KubeCon CloudNativeCon
North America 2026

Agents Fail. Workflows Recover.

Durable agent orchestration with Cadence

Diana Zawadzki
Software Engineer
Cadence team, Uber

Shijie Sheng
Staff Software Engineer
Cadence team, Uber

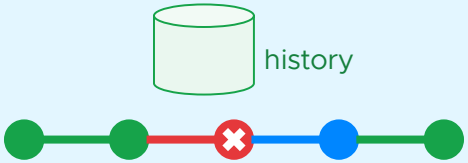
AI samples

Python SDK

Code that outlives its process

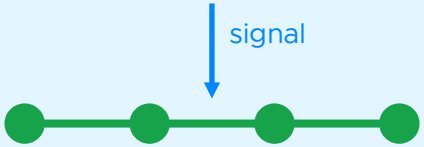
Your Workers run the code. Cadence records the progress.

Survives crashes



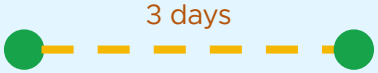
Completed Activity results are saved.
After a crash, replay rebuilds state.

Accepts input



Workflows accept Signals.
Workers answer Queries.

Waits for days



Cadence persists timers and waits.
Waits survive Worker restarts.

We've built for failure before

Disks fail. **RAID**

Adds redundancy.

Packets drop. **TCP**

Retransmits.

Machines fail. **Replicated logs**

Preserve committed state.

Workers restart.

Durable execution

Preserves recorded progress.

Failure is normal

What matters is how you design for it.

01 Rate limits
An API returns 429.

02 Repeated tool calls
The agent asks for the same lookup.

03 Unsupported conclusions
A draft claims more than the logs show.

04 Waiting for review
The reviewer is away.

05 Worker failure
The process disappears.



Cadence at Uber

4,000+

Cadence use cases at Uber

Shared workflow infrastructure across Uber

Infrastructure

rollouts, rollbacks, DB backup and restore

Payments

payments, tips, disputes, billing

Product

onboarding, compliance checks, trip reports

AI workloads

scheduled and background tasks

Uber's internal Agent Execution Service (AES) runs agents on Cadence.

Cadence beyond Uber

NetApp

Fleet maintenance

Tens of thousands of hosts on Instaclustr.

DOORDASH

ETA, fulfillment and orders

Plus storage, catalog, photos and ads.

CLOUDERA

Deployments, backup and restore

Control plane for core data services.

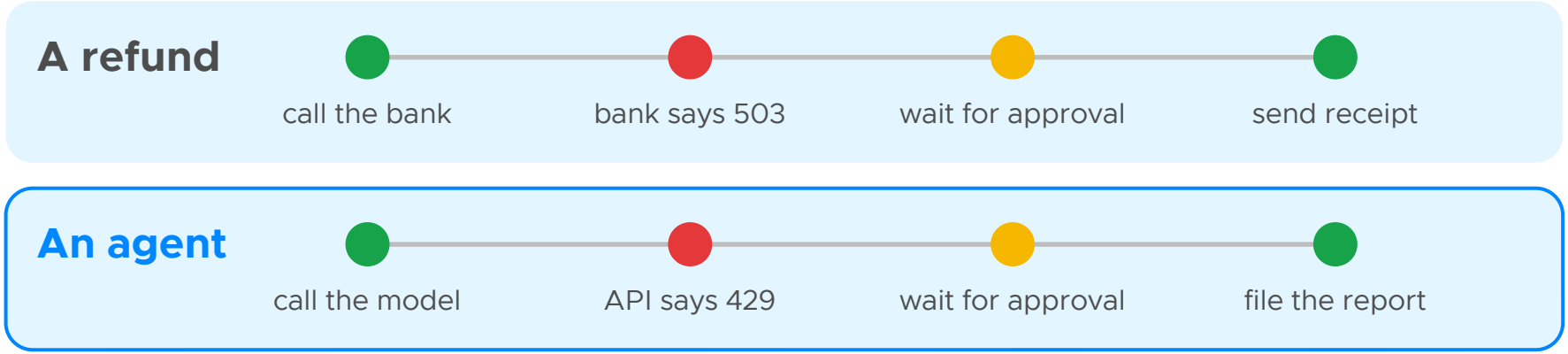
athenahealth

Revenue cycle workflows

Durable, high-throughput, fault-tolerant execution.

Using Cadence? Add your company to ADOPTERS.md in the Cadence repo.

Agent workflows are also just workflows.



Both call unreliable APIs and wait for people.

Recorded progress gives both a place to resume.

An on-call agent investigates for you

service-error-analyser: a service error analyser we're turning into an on-call agent with Cadence.

