



Agentic AI Foundation

AGENTCon + MCPCon North America

Where the **agentic stack** is **being built**.

MCPCon Beginner Breakout · Oct 23 · 1:45 PM · LL20A

One Server, Many Apps

A Composable MCP Architecture for Observability

Shenoy Pratik · Kyle Hounslow · OpenSearch

Are Dashboards Dead?

Let's find out together by the end of this session.



Where the **agentic stack** is being built.

#AGNTCon #MCPCCon

Who we are



Shenoy Pratik

Software Engineer, OpenSearch



Kyle Hounslow

Sr. Engineer, AWS Open Source Observability

Where the **agentic stack** is being built.

#AGNTCon #MCPCCon

Agenda

- 1 The observability problem: the 3 AM page
- 2 MCP Apps: what, why and how
- 3 Demo
- 4 Composable MCP Apps: six shared capabilities
- 5 What's next

3 AM: the SRE's journey (pre Agentic AI)

An on-call engineer, or an AI agent, crosses four tools. Every hop drops context.

21 minutes carrying context by hand

02:58



which cluster?

03:04



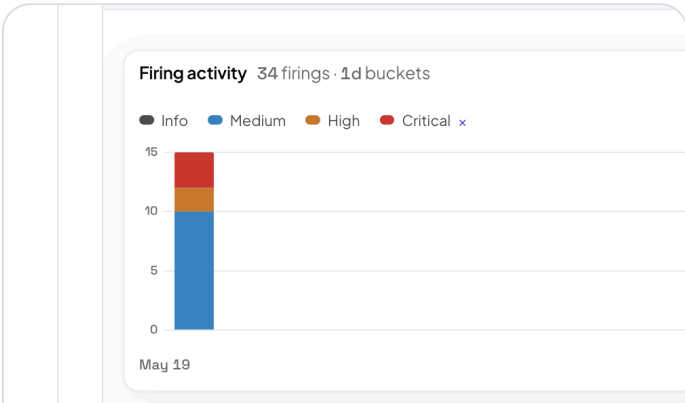
time range re-typed

03:11

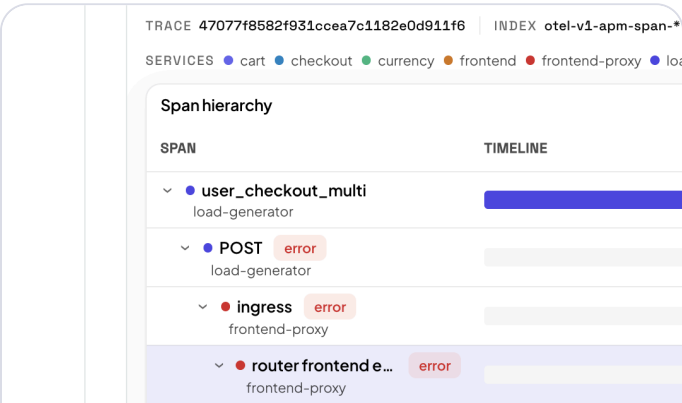


tracelId copy-pasted

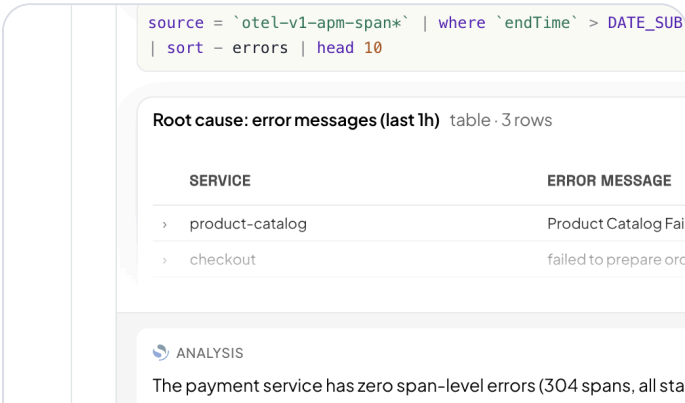
03:19



Alert fires
Checkout errors breach the SLO



Find the trace
The slow, failing span



Read the logs
What that span emitted



Check the trend
Metrics and blast radius

Humans hop between browser tabs, re-authenticate and re-enter filters.

Agents face a new API, auth and ID format per tool.

Both need to see the evidence and verify the other's conclusion.

