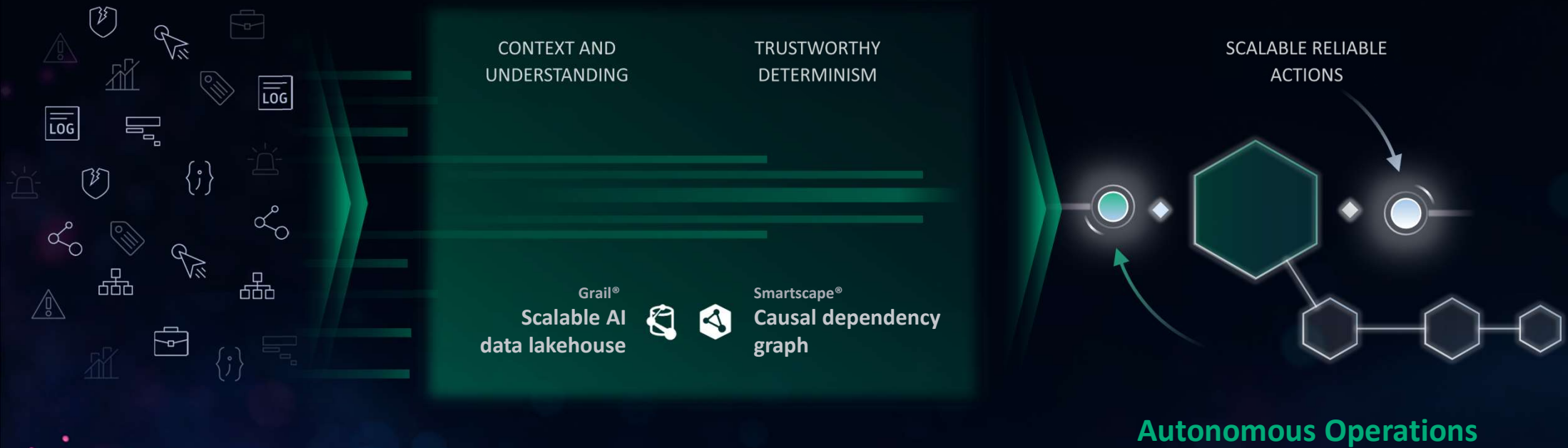
arXiv:2401.11817

“You can embrace creativity and uncertainty at the AI layer only if your data layer is absolutely deterministic.”