Trace › **Read logs** › **Draft report** › **Engineer reviews**

Human review is a Workflow wait

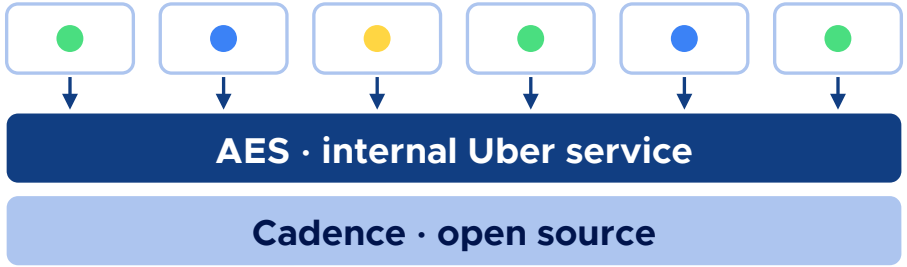
Draft › **Wait** › **Revise / Approve**

Signal

Uber's Agent Execution Service

Teams write the agent.

Cadence preserves recorded progress when a Worker stops.



830,000+

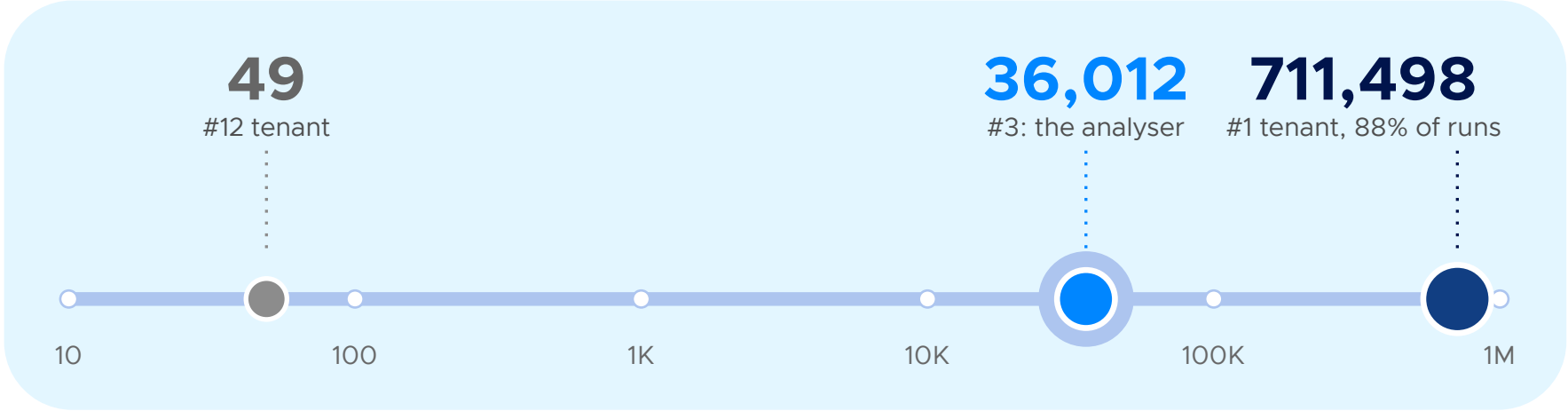
on-demand runs ended · Sep 2026 · PHX
Workflow type: AESAgentWorkflow

Background agent work

AES runs background tasks and scheduled jobs on Cadence.

AES contributors: Ethan Lee, Sean Po, Daniel Yao and the Michelangelo team

Agent workloads vary widely by tenant



32 tenants. Three examples shown on a logarithmic scale.

Agent runs per tenant, 30 days to Sep 14.

Uber for Business's error analyser

Gateway › Service › Service › Failure

36K runs — **6,023 timed out**

Sept: 30 days to Sep 14 · 17% timed out

5,392 runs — **0 failed**

Last 30 days to Nov 11 · 0 timed out

Sample: ~3% of sampled runs hit an agent timeout; Cadence retried and they finished.

Authorization never arrived

7 log requests denied

No human authorized access

Activity returned normally

Workflow completed without a diagnosis

You lost a customer at checkout.