3 AM · 02:58

3 AM, with an agent in the loop

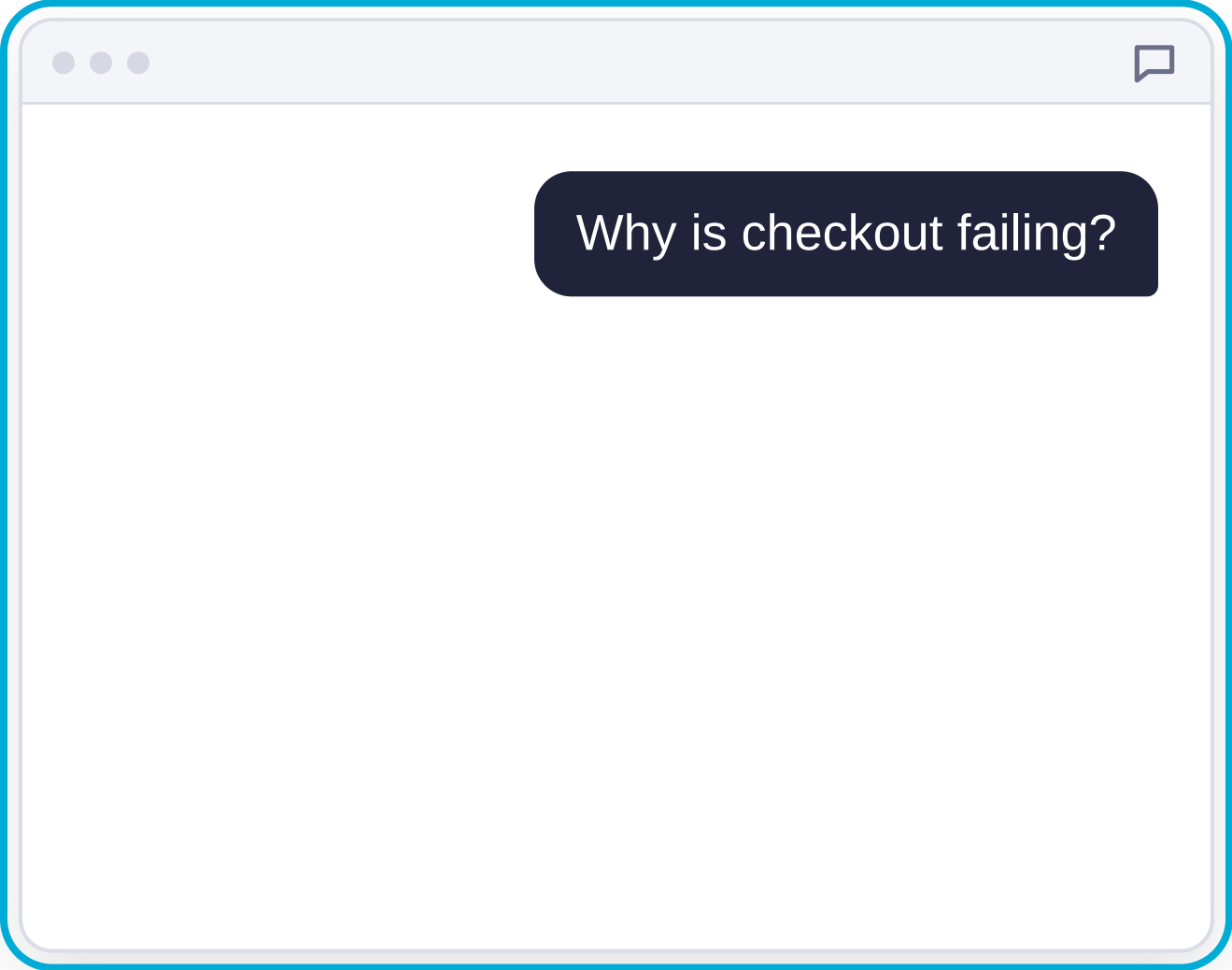


checkout 5xx

3 AM · 03:00

3 AM, with an agent in the loop

 checkout 5xx

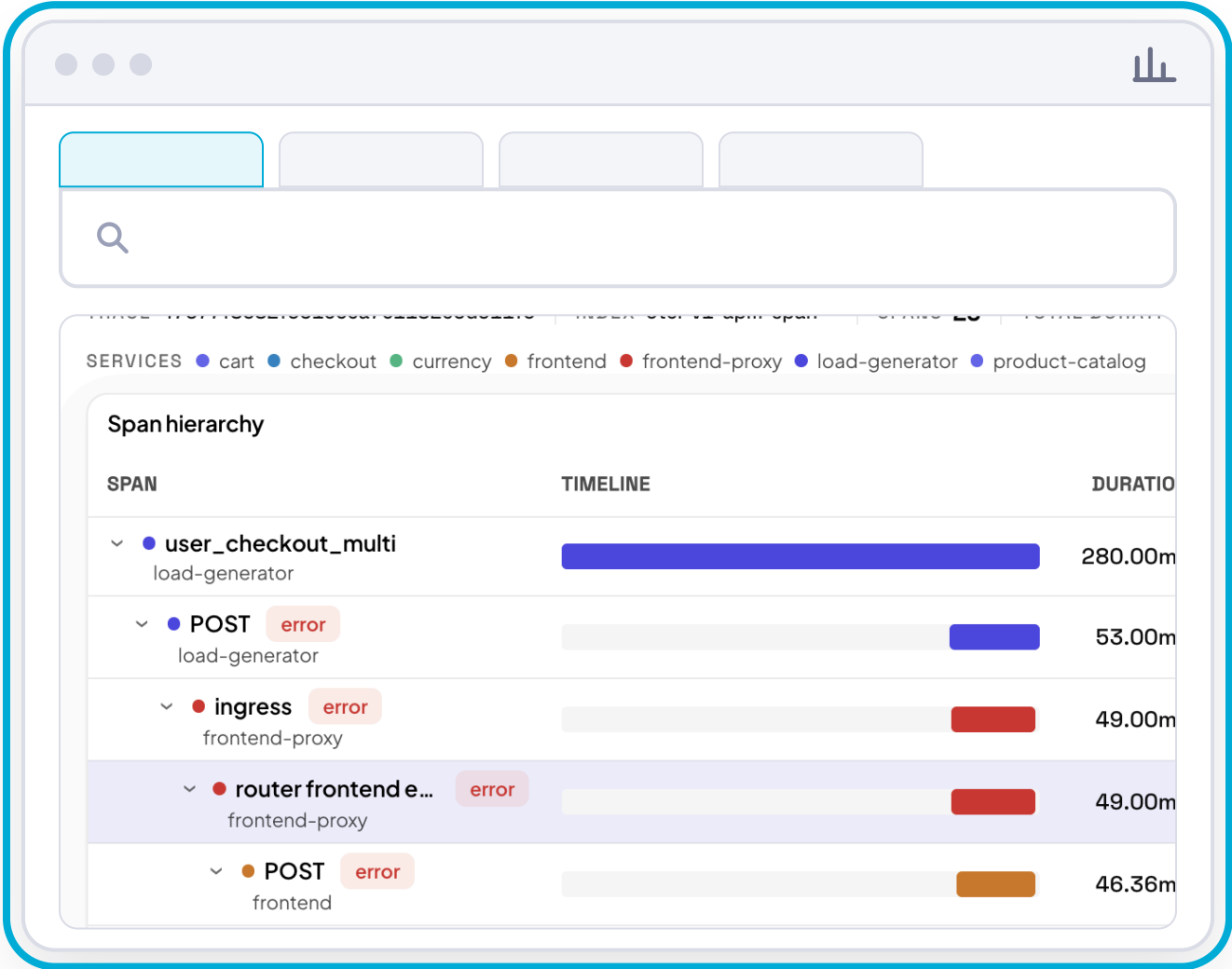


Why is checkout failing?

3 AM, with an agent in the loop

 checkout 5xx

Why is checkout failing?



3 AM, with an agent in the loop

 checkout 5xx

Why is checkout failing?

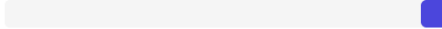
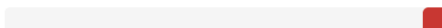
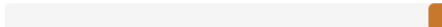
payment-db timeout

 a3f9...c21

SERVICES ● cart ● checkout ● currency ● frontend ● frontend-proxy ● load-generator ● product-catalog

Span hierarchy

SPAN	TIMELINE	DURATION
▼ ● user_checkout_multi load-generator		280.00m
▼ ● POST error load-generator	 	53.00m
▼ ● ingress error frontend-proxy	 	49.00m
▼ ● router frontend e... error frontend-proxy	 	49.00m
▼ ● POST error frontend	 	46.36m

3 AM, with an agent in the loop

🔔 checkout 5xx

Why is checkout failing?

payment-db timeout

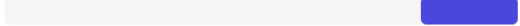
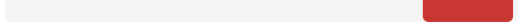
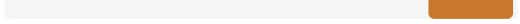
 a3f9...c21

15m





SERVICES ● cart ● checkout ● currency ● frontend ● frontend-proxy ● load-generator ● product-catalog

SPAN	TIMELINE	DURATION
user_checkout_multi load-generator		280.00m
POST error load-generator		53.00m
ingress error frontend-proxy		49.00m
router frontend e... error frontend-proxy		1h
POST error frontend		46.36m