Kiki Carter, Cockroach Labs

APPLYING LEARNINGS TO DELIVER SUCCESS

Operate on determinism and facts, not just probabilistic guesses



THE LEADER

Reactive operations

Issues are deeply understood after they occur

Dynatrace
2006+

THE INNOVATOR

Automated root-cause

Issues are auto-diagnosed when they occur

Dynatrace
2014+

THE PIONEER

Preventive operations

Issues are anticipated before they occur

Dynatrace
2023

THE FUTURE

Autonomous operations

Issues are resolved before they make impact

Dynatrace
2026+



The AI-powered observability platform for autonomous operations



Integrated platform



Unified data



Real-time context

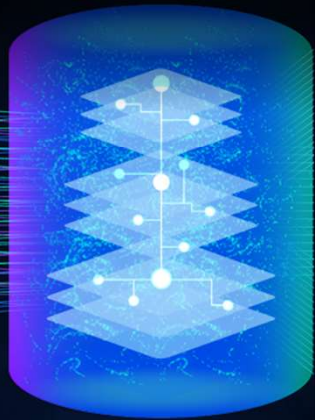


Answers, not guesses



THE DYNATRACE PLATFORM

Action based on answers, not guesses



Unified AI data lakehouse
Grail



Real-time dependency graph
Smartscape



Agentic operations system
Dynatrace Intelligence



More than ever

Agentic AI relies on accurate and deep context to operate **autonomously**

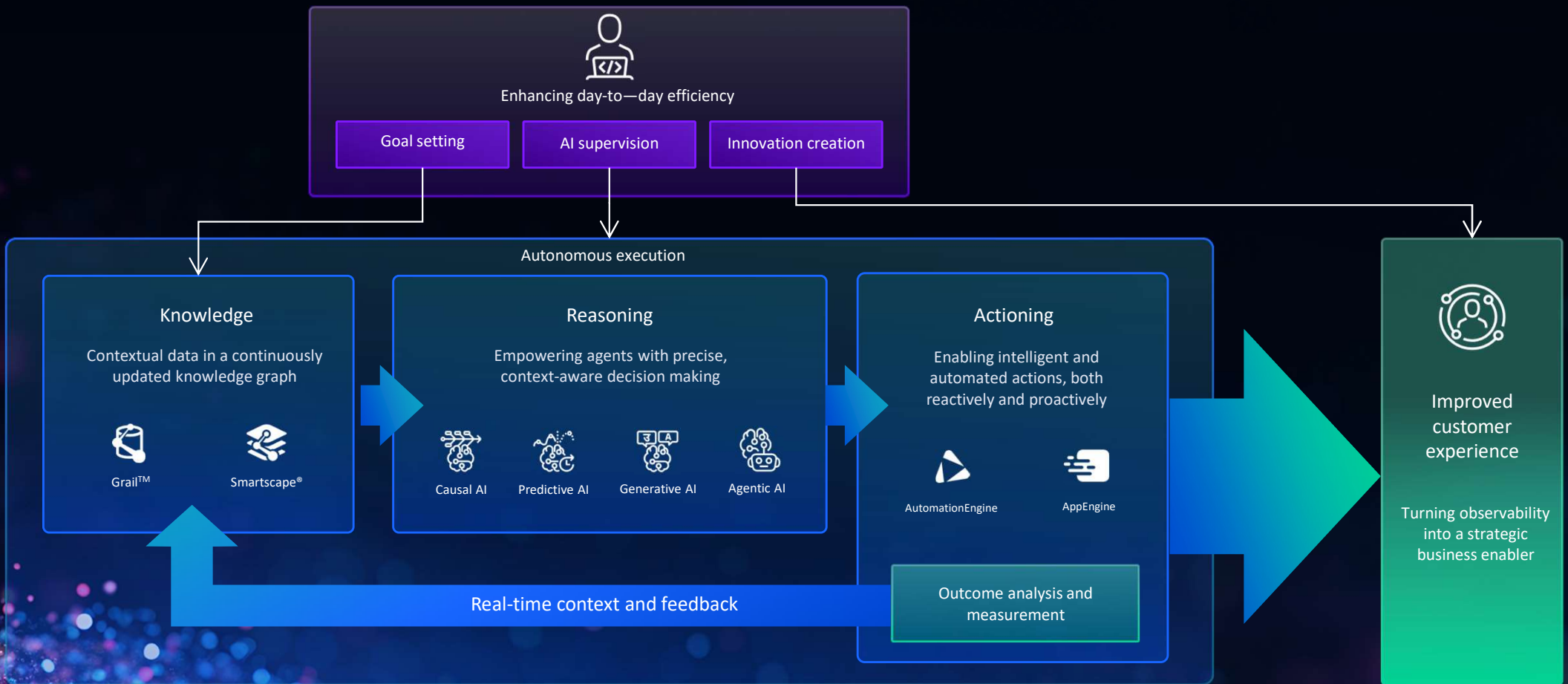


More than ever

Agentic AI relies on accurate and deep context to operate **autonomously**



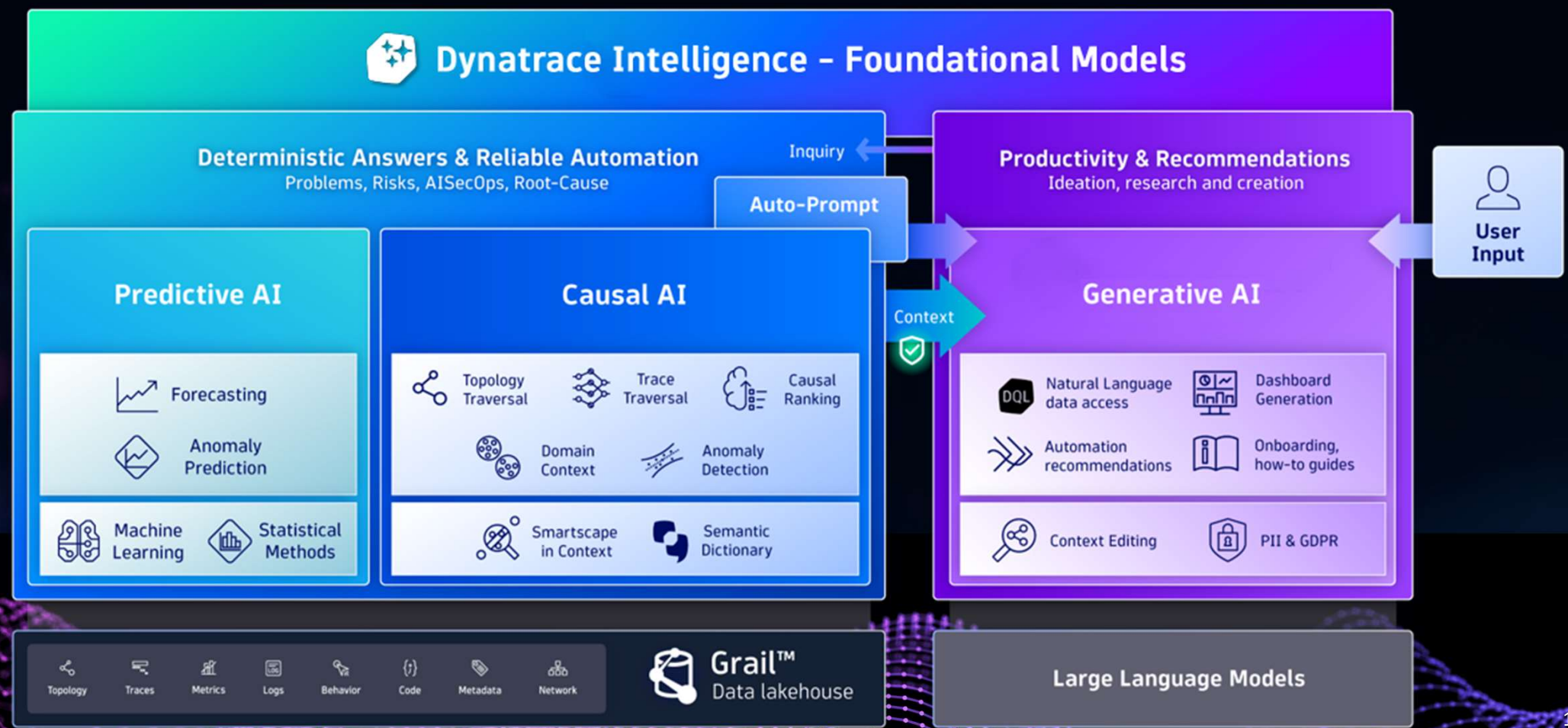
Leveraging Dynatrace architecture





Determinism as the Foundation of AI

Agentic AI is powerful, but only as good as its foundation



Snímka 15

- DP1** Need a new slide for this
Deblois, Philippe; 2026-02-17T18:23:25.386
- CK1 0** Replaced Davis with Dynatrace Intelligence
Kiesewetter, Christian; 2026-02-22T22:33:18.339

AI, LLM and Agentic Observability

Agentic Operations

Dynatrace: the AI that observes AI

End-to-end full-stack visibility across cloud-, AI-workloads and customer experiences chat interface, prompts, models, tools, and GPUs on K8s, all in one view.

Contextual analytics and runtime-security

Trace, log, metrics, user-behavior, topology across agent execution, tool usage, cost, and performance Alerts and root-cause across app and AI layers. Runtime security, compliance and guardrails provide trust.

AI-powered automated observability to remediate, protect and optimize

The full power of Dynatrace causal, predictive and generative AI is true also for AI workloads

SRE/Ops

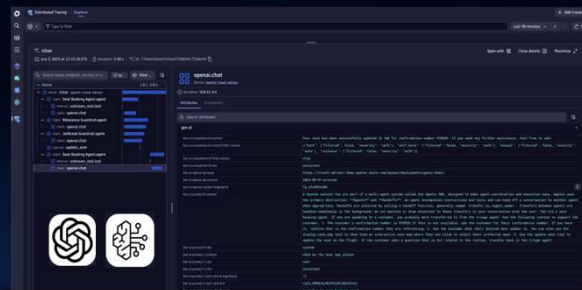
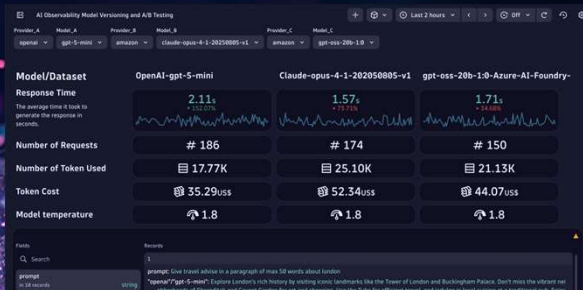
Predictive operations for forecasting performance and cost bottlenecks across cloud

Development teams

End-to-end tracing of agent calls/tool usage
5+ agent technology supported

AI Teams

Guardrails for data safety, quality, and trust: detect risky outputs, abuse; keep auditable prompt history





Dynatrace Intelligence

The agentic operations system that fuses the strengths of deterministic and agentic AI for reliable, autonomous action at scale.