HTTP 500

POST /checkout

AES on-call and our demo

AES on-call run

One Activity

model **tool** **model** **tool**

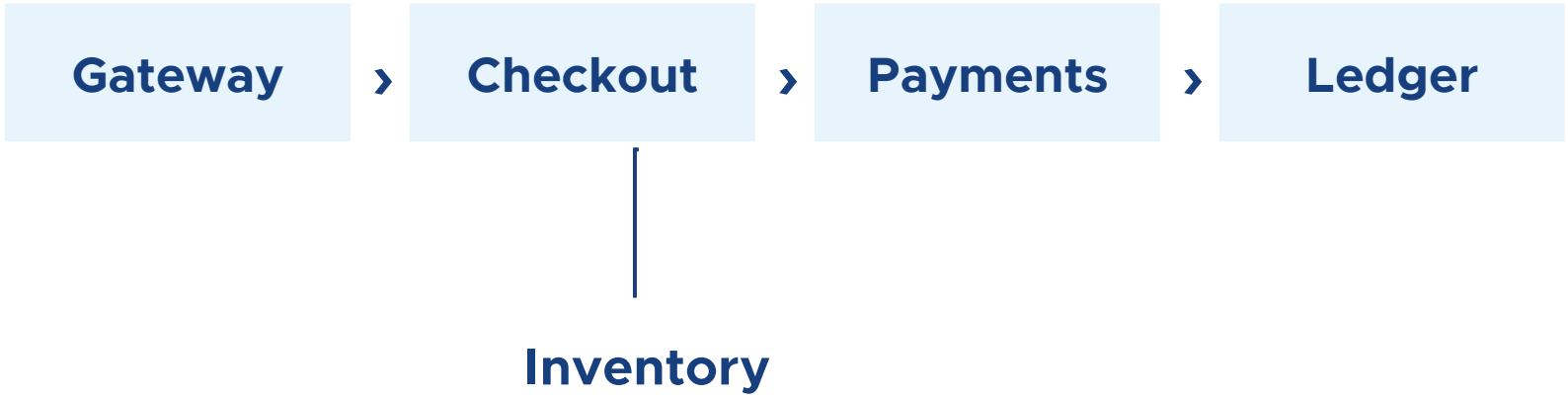
Local demo

Separate Activities

model | **tool** | **model** | **Review wait**

Separate Run. Synthetic logs. Injected 429. Workers stopped.

The checkout request crosses services



Only the failed lookup retries

Graph saved

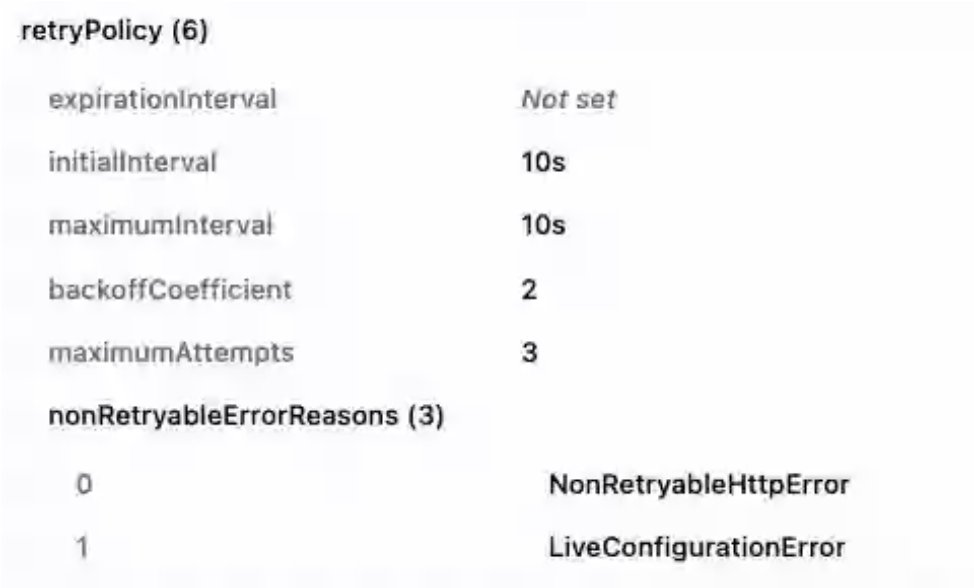
Gateway logs

429

Gateway lookup has a retry policy

DEMO 1/5 | API retry | Step 1/3

At most 3 attempts, 10 seconds apart.



Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- **Gateway logs**

Demo 1 | Source recording at 25.58s | Event 25

The API rejects the lookup

DEMO 1/5 | API retry | Step 2/3

HTTP 429: too many requests. Cadence retries the same Activity.



Lookup failed
Cadence retries

maximumAttempts	3
lastWorkerIdentity	activities-A
startedWorkerIdentity	activities-A
state	Started

lastFailureDetails

```
_internal/activity/_context.py", line 46, in execute return await self._activity_task ^^
^^^^^^^^^^^^^^^^^^^^^^^^^^ File "/Users/zawadzki/Documents/Minion/contribFest/cursor/codex-
preflight-2026-10-30/timing-tree/recipes/incident-triage/python/bare/.venv/lib/python3.1
2/site-packages/cadence/_internal/activity/_context.py", line 60, in _run_activity retur
n await self._activity_def.impl_fn(*params) ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ Fi
le "/Users/zawadzki/Documents/KubeCon Materials/Finalization/2026-11-01-evidence-and-wor
kflow-theme/build/cohesive-demo/source/activities.py", line 146, in get_logs raise http_
error(status) activities.RateLimited: HTTP 429 Too Many Requests; Retry-After: 1 "
```

lastFailureReason	RateLimited

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- **Gateway logs**

Clip 1 at 7.00s | Event 32

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The retry returns the gateway logs

DEMO 1/5 | API retry | Step 3/3

The lookup succeeds. Its logs contain an upstream HTTP 503.