3 AM, with an agent in the loop

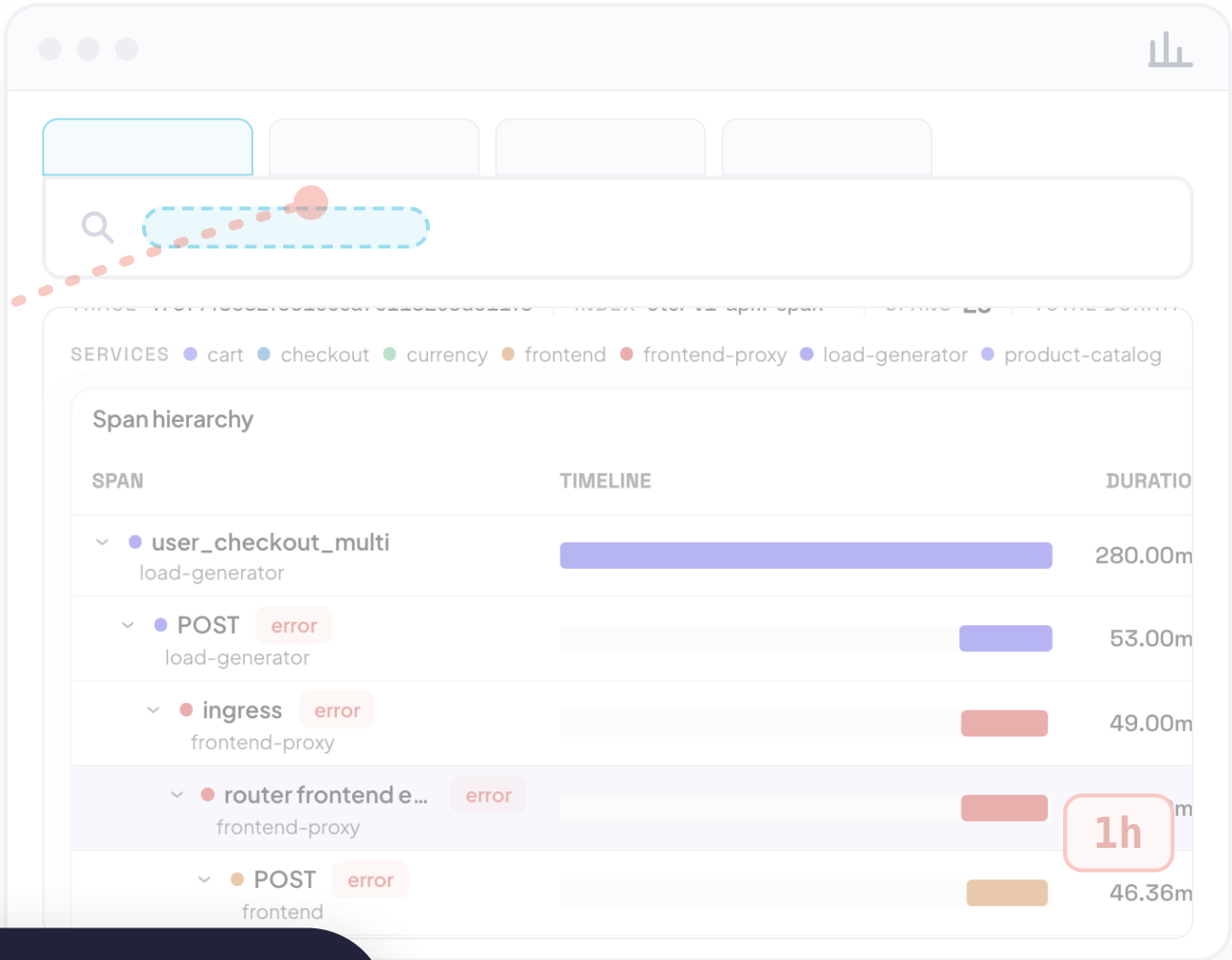
🔔 checkout 5xx

Why is checkout failing?

payment-db timeout

 a3f9...c21

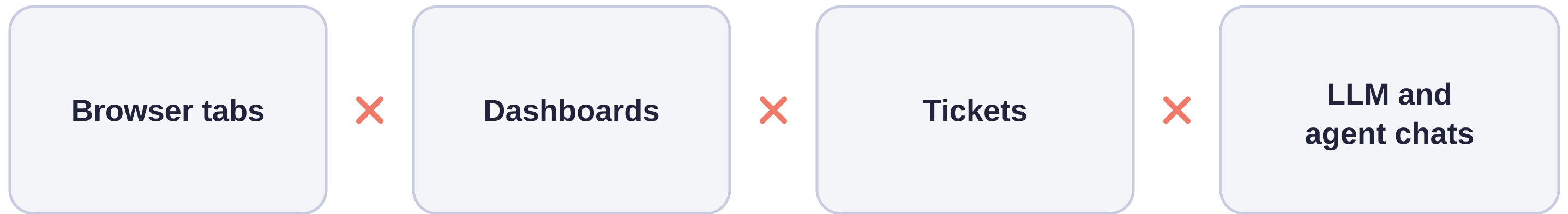
15m



≠

Is it right?

Every hop loses context



**Losing context breaks the chain of thought
and interrupts human reasoning and verification.**

Are Dashboards Dead?

Not yet. Chat interfaces aren't convincing enough on their own.

An abstract graphic at the bottom of the slide features a network of glowing lines and dots in various colors (red, orange, yellow, green, blue, purple) against a dark background. The lines and dots are arranged in a way that suggests a complex, interconnected system, possibly representing an 'agentic stack' or a data network.

Where the **agentic stack** is being built.

#AGNTCon #MCPCCon

A tool call that comes back with a UI

MCP server with tools

traces_details(traceId)

TOOL OUTPUT

```
{
  "traceId": "47077f8582f931ccea7c1182e0d911f6",
  "spanCount": 20,
  "durationMs": 280,
  "spans": [
    {
      "name": "user_checkout_multi",
      "service": "load-generator",
      "status": "OK",
      "durationMs": 280
    },
    {
      "name": "POST",
      "service": "load-generator",
      "status": "ERROR",
      "durationMs": 53
    },
    {
      "name": "ingress",
      "service": "frontend-proxy",
      "status": "ERROR",
      "durationMs": 49
    },
    {
      "name": "router frontend egress",
      "service": "frontend-proxy",
      "status": "ERROR",
      "durationMs": 49
    },
    {
      "name": "POST /api/c...",
      "service": "frontend",
      "status": "ERROR",
      "durationMs": 46.36
    },
    {
      "name": "POST /api/c...",
      "service": "frontend",
      "status": "ERROR",
      "durationMs": 46.05
    }
  ]
}
```