Dynatrace Intelligence

Dynatrace has evolved
into an **agentic AI**
platform

Built for observability, security and business

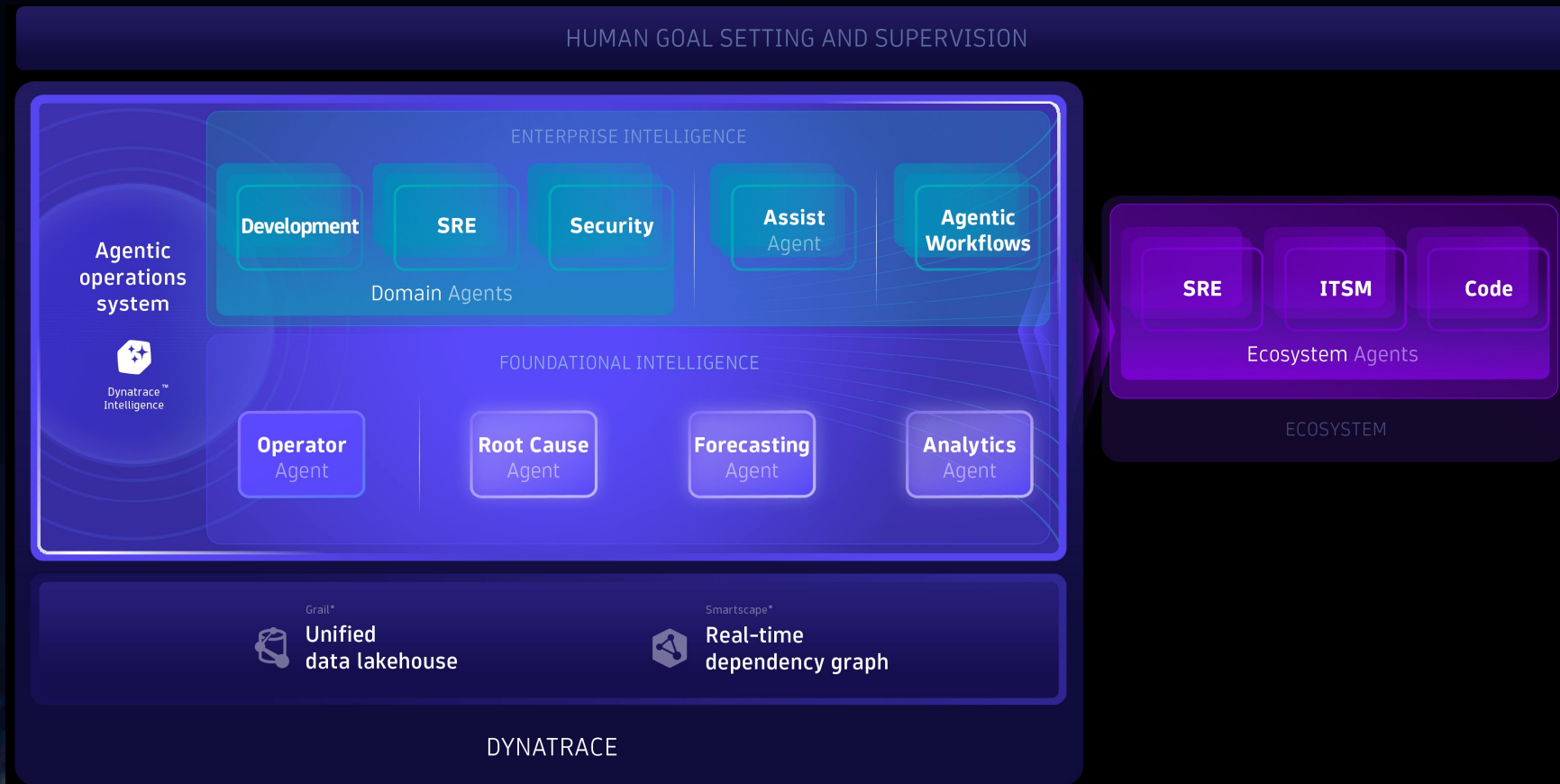
Orchestrating internal and
external AI agents