Lookup completed

Result saved

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs


```
{
  service: "api-gateway",
  lines: [
    {
      at: "10:05:02.411",
      level: "ERROR",
      trace_id: "4bf92f3577b34da6",
      msg: "upstream returned 503",
      endpoint_group: "checkout",
      endpoint_id: "post-v1-checkout",
      status: "500",
      upstream: "checkout"
    }
  ]
}
```


Clip 1 at 21.00s | Event 33

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

This lookup retries temporary failures

Retry

429
Too many requests

503
Service unavailable

Connection reset
Connection dropped

Fix access or input

401
Authentication required

403
Access forbidden

404
Resource not found

The same lookup needs no second dispatch

inventory

Already recorded

Copied proposal blocked

The agent chooses checkout

DEMO 2/5 | Duplicate-call guard | Step 1/7

get_logs_by_trace_id: checkout

```
{
  tool: "get_logs_by_trace_id",
  args: {
    service: "checkout"
  },
  hypothesis: "Identify the first error in the checkout service logs"
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- • **Checkout logs**

Demo 2 | Source recording at 84.36s | Event 43

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Checkout records an inventory warning

DEMO 2/5 | Duplicate-call guard | Step 2/7

A slow stock check appears at WARN level.



Warning in logs

Lookup completed

```
{
  at: "10:05:01.120",
  level: "WARN",
  trace_id: "4bf92f3577b34da6",
  msg: "inventory stock check slow",
  downstream: "inventory",
  latency_ms: "320"
},
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs

Demo 2 | Source recording at 96.89s | Event 53

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Checkout reports a payment failure

DEMO 2/5 | Duplicate-call guard | Step 3/7

An ERROR names payments. HTTP 503 means service unavailable.



Failure in service logs

Lookup completed

```
  {
    at: "10:05:02.402",
    level: "ERROR",
    trace_id: "4bf92f3577b34da6",
    msg: "payments authorize failed",
    downstream: "payments",
    status: "503",
    wait_ms: "2000"
  }
]
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs

Demo 2 | Source recording at 96.89s | Event 53

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The agent follows inventory

DEMO 2/5 | Duplicate-call guard | Step 4/7

get_logs_by_trace_id: inventory

```
{
  tool: "get_logs_by_trace_id",
  args: {
    service: "inventory"
  },
  hypothesis: "Investigate if the slow stock check in inventory is related to the paym
  ents authorize failure."
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- • **Inventory logs**

Demo 2 | Source recording at 109.33s | Event 63

Inventory shows a warning

DEMO 2/5 | Duplicate-call guard | Step 5/7

A slow stock check is a warning, not proof of the payment failure.



Warning in logs

Lookup completed

```
{
  service: "inventory",
  lines: [
    {
      at: "10:05:01.118",
      level: "WARN",
      trace_id: "4bf92f3577b34da6",
      msg: "stock-check p99 above 300ms",
      p99_ms: "320"
    }
  ]
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs

Demo 2 | Source recording at 121.88s | Event 73

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The copied lookup stops here

DEMO 2/5 | Duplicate-call guard | Step 6/7

The application blocks an intentionally copied inventory lookup.



Duplicate blocked

No repeated call

```
phase: "RUNNING get_logs_by_trace_id{"service": "inventory"}",
last_model_action: {
  tool: "get_logs_by_trace_id",
  args: {
    service: "inventory"
  },
  hypothesis: "Investigate if the slow stock check in inventory is related t
  o the payments authorize failure."
},
injected_proposal: {
  tool: "get_logs_by_trace_id",
  args: {
    service: "inventory"
  },
  source: "deliberate fault injection after second successful tool lookup",
  model_generated: false,
  executed: false
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs

Clip 2 at 27.64s | Event 73

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The agent drafts an explanation

DEMO 2/5 | Duplicate-call guard | Step 7/7

The draft suggests inventory caused the failure. The evidence is insufficient.

result


```
{
  tool: "draft",
  args: {
  },
  hypothesis: "The checkout service initiated a request that led to a slow stock check
  in the inventory service, followed by a failed payment authorization in the payments
  service.",
  conclusion: {
    call_path: [
      "checkout",
      "inventory",
      "payments"
    ],
    root_cause: "The slow stock check in the inventory service may have caused the p
    ayment authorization to fail due to timing issues or resource contention.",
```




Draft needs review

Cause unproven

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- • **First draft**

Low confidence

Clip 2 at 35.96s | Event 87 | Draft summary paraphrased above

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The first draft

Inventory may explain payment failure

Low confidence

Missing: payments, ledger

Revision changes wording, not evidence

Quote the ERROR logs.