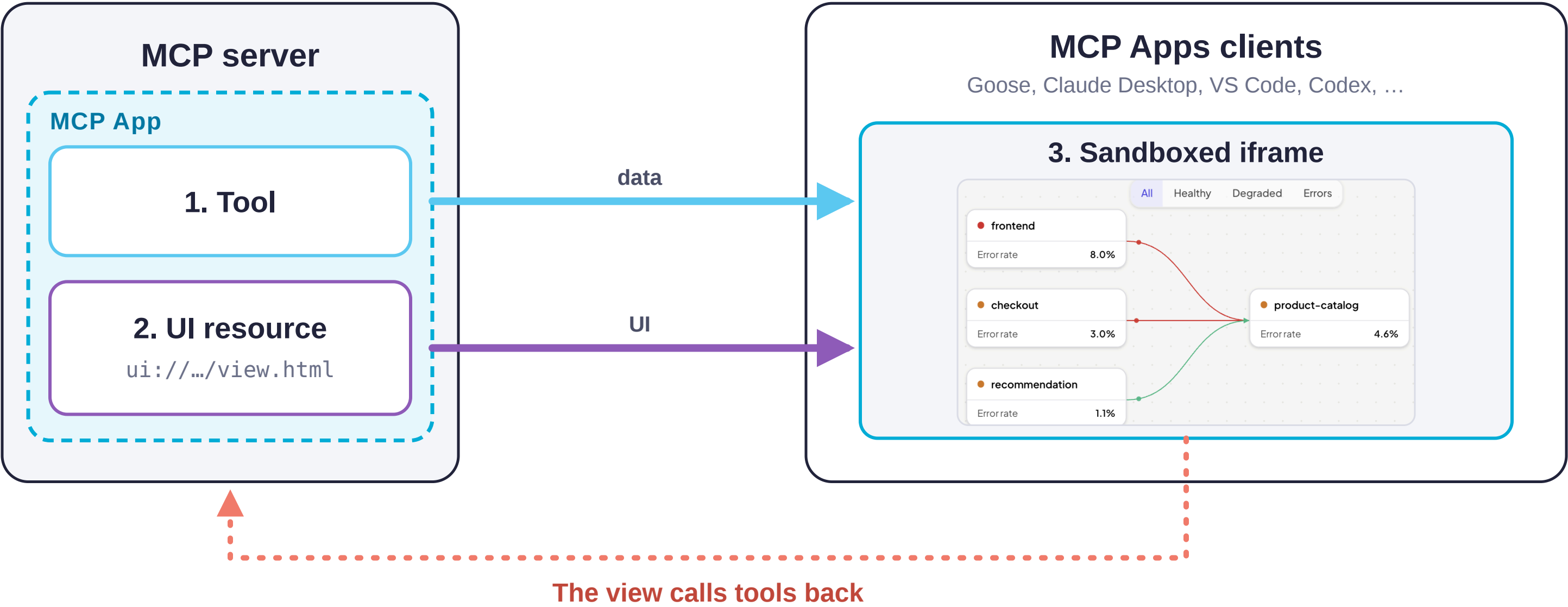
MCP server with Apps (tool + UI)

traces_details(traceId)

20 spans · 280 ms · checkout failed

user_checkout_multi	load-generator	280.00ms
POST	load-generator	53.00ms
ingress	frontend-proxy	49.00ms
router frontend e...	frontend-proxy	49.00ms
POST	frontend	46.36ms
POST /api/c...	frontend	46.05ms

How an MCP App is built



An MCP App is a list of routes

```
// apps/traces/index.ts
export const tracesApp = defineApp({
  id: "traces", title: "Traces",
  routes: [finderRoute, detailsRoute,
           crossSignalJoinRoute, serviceMapRoute,
           dependenciesRoute, healthOverviewRoute],
});

// apps/index.ts: the whole registry
export const apps = [pplApp, reportApp, tracesApp,
  agentTraceApp, alertsFeedApp, metricsApp,
  instrumentationScoreApp, sloApp];
```

schema.ts

input and props

tool.ts

fetch, then text

view.tsx

widget (optional)

DEMO

Alert to root cause, in one thread

Where the **agentic stack** is being built.

#AGNTCon #MCPCCon

THE STACK WE BUILT ON

OpenSearch for Agents

- Open-source observability, search, analytics and vector engine
- Strong capabilities for agentic, hybrid and RAG search
- Observability Stack: one solution for monitoring and evaluating agents