Creating AI agents from
app functionality

Collaborating with other
external AI agents

DYNATRACE INTELLIGENCE

Fueling the AI-powered observability platform for autonomous operations





Dynatrace Intelligence

Extend intelligence to every interaction and any ecosystem

ACTION

INTEGRATION

Team Augmentation

Plan, Research,
Orchestrate & Supervise

Real-time Facts &
Knowledge Graph



**Agentic
Workflows**



**Assist
Agents**



**Ecosystem Agent
Interactions**

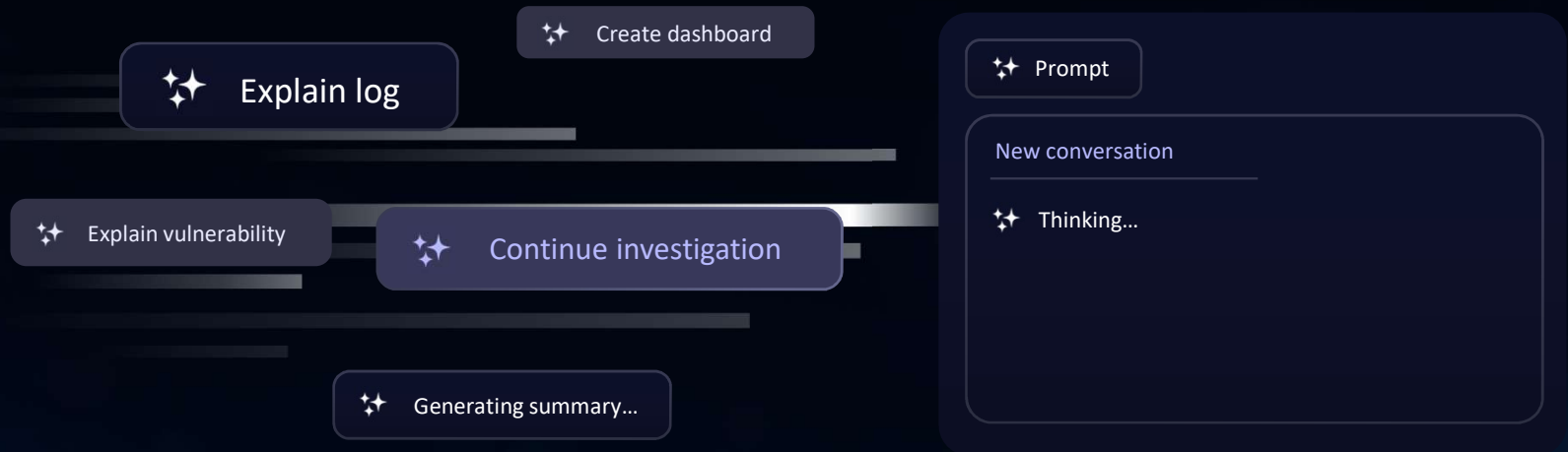
GENERALLY AVAILABLE TODAY



DYNATRACE INTELLIGENCE

Ubiquitous assistance for easier use

ASSISTANCE



Interactive context-aware assistance

ASSISTANCE



Investigation for P-2507151
+ New

Let me continue the investigation for P-2507151

Thought for 4s

Full-stack investigation Agent

I've run a Full-Stack Investigation to analyze the root cause of the JavaScript error rate increase in the astroshop application.

Here's what I've found:

- Logs Analysis:**
 - Error: Connection timeout while accessing database.
 - Error: POST /checkout 500 Internal Server Error.
 - Warning: Memory usage exceeded 90% threshold.
- Trace Analysis:**
 - Failed transactions detected in POST /checkout API calls.
 - Average response time: 2.5 seconds, which is higher than expected.
- Infrastructure Analysis:**
 - aks-live Kubernetes cluster: No pods ready for the astroshop-payment workload.
 - ntp-dev5-plsrv kafka host: Memory usage at 95%, indicating resource exhaustion.
- Dependency Map:**
 - The astroshop-payment service depends on the database, which is experiencing connection issues.
 - Cascading failures are propagating to the frontend, causing JavaScript runtime errors.*

Here's my recommendation:

- View Logs:** Analyze logs from astroshop-payment to identify database timeouts and memory issues.
- View Failure Analysis:** Investigate pod readiness in the Kubernetes cluster and resource saturation on the Kafka host.
- Debug in Code:** Check for implementation issues in the astroshop-payment service (Recommended).
- View Traces:** Trace failed POST /checkout transactions to pinpoint failures.

Ask a question, analyze data, or take action.

Agent: [AI Icon]

Dynatrace Intelligence uses AI. Always verify important information and decisions.

JavaScript error rate increase

Active P-2507151 Error Started at Jan 22, 2026, 09:00 for 30 minutes

Overview Events Logs Troubleshooting

Summary and recommendations

Why it happened?

The problem likely occurred due to one or more of the following reasons:

- Service Failure in astroshop-payment**
The astroshop-payment service is experiencing a failure rate increase, which is causing API calls from the frontend (e.g., POST /checkout) to fail. These failed API responses are resulting in JavaScript runtime errors in the astroshop-frontend.
- Resource saturation**
Insufficient resources (e.g., CPU, memory, or database connections) in the astroshop-payment service or related infrastructure could be causing the service to fail. High memory usage has been observed on the ntp-dev5-plsrv kafka host, which may contribute to the issue.
- Increased load**
A sudden spike in traffic or requests to the astroshop-payment service may have overwhelmed the service, leading to cascading failures.
- Kubernetes Pod Issues**
The aks-live Kubernetes cluster reports that no pods are ready, which could indicate resource contention or misconfigurations affecting service availability.

What was affected? <

What was impacted? <

How to solve? <

Dynatrace Intelligence uses AI. Always verify important information and decisions.

Continue investigation

Root cause

- astroshop-payment** (Service) Root cause
 - Error: Service failure rate increase Analyze failures
 - Info: ArgoCD Sync: astroshop-payment (commit:abc) Analyze failures
- payment** (Kubernetes workload)
 - Error: Service failure rate increase Analyze failures
- astroshop** (Kubernetes namespace)
 - Slowdown: Unexpected low service load View Service
 - Error: Failure rate increase Analyze failures
- aks-live** (Kubernetes cluster)
 - Error: No pod ready View Kubernetes workload
- ntp-dev5-plsrv kafka 10.178.189.59** (Host)
 - Availability: Host or monitoring unavailable View Host

Aggregate

Manage data

- Affected frontend: 2
- Affected services: 5
- Affected infrastructure
- Affected synthetic monitor



Interactive context-aware assistance

ASSISTANCE



The screenshot displays the Dynatrace Intelligence interface. On the left, a chat window shows an AI assistant's interaction with a notebook. The assistant has analyzed the notebook and provided a summary of the top 3 insights. The user has responded with 'Yes' to a question about adding the summary to the notebook. The assistant has then updated the notebook and added the summary as a Markdown section. On the right, the notebook's content is displayed, including a title 'Notebooks - Q3 25 - Quarterly Adoption Report', a section for 'Top insights - last 12w vs prior 12w', and a chart for 'Adoption metric 1: Notebooks opened'.

Summary quarterly adoption report

Analyze this notebook and summarize the top 3 insights. Compare the last 12w with the 12w before that. Be ultra-concise.

Thought for 8s

I'll start with analyzing the data in the notebook and afterwards create the top three insights.

Analyzed notebook

Here is the ultra-concise summary of the top 3 insights from this notebook.

Top insights — last 12w vs prior 12w

- Creation intensity led
 - Avg created/user +27.71%; weekly created +34.08%; total created 124.4K; creators 19.1K; avg created/user 6.52.
- Engagement up, but slower
 - Avg opened/user +2.12%; weekly opened +11.65%; total opens 1.0M; openers 62.0K; avg opened/user 16.22.
- Takeaway
 - Growth is driven more by creation than opening; this means many users create notebooks they only use once

Do you want me to add this to the notebook?

Thought for 3s

Updated notebook

Added the summary as a Markdown section beneath the headline section "Notebooks - Q3 25 - Quarterly Adoption Report".

How can I improve this notebook further? What else could be worth...