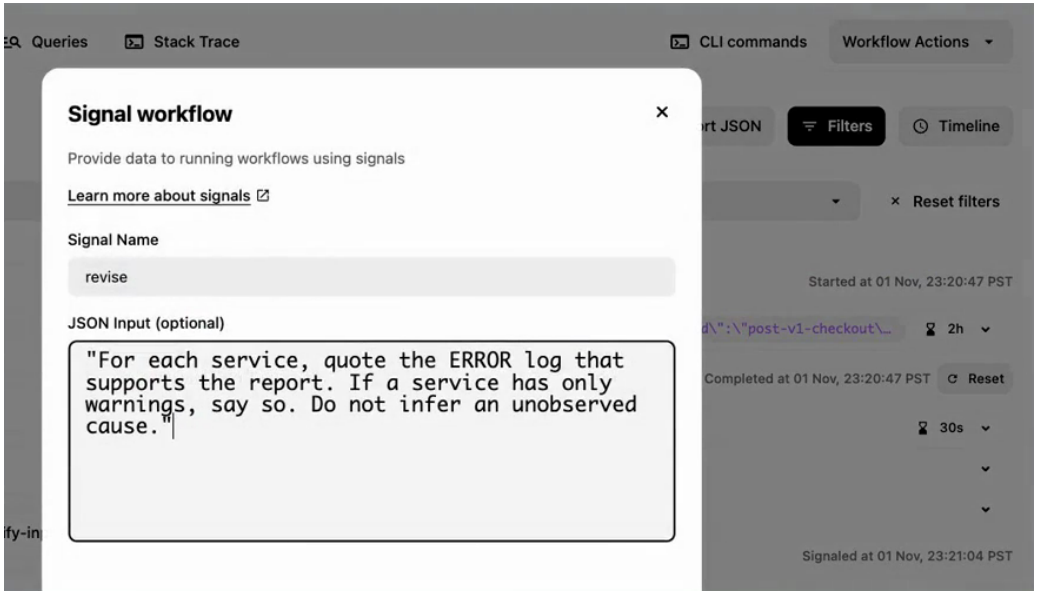
revise Signal

No new lookups

Review asks for supporting logs

DEMO 3/5 | Review existing evidence | Step 1/2

A revise Signal asks the agent to cite supporting ERROR logs.



Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- • **Revision**

Clip 3 at 1.00s | Event 95

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Confidence rises without new evidence

DEMO 3/5 | Review existing evidence | Step 2/2

The model claims high confidence without fetching new logs.



Still needs evidence

No new lookups

eventId 101

timestamp 01 Nov, 23:23:56 PST

result

```
{
  call_path: [
    "api-gateway",
    "checkout",
    "inventory",
    "payments"
  ],
  root_cause: "payments authorize failed",
  confidence: "high",
  summary: "The checkout service encountered an ERROR when attempting to authorize pay
  ments, which led to an overall failure in the transaction chain.",
  revision_request: "For each service, quote the ERROR log that supports the report. I
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision

High confidence

Self-reported, unverified.

Clip 3 at 10.76s | Event 101

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Confidence rises. Evidence does not.

High confidence

Unverified

0 new lookups

Cause still unproven

Guidance sends the agent back to work

payments › **ledger**

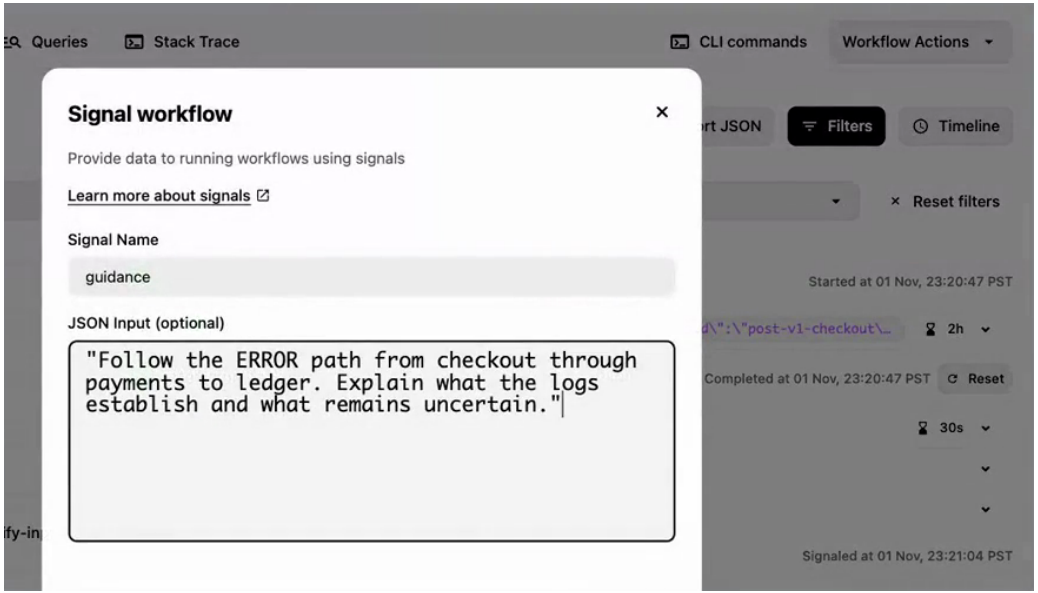
guidance Signal

Trace | Graph | **Logs** | Draft | Review | Save

Guidance requests a deeper investigation

DEMO 4/5 | Gather new evidence | Step 1/12

A guidance Signal asks the agent to investigate checkout, payments and ledger.



Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance

Clip 4 at 1.00s | Event 105

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The agent checks checkout again

DEMO 4/5 | Gather new evidence | Step 2/12

get_logs_by_trace_id: checkout

```
{
  tool: "get_logs_by_trace_id",
  args: {
    service: "checkout"
  },
  hypothesis: "Trace the ERROR path from checkout to identify the first point of failu
  re."
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- • **Checkout again**

Demo 4 | Source recording at 204.97s | Event 111

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Checkout records an inventory warning

DEMO 4/5 | Gather new evidence | Step 3/12

A slow stock check appears at WARN level.