2021



Launch

Apache 2.0, community-driven

2024



Linux Foundation

Vendor-neutral foundation

2025



OpenSearch 3.0

First major release since 2022

2.4B+

downloads

3,000+

contributors

400+

organizations

130+

repos

1.4M

monthly site views

4,000+

Slack members

7,000+

forum members

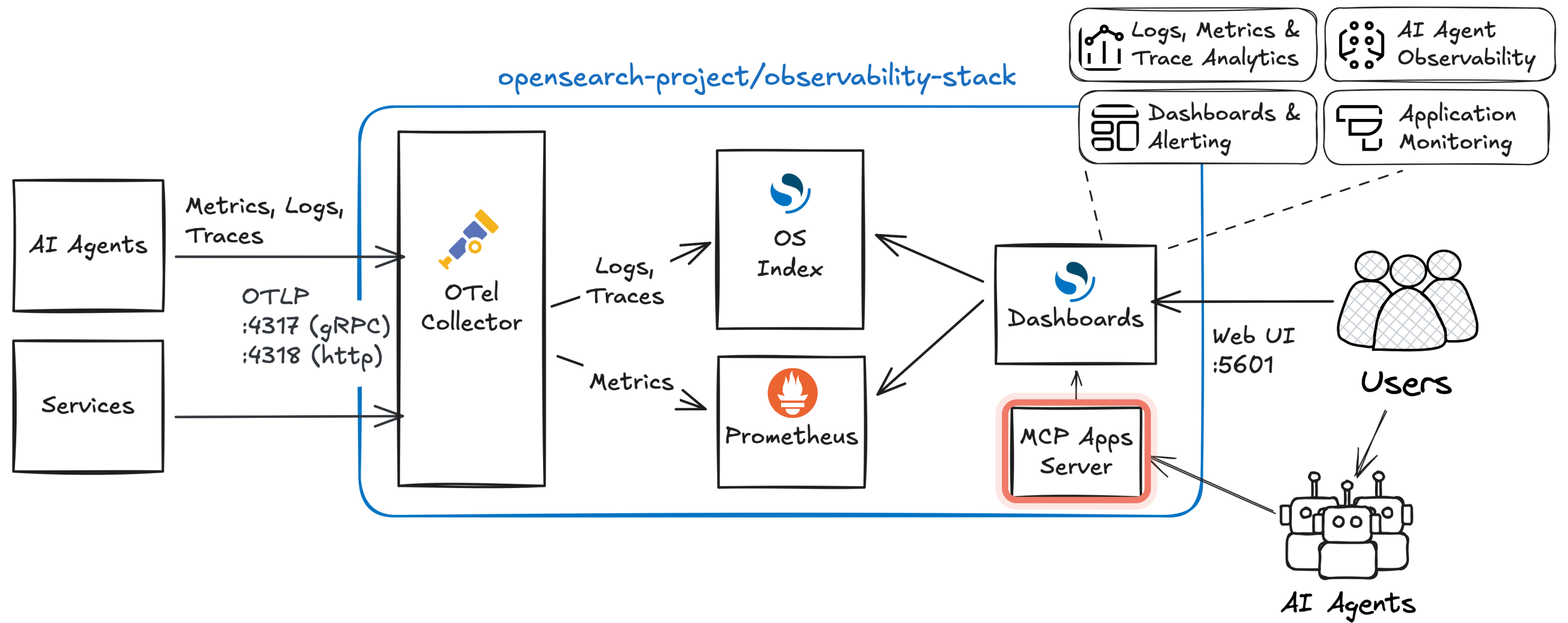
90+

partners

Where the **agentic stack** is being built.