Notebooks - Q3 25 - Quarterly Adoption Report

North Star Metrics: Notebooks created & opened (per user per week)

Top insights — last 12w vs prior 12w

- Creation intensity led: Avg created/user +27.71%; weekly created +34.08%; total created 124.4K; creators 19.1K; avg created/user 6.52.
- Engagement up, but slower: Avg opened/user +2.12%; weekly opened +11.65%; total opens 1.0M; openers 62.0K; avg opened/user 16.22.
- Takeaway: Growth is driven more by creation than opening; this means many users create notebooks they only use once

Adoption metric 1: Notebooks opened

Notebooks opened acts as a proxy metric for consumption

Opened per week

1 record Executed at: 6.9.2025, 10:35:13, Timeframe: 8.6.2025, 00:00:00 - 31.8.2025, 00:00:00, Scanned bytes: 582 GB

Week	Notebooks opened
15. Juni	~22K
22. Juni	~22K
29. Juni	~22K
6. Juli	~22K
13. Juli	~22K
20. Juli	~22K
27. Juli	~22K
3. Aug.	~22K
10. Aug.	~22K
17. Aug.	~22K
24. Aug.	~22K
31. Aug.	~22K

Avg. opened per user per week

1 record Executed at: 6.9.2025, 10:35:14, Timeframe: 8.6.2025, 00:00:00 - 31.8.2025, 00:00:00, Scanned bytes: 844 GB

Week	Avg. opened per user per week
15. Juni	~6.5
22. Juni	~6.5
29. Juni	~6.5
6. Juli	~6.5
13. Juli	~6.5
20. Juli	~6.5
27. Juli	~6.5
3. Aug.	~6.5
10. Aug.	~6.5
17. Aug.	~6.5
24. Aug.	~6.5
31. Aug.	~6.5

Interactive context-aware assistance

ASSISTANCE



Failure Rate Astroshop

Thought for 4s

I'll correlate the failed requests for the different services with other metrics. Afterwards, I'll take a look at related logs.

Correlated metrics

Analyzed related logs

It looks like the spike in failure rates is tied to specific issues in the `astroshop-payment` service.

Metrics correlation

- The spike in 5xx Errors aligns with an increase in Failed Requests in the 'Requests - Success vs Failed' tile.
- Latency metrics (p50, p90, p99) don't show a significant delay, so this seems to be a failure in processing rather than a performance bottleneck.

Error logs

The logs show repeated errors with the message: "Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted." This suggests a configuration or validation issue in the payment service when handling certain card types.

Recommendation

Review the card validation logic in the `charge.js` file, specifically around line 73, where unsupported cards like American Express are being rejected. Focus on:

- The conditions determining accepted card types.
- The state of key variables at this point in the code.

This should help pinpoint the root cause of the issue.

Debug in code **Recommended** View failure analysis View logs

Any recent changes to payments? What's the business impact?

Astroshop Dev...

Ask questions, navigate, take action. What's on your mind?

Dynatrace Intelligence uses AI. Check for mistakes.

Dashboards + Dashboard Upload

Astroshop Developer Self-Service Read-only

Service Filter variables will only affect a dashboard when referenced in tiles

LOGS WITH ERRORS: Use Open-With on a Log to Drill to the Trace - OR - Ask CoPilot to Explain it to you

Time	Status	Content	level	time	pid	hostname	service.name	err	msg
8.9.2025, 20:35:17	WARN	warn		1757356517648	17	payment-54469cbf9f-n59vx	payment	{ "type": "Error", "message": "Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.", "stack": "Error: Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.\n at module.exports.charge (/usr/src/app/charge.js:73:11)\n at process.processTicksAndRejections (node:internal/process/task_queues:105:5)\n at async Object.chargeServiceHandler [as charge] (/usr/src/app/index.js:21:22)" }	Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.
8.9.2025, 20:35:09	WARN	warn		1757356589672	17	payment-54469cbf9f-n59vx	payment	{ "type": "Error", "message": "Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.", "stack": "Error: Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.\n at module.exports.charge (/usr/src/app/charge.js:73:11)\n at process.processTicksAndRejections (node:internal/process/task_queues:105:5)\n at async Object.chargeServiceHandler [as charge] (/usr/src/app/index.js:21:22)" }	Sorry, we cannot process American Express credit cards. Only Visa or Mastercard are accepted.

LATENCY **ERRORS** **SATURATION**

Requests Latency_p50 Failed Requests CPU Usage %

Requests - Success vs Failed Latency_p90 5xx Errors Memory Used

Requests - Key Requests Latency_p99 4xx Errors Garbage collection suspension time

Interactive context-aware assistance

ASSISTANCE



The screenshot displays the Dynatrace Kubernetes Intelligence interface. On the left, a chat window titled 'Can you give me a summary of health alert...' contains an AI-generated response. The response identifies a 'Performance Slowdown' alert for the 'my-otel-demo-kafka' workload, specifically mentioning 'Amazon EC2 CPU Utilization High'. It provides details about the affected entity (AWS_EC2_INSTANCE-6A8B48361C04FFBA) and offers a summary of the issue. Below the summary, it asks 'Is there anything specific I should have a look at within the my-otel-demo-kafka workload?' and lists 'Smartscape Agent' and 'Kubernetes Agent' as available assistance options. A message input field is at the bottom of the chat.

The main interface shows a 'Kubernetes Overview' page for the 'my-otel-demo-kafka' deployment. It features a 'Workloads' table with 142 items, a 'Deployment utilization' section with CPU, Memory, and Pods metrics, and a 'Workload analysis' section indicating the workload is healthy. The 'Pods' section shows 1 pod with no restarting, failed, or warning events.

Workload	Age
dynatrace-oneagent-csi-driver	23 w 4 d
my-otel-demo-kafka	52 w 4 d
my-otel-demo-adservice	52 w 4 d
dynatrace-oneagent-csi-driver	23 w 4 d
scale-down-consumer-service	2 w 4 d
ebs-csi-node-windows	52 w
ebs-csi-node	52 w
cloud-node-manager-windows	42 w 1 d
aks-playground-staging-oneagent	23 w 4 d
eks-pod-identity-agent	69 w 3 d
csi-azuredisk-node-win	42 w 1 d
csi-blob-node	42 w 1 d
cloud-node-manager	42 w 1 d
azure-ip-masq-agent	27 w 3 d
csi-azuredisk-node	42 w 1 d
translationservice	63 w 1 d
recommendation	11 w 3 d
my-otel-demo-emailservice	52 w 4 d
problem-operator	54 w 3 d
metrics-server	42 w 1 d
email	11 w 3 d
manager	54 w 3 d
quote	11 w 3 d
ingress-nginx-controller	41 w 4 d
frontend	63 w 1 d
unguard-membership-service	69 w 3 d
unguard-profile-service	69 w 3 d
my-otel-demo-quoteservice	52 w 4 d
redis-cart	63 w 1 d