Warning in logs

Lookup completed

```
{
  at: "10:05:01.120",
  level: "WARN",
  trace_id: "4bf92f3577b34da6",
  msg: "inventory stock check slow",
  downstream: "inventory",
  latency_ms: "320"
},
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again

Demo 4 | Source recording at 217.16s | Event 121

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Checkout reports a payment failure

DEMO 4/5 | Gather new evidence | Step 4/12

An ERROR names payments. HTTP 503 means service unavailable.

```
  {
    at: "10:05:02.402",
    level: "ERROR",
    trace_id: "4bf92f3577b34da6",
    msg: "payments authorize failed",
    downstream: "payments",
    status: "503",
    wait_ms: "2000"
  }
]
```



Failure in service logs

Lookup completed

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again

Demo 4 | Source recording at 217.16s | Event 121

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The agent requests payments logs

DEMO 4/5 | Gather new evidence | Step 5/12

get_logs_by_trace_id: payments

```
{
  tool: "get_logs_by_trace_id",
  args: {
    service: "payments"
  },
  hypothesis: "Trace ID: 4bf92f3577b34da6 - Payments authorization failed with a 503 s
  tatus."
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- • **Payments logs**

Demo 4 | Source recording at 229.37s | Event 131

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Payments reports a full ledger connection pool

DEMO 4/5 | Gather new evidence | Step 6/12

The connection pool is full: 50 of 50 in use.



Failure in service logs

Lookup completed

```
{
  at: "10:05:00.402",
  level: "ERROR",
  trace_id: "4bf92f3577b34da6",
  msg: "ledger connection pool exhausted",
  version: "v2.14",
  in_use: "50",
  max: "50",
  downstream: "ledger"
},
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs

Demo 4 | Source recording at 241.76s | Event 141

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The call to ledger times out

DEMO 4/5 | Gather new evidence | Step 7/12

Payments records a ledger timeout after 2,000 milliseconds.



Failure in service logs

Lookup completed

```
{
  at: "10:05:02.398",
  level: "ERROR",
  trace_id: "4bf92f3577b34da6",
  msg: "ledger call timed out",
  version: "v2.14",
  downstream: "ledger",
  wait_ms: "2000"
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs

Demo 4 | Source recording at 241.76s | Event 141

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The agent follows ledger

DEMO 4/5 | Gather new evidence | Step 8/12

get_logs_by_trace_id: ledger

```
{
  tool: "get_logs_by_trace_id",
  args: {
    service: "ledger"
  },
  hypothesis: "ledger connection pool exhausted and call timed out"
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- • **Ledger logs**

Demo 4 | Source recording at 254.01s | Event 151

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Ledger confirms exhausted connections

DEMO 4/5 | Gather new evidence | Step 9/12

Ledger reports its connection limit. The trigger is still unknown.



Failure in service logs

Lookup completed

```
{
  service: "ledger",
  lines: [
    {
      at: "10:05:00.401",
      level: "ERROR",
      trace_id: "4bf92f3577b34da6",
      msg: "max connections reached for client payments",
      client: "payments",
      max_connections: "50"
    }
  ]
}
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs

Demo 4 | Source recording at 266.36s | Event 161

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The new draft still needs review

DEMO 4/5 | Gather new evidence | Step 10/12

New logs arrive. The draft still makes an unsupported inventory claim.