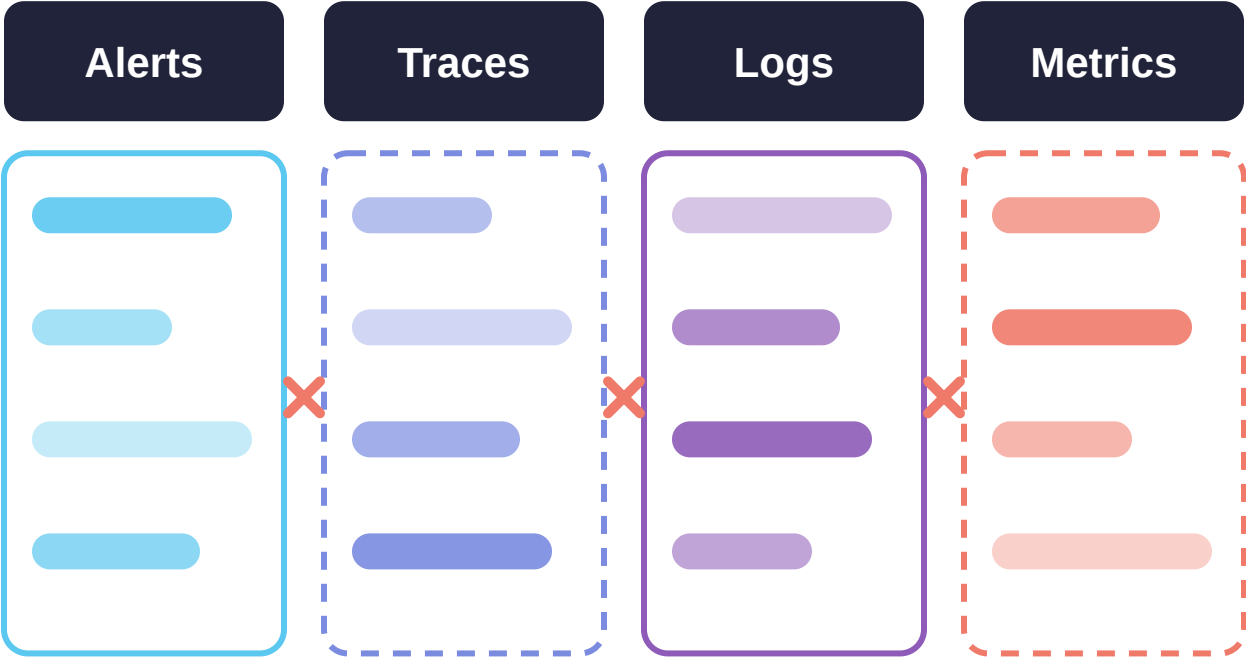
#AGNTCon #MCPCCon

Humans and agents read from the same stack



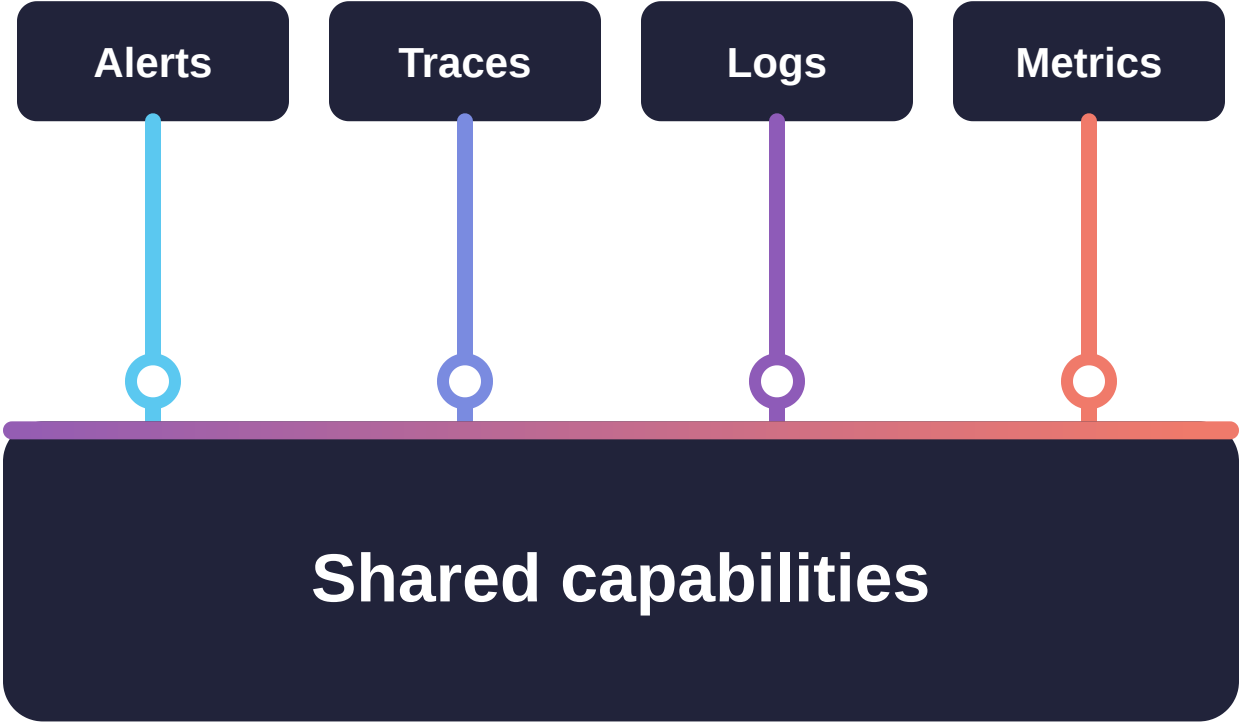
Composable MCP Apps are the way to go

Vertical slices

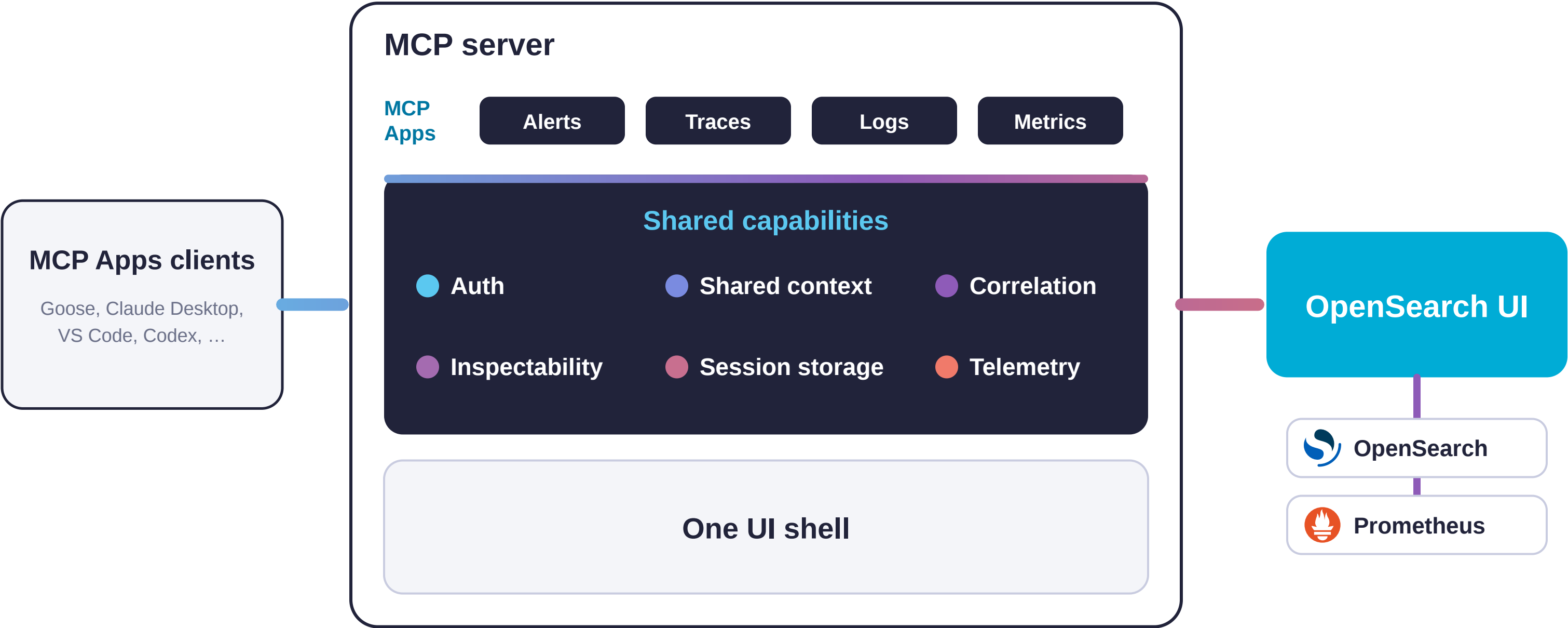


Each one owns its auth, routing, time and UI

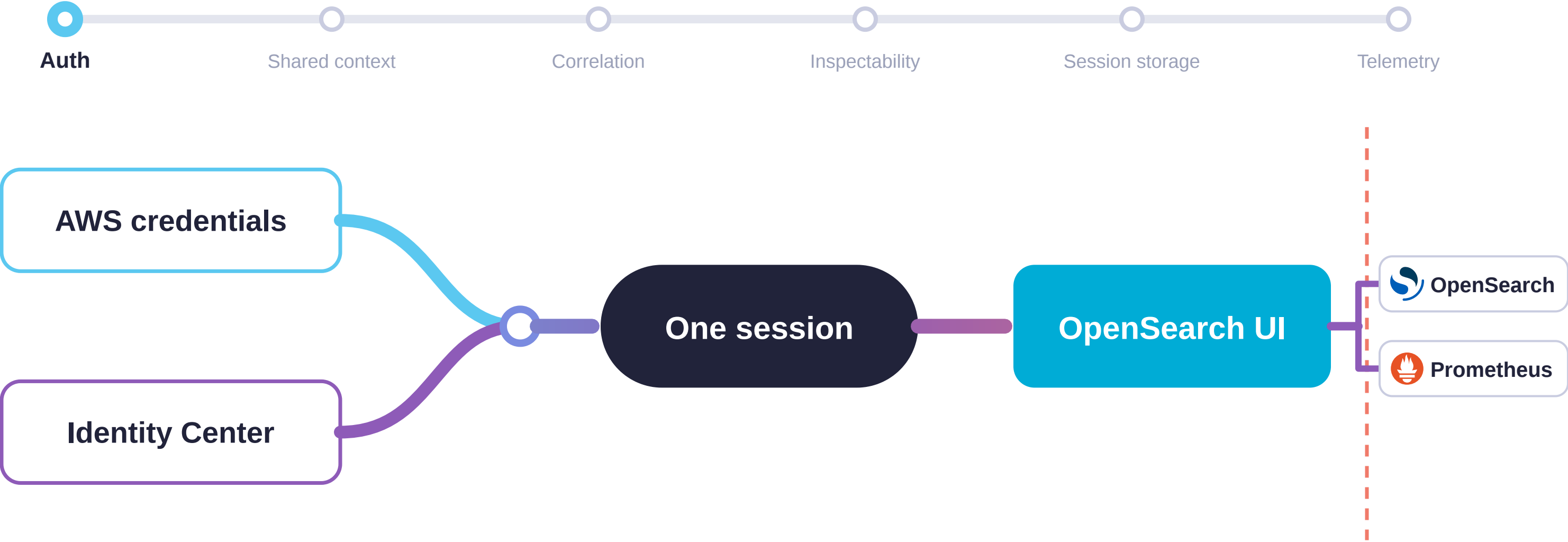
Composable



Six capabilities live in the server

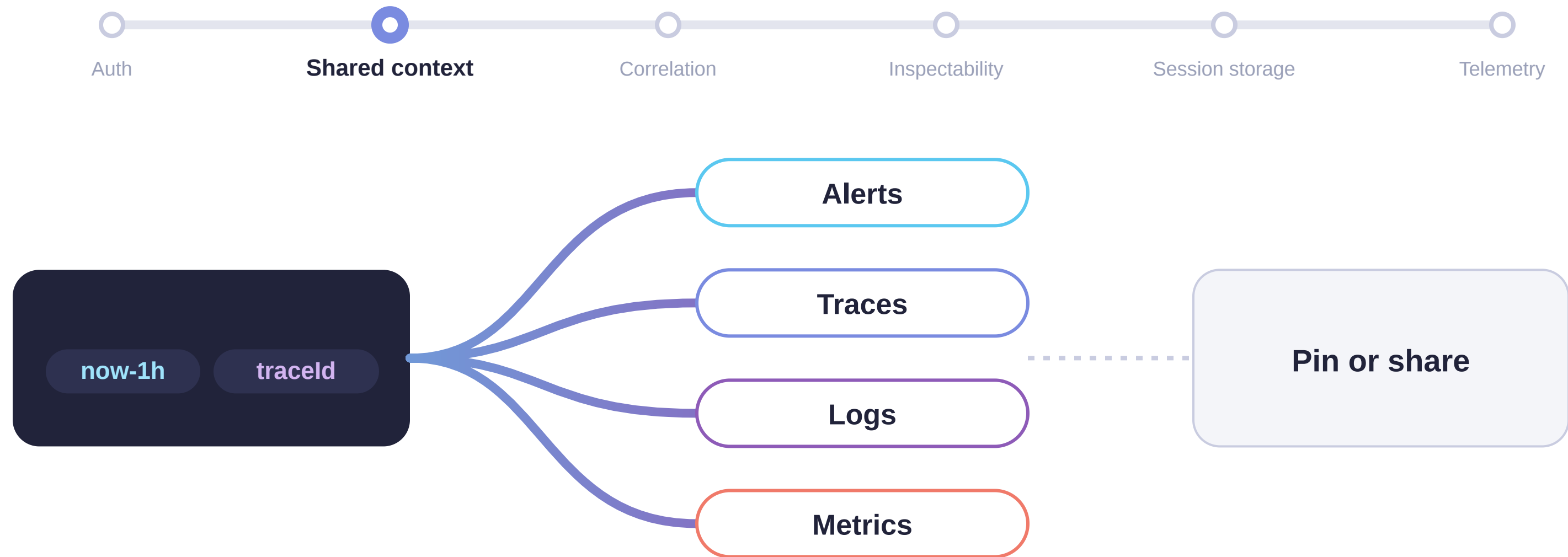


Auth



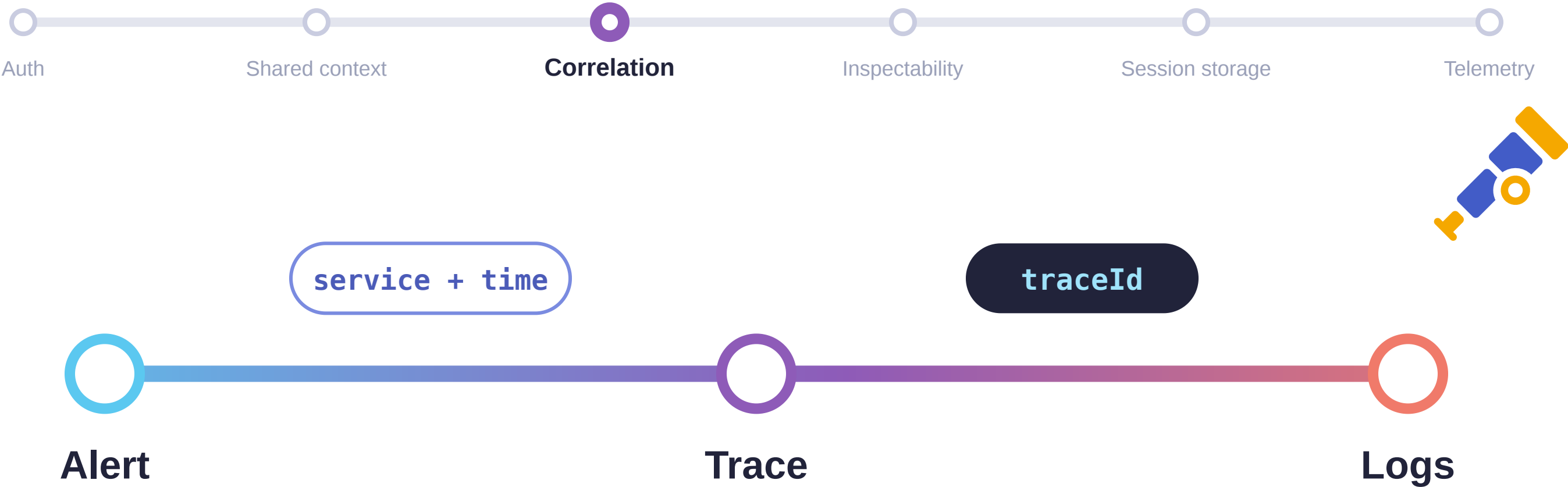
Every MCP App reuses your OpenSearch UI sign-in and its data access.

Shared context



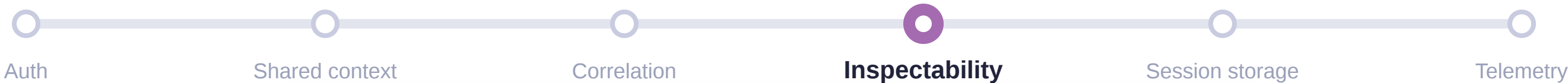
MCP Apps hand off the time window and IDs, [via the calling agent or a click](#).

Correlation



MCP Apps follow OpenTelemetry semantic conventions, so agents can correlate across signals and MCP Apps.

Inspectability



REPORT OpenSearch Observability [View in OpenSearch ↗](#)

Error count by service (last 1h) bars · 10 rows

frontend	<div></div>	48%	1,228
frontend-proxy	<div></div>	24%	616
load-generator	<div></div>	12%	309
product-catalog	<div></div>	12%	307
checkout	<div></div>	2%	58
travel-planner	<div></div>	1%	20
weather-agent	<div></div>	1%	18
flagd	<div></div>	1%	16