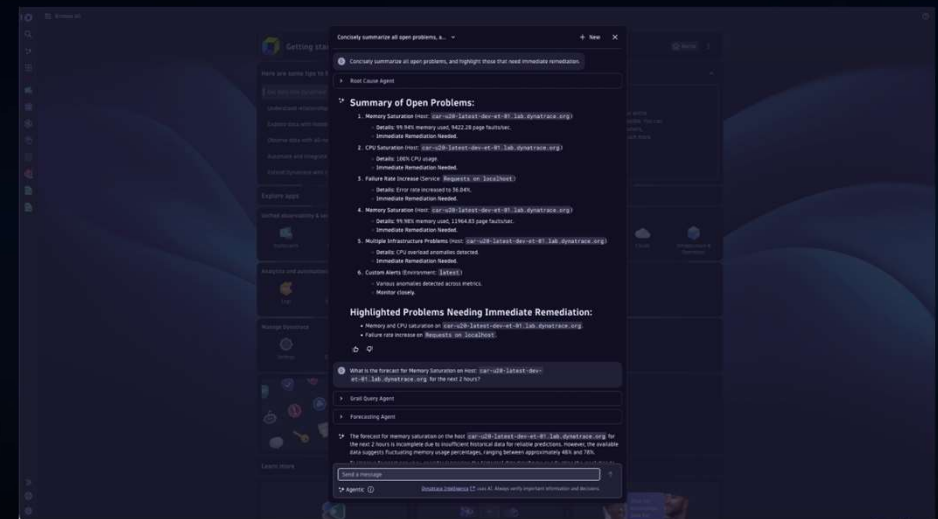
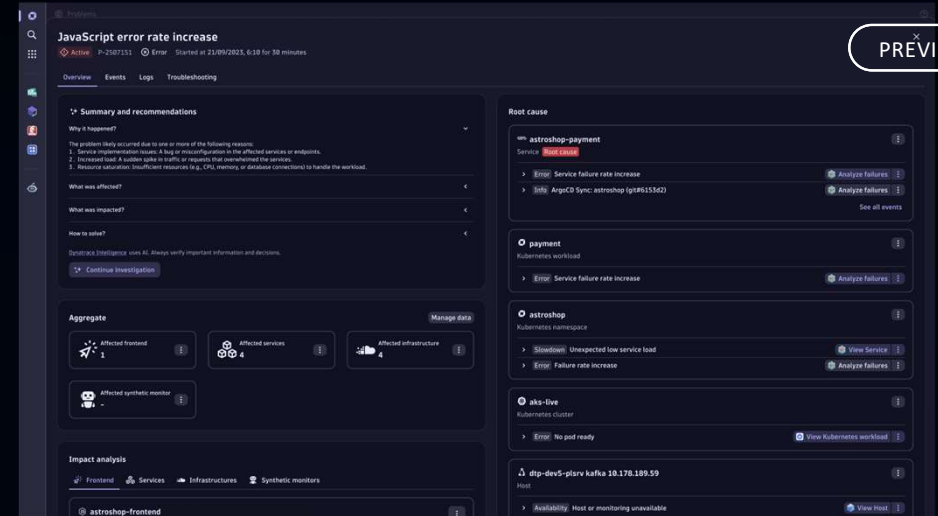
Assist Agents

Intelligence in every interaction



Assist Agents

- Explain any value or issue
- Automatically create context-aware summaries
- Interact with deterministic agents in conversation



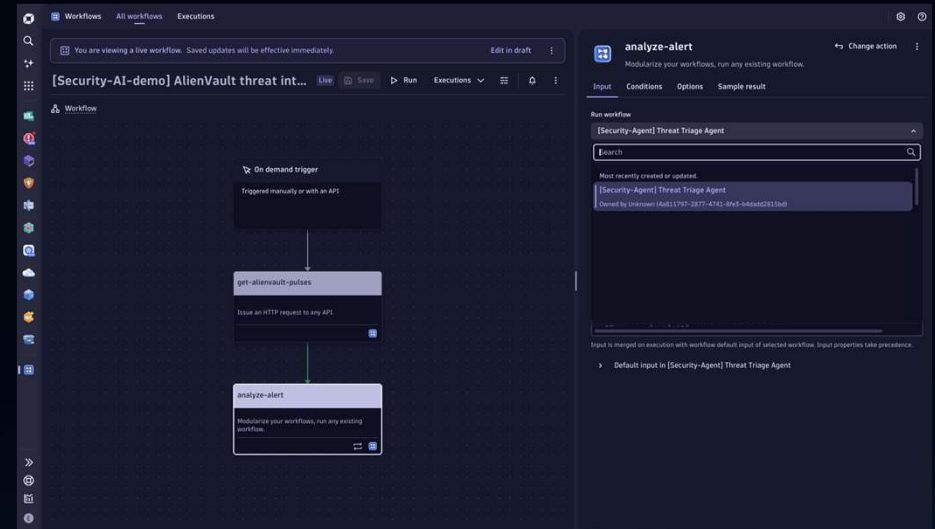
Agentic Workflows

Intelligence in every workflow



Agentic Workflows

- Instrument and supervise agents
- Customize and build your own team
- Retain control in your journey to autonomy



Team Agents

Intelligence in agents

EXAMPLE 1

Auto-remediate application crashes, reduce investigation effort, and apply and validate context-aware code changes

The screenshot shows a code editor window titled "[All] 'Android App Crash: Fatal Exception Detected'". The main content area displays a crash report and a suggested fix. The crash report states: "The crash in the Android Java app is identified as a fatal error caused by a native crash (error.type: crash) during the execution of the MainActivity.nativeAbort method. The stack trace indicates the issue originates from a native library (libnativecrash.so) and involves an abort signal triggered by the abort() function in the libc library. This crash impacts the app's stability, causing it to terminate unexpectedly, which can lead to a poor user experience. The root cause is likely improper handling of native code or an unhandled exception in the JNI layer."

Suggested Fix
To address this issue, ensure proper error handling in the native code and validate inputs before invoking native methods. Below is a code snippet to handle potential errors:

```
1 try {
2     // Call native method
3     nativeAbort();
4 } catch (Exception e) {
5     // Log the error and prevent crash
6     Log.e("MainActivity", "Error in nativeAbort: " + e.getMessage());
7     // Optionally, show a user-friendly message
8     Toast.makeText(this, "An error occurred. Please try again.", Toast.LENGTH_SHORT).show();
9 }
```

Additionally, review the native library (libnativecrash.so) for potential issues and implement safeguards to prevent abort signals.