Draft needs review

Cause unproven

```
{
  tool: "draft",
  args: {
  },
  hypothesis: "The incident originated in the checkout service, where a slow inventory
  check was observed. This led to a failure in the payments service, which could not a
  uthorize the payment due to a connection pool exhaustion and a subsequent timeout. T
  he ledger service was unable to process the request due to reaching its maximum conn
  ection limit.",
  conclusion: {
    call_path: [
      "checkout",
      "payments",
      "ledger"
    ],
    root_cause: "The root cause appears to be a combination of slow inventory checks
    and connection pool exhaustion in the payments service, leading to a timeout and
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- • **New draft**

Low confidence

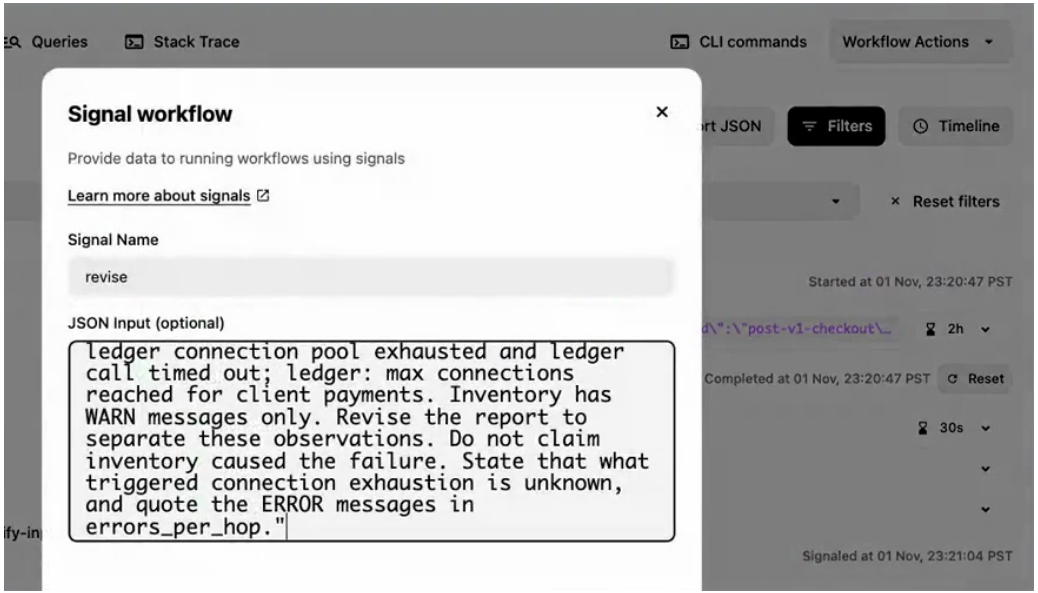
Clip 4 at 51.40s | Event 171 | Draft summary paraphrased above

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Review separates evidence from cause

DEMO 4/5 | Gather new evidence | Step 11/12

A second revise Signal asks it to separate observations from cause.



Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- • **Revision**

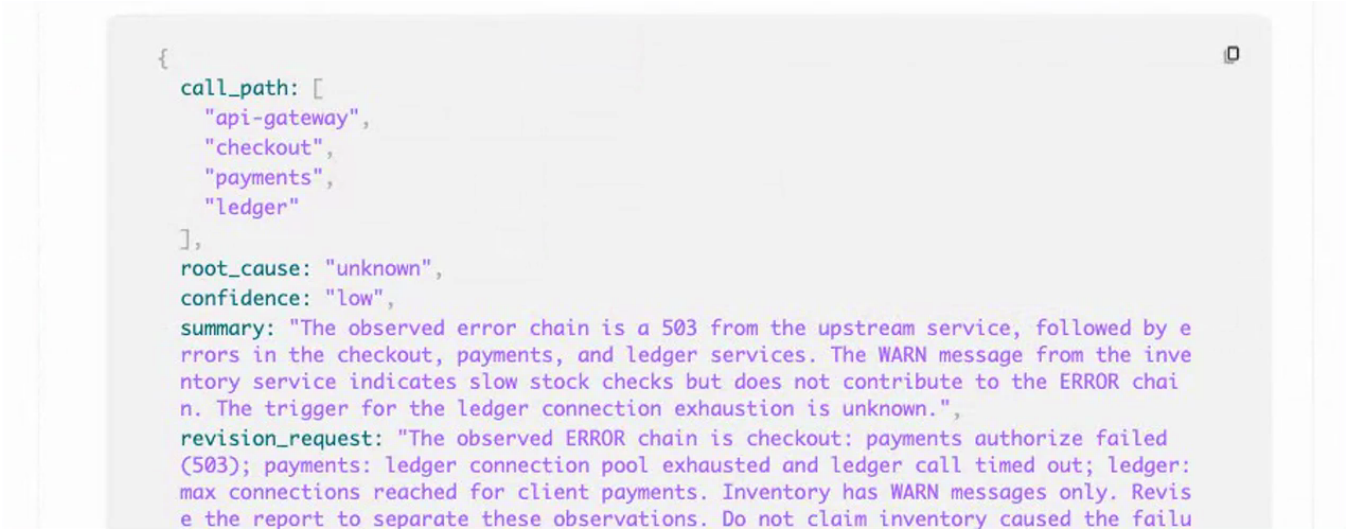
Clip 4 at 57.72s | Event 180

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The report preserves what we know

DEMO 4/5 | Gather new evidence | Step 12/12

The report records the error chain and leaves the root cause unknown.



```
{
  call_path: [
    "api-gateway",
    "checkout",
    "payments",
    "ledger"
  ],
  root_cause: "unknown",
  confidence: "low",
  summary: "The observed error chain is a 503 from the upstream service, followed by e
  rrors in the checkout, payments, and ledger services. The WARN message from the inve
  ntory service indicates slow stock checks but does not contribute to the ERROR chai
  n. The trigger for the ledger connection exhaustion is unknown.",
  revision_request: "The observed ERROR chain is checkout: payments authorize failed
  (503); payments: ledger connection pool exhausted and ledger call timed out; ledger:
  max connections reached for client payments. Inventory has WARN messages only. Revis
  e the report to separate these observations. Do not claim inventory caused the failu
```



Report revised

Ready for approval

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision

Low confidence

Clip 4 at 67.48s | Event 186

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Known failure. Unknown trigger.

checkout › payments › ledger

Connection pool exhausted

Low confidence

Trace | Graph | Logs | Draft | **Review** | Save

The Workers stop. The Workflow waits.

Workers stopped

History saved

Awaiting approval

The Workflow waits for approval

DEMO 5/5 | Worker recovery | Step 1/6

The report is ready. The Workflow waits for an approval Signal.



Waiting for approval

Workflow remains open

These are live local process commands.