product-reviews	<div></div>	0%	6
recommendation	<div></div>	0%	6

```
source = `otel-v1-apm-span*` | where `endTime` > DATE_SUB(NOW(), INTERVAL 1 HOUR) | where `status.code` = 2 | stats count() as errors by `serviceName` | sort - errors | head 10
```

Root cause: error messages (last 1h) table · 3 rows

SERVICE	ERROR MESSAGE	COUNT
> product-catalog	Product Catalog Fail Feature Flag Enabled	308
> checkout	failed to prepare order: failed to get product #OLJCESPC7Z'	29

Click to expand

ANALYSIS

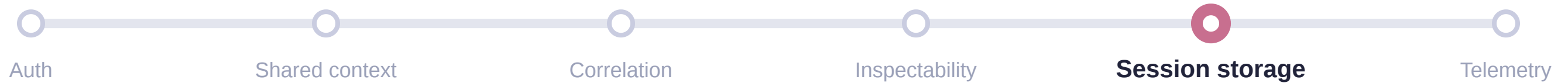
The payment service has zero span-level errors (304 spans, all status.code=0, p99 1.46ms). The critical OtelDemoPaymentFailures alert is a false positive — the real root cause is the productCatalogFailure feature flag being active in flagd, causing ProductCatalogService/GetProduct to throw 308 errors/hr. This cascades into CheckoutService/PlaceOrder failing with 'failed to prepare order: failed to get product #OLJCESPC7Z' (29 errors/hr), blocking orders before they ever reach the payment service. The fix is to disable the productCatalogFailure feature flag in flagd.

link back

the query

the analysis

Session storage: re-fetch or persist?