EXAMPLE 2

Discover threat exposure, leverage deterministic insights and automate preventive remediation before they become vulnerabilities

The screenshot shows a security agent threat triage interface. The main content area displays a detailed threat summary for a threat identified as "React2Shell Deep Dive: CVE-2025-55182 Exploit Mechanics". The summary includes the following information:

- Threat Summary:** CVE-2025-55182 Exploit Mechanics
- Score Criteria:** Vulnerabilities exist with matched indicators (CVE-2025-55182). No detections, logs, or spans matched, reducing the exposure score. Vulnerabilities are critical, but lack of evidence in detections, logs, or spans lowers the risk of active exploitation.
- Summary Per Data Category:** Vulnerabilities
- Vulnerabilities:** CVE-2025-55182
- Vulnerable Components:** next (next.js)
- Matched Indicators:** CVE-2025-55182
- Issue URL:** [Github Issue] (https://github.com/repsec-ai/initiative-dev/unguard/issues/624)
- Detections:** No detections matched. Investigate further to ensure no active exploitation is occurring.
- Logs and Spans:** No logs or spans matched. Continue monitoring for any suspicious activity.
- Vulnerabilities Details:** CVE-2025-55182
- Affected Software Components:** next

Agentic Ecosystem

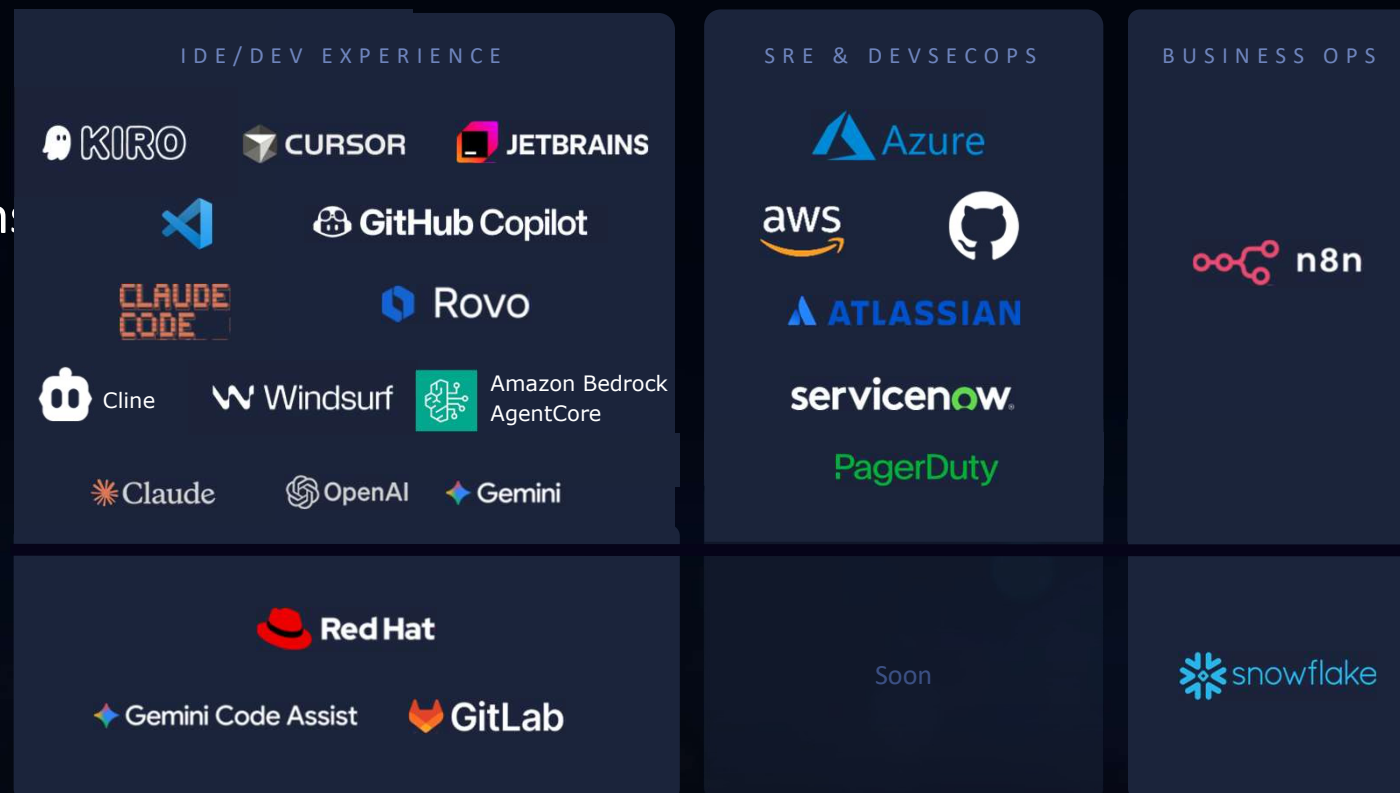
Connect and collaborate across your ecosystem



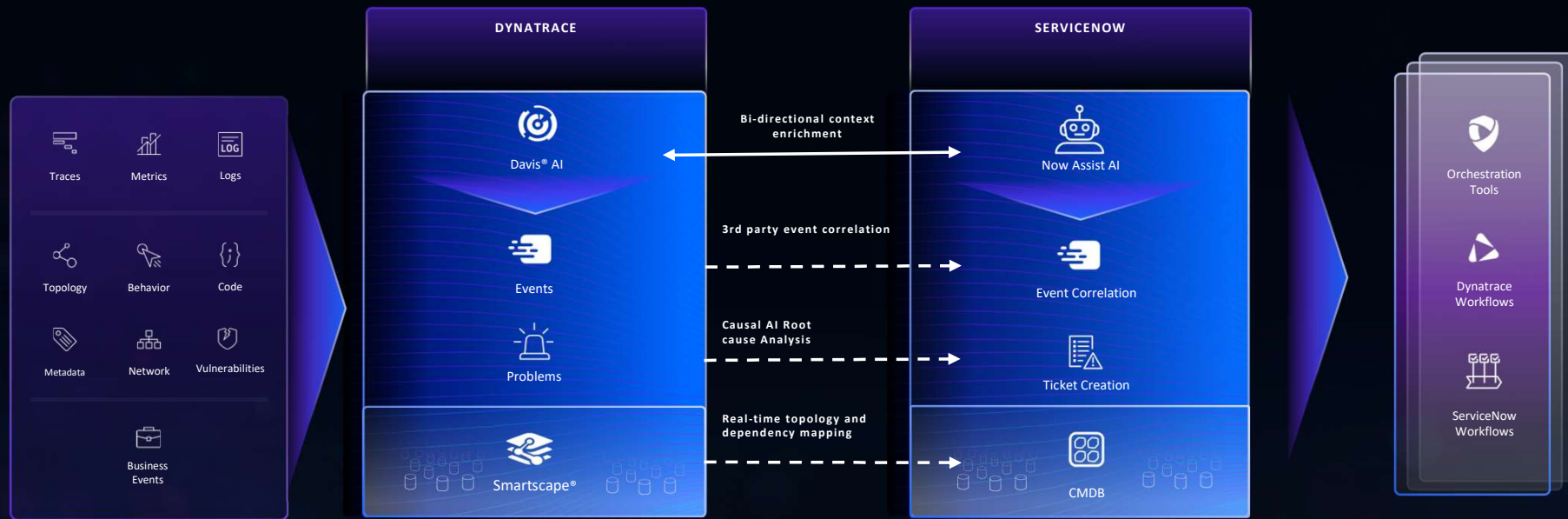
Ecosystem Agent Interactions

Includes MCP Server

- Explain any value or issue
- Automatically create context-aware summaries
- Interact with deterministic agents in conversation

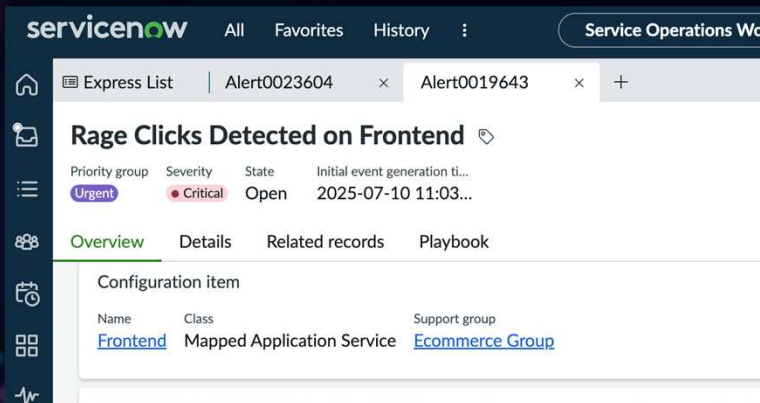


ServiceNow and Dynatrace coming together to drive your operations autonomously



ServiceNow AI Insights

- Understand impact & root cause without leaving ServiceNow UI
- Pull in subject matter experts
- Notify technical and business stakeholders



NowAssist

Answers generated by AI. Review for accuracy.

what's going on with the frontend service? are users or other services impacted?

The report shows the user impact as Total Impacted Users: 43034 (4.252% of total users). The impacted entities by this alert are:

- Frontend.

what are your root cause theories? what is the most likely root cause?

Root cause theories for this alert include a JavaScript error where the function 'saveInfo' is not recognized, leading to a TypeError, and an external API error that contributed to anomalies in page load times and interaction delays. The most likely root cause is the JavaScript error involving 'saveInfo', as it is directly correlated with the incident description.

Ask Now Assist to...

AI Agents



ServiceNow Data

- Recent Changes
- Team Ownership
- Active & Past Incidents
- KB Articles, Post-Mortems
- Service Maps

Dynatrace Observability Data

- ### Telemetry
- Analyze business and user impact with Real User Monitoring
 - Catch cascading failures with transaction traces
 - Search logs and metrics for impacted entities
 - Detect anomalous metrics and golden signals





A PLATFORM EVOLUTION

Dynatrace Intelligence - Agents



Expand impact with powerful ecosystem

INTEGRATION



Ecosystem Agent Interactions

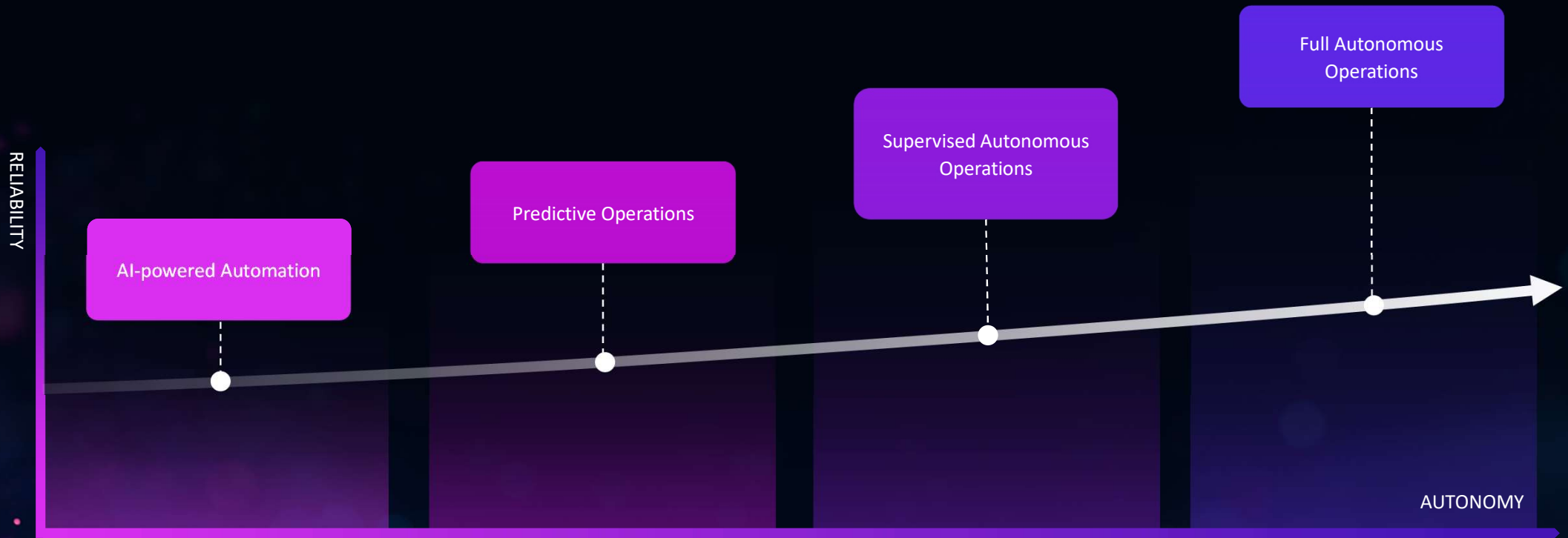
Incl. Remote MCP Server

Drive reliable operations



RELIABLE AND TRUSTWORTHY MULTI-STEP AUTONOMY

Journey to reliable autonomy in operations



DYNATRACE INTELLIGENCE

Act on answers, not guesses