WHAT WE DO TODAY

Re-fetch

Re-run each view from its saved query

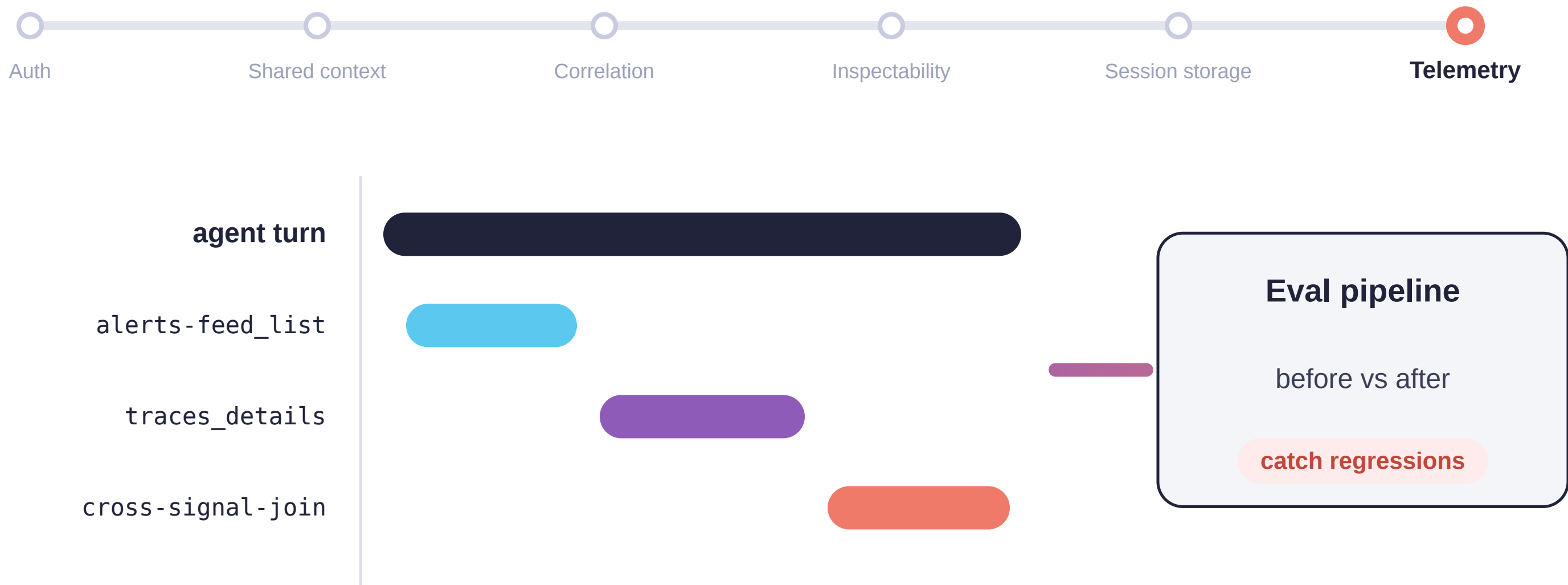
- + Always current, nothing sensitive stored
- Old investigations can change or expire

Persist

Store results with the session

- + Replay past RCAs, share as a report
- You now store data and need retention rules

Telemetry: instrument MCP Apps, then run evals



Measure how each MCP App change shifts agent behavior.

WHAT SHIPS TODAY

One install covers every signal

Alerts

Logs

Traces

Metrics

Topology

SLOs

Agent observability

Stack health

Instrumentation score

Where the **agentic stack** is being built.

What this buys SREs

- | Faster root cause
- | Correlation built in
- | No new plumbing
- | One sign-in
- | Verifiable answers
- | Traceable agents

Patterns from the MCP Apps docs

What we use

- Passing context to the model
- Errors back to the model
- Following the host theme

What we'd add next

- Lowering perceived latency
- Fullscreen display mode
- Progress while queries run

GitHubSpecification

Patterns /

MCP Apps Patterns

This document covers common patterns and recipes for building MCP Apps.

Tools that are private to Apps

Set `Tool._meta.ui.visibility` to `["app"]` to make tools only callable by Apps (hidden from the model). This is useful for UI-driven actions like updating server-side state, polling, or other interactions that shouldn't appear in the model's tool list.

```
registerAppTool(
  server,
  "update-quantity",
  {
    description: "Update item quantity in cart",
    inputSchema: { itemId: z.string(), quantity: z.number() },
    _meta: {
      ui: {
        resourceUri: "ui://shop/cart.html",
        visibility: ["app"],
      },
    },
  },
  async ({ itemId, quantity }) => {
    const cart = await updateCartItem(itemId, quantity);
    return { content: [{ type: "text", text: JSON.stringify(cart) }] };
  },
);
```

Copy

> Settings

> On This Page

MCP Apps Patterns

Tools that are private to Apps

Polling for live data

Reading large amounts of data via chunked tool calls

Giving errors back to the model

Serving binary blobs via resources

Configuring CSP and CORS

Adapting to host context (theme, styling, fonts, and safe areas)

Entering / exiting fullscreen

Passing contextual information from the App to the model

Sending large follow-up messages

Persisting view state

Pausing computation-heavy views when offscreen

Lowering perceived latency

apps.extensions.modelcontextprotocol.io · Patterns

FINALLY

Are Dashboards Dead?

We don't know!

Agents don't need dashboards to find the problem.

Humans still need a chart before they believe the answer.

With MCP Apps, the evidence arrives with the answer.



Where the **agentic stack** is being built.

#AGNTCon #MCPCCon

TRY IT

Scan to get started



OpenSearch Observability Stack

observability.opensearch.org



AWS OpenSearch MCP Apps docs

[docs.aws.amazon.com](https://docs.aws.amazon.com/mcp/apps) · MCP Apps

Where the **agentic stack** is being built.

#AGNTCon #MCPCon



Agentic AI Foundation

AGENTCon + MCPCon North America

Where the agentic stack is being built.

Thank you!

Questions?

Shenoy Pratik

linkedin.com/in/pshenoy36

Kyle Hounslow

linkedin.com/in/kylehounslow

Demo storyboard