```
Initial Workers
$ ps -p 7912,7913 -o pid=,stat=,ucomm=
7912 Rs   python3.12
 7913 Rs   python3.12
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision

Clip 5 at 1.00s | Worker process recording

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Both demo Workers stop

DEMO 5/5 | Worker recovery | Step 2/6

Both Workers stop. Cadence and its database keep running.



Workers stopped

Progress remains saved

The terminal confirms zero matching processes.

```
Initial Workers
$ ps -p 7912,7913 -o pid=,stat=,ucomm=
7912 Rs   python3.12
 7913 Rs   python3.12
$ kill -KILL 7912 7913
(command completed)
$ ps -p 7912,7913 -o pid=,stat=,ucomm=
(no matching processes)
Owned demo Workers alive: 0
Run remains open. Cadence and its database are
running.
Model responses before stop: 9
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision

Clip 5 at 9.96s | Worker process recording

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Replacement Workers resume the same Run

DEMO 5/5 | Worker recovery | Step 3/6

Replacement Workers resume the same Run with no new model calls.



Workers recovered

Same Run resumes

New process IDs. Same Run. Still waiting for approval.

```
$ ps -p 7912,7913 -o pid=,stat=,ucomm=
7912 Rs   python3.12
 7913 Rs   python3.12
$ kill -KILL 7912 7913
(command completed)
$ ps -p 7912,7913 -o pid=,stat=,ucomm=
(no matching processes)
Owned demo Workers alive: 0
Run remains open. Cadence and its database are
running.
Model responses before stop: 9
Replacement Workers
$ ps -p 8871,8872 -o pid=,stat=,ucomm=
8871 Rs   python3.12
 8872 Ss   python3.12
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision

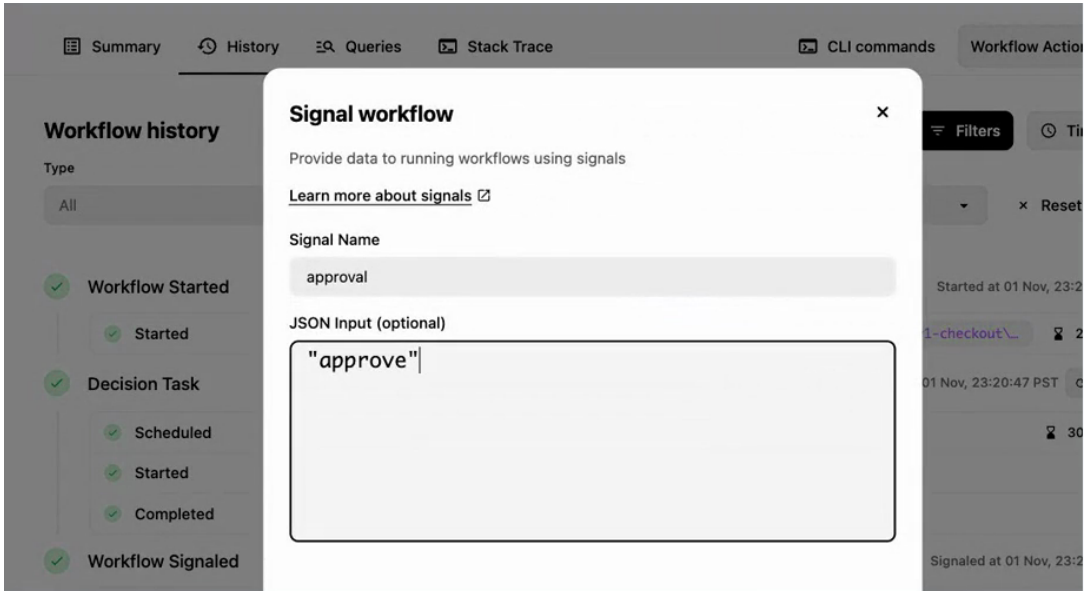
Clip 5 at 21.64s | Worker process recording

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Approval lets the Workflow continue

DEMO 5/5 | Worker recovery | Step 4/6

The recorded approval Signal releases the wait.





Approval received

Wait released

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision
- • **Approval**

Clip 5 at 31.76s | Cadence Web Signal

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The replacement Worker saves the report

DEMO 5/5 | Worker recovery | Step 5/6

The replacement Worker commits the report.

result

```
{
  receipt: "receipt-keynote-9cce44f7-1c41e5a8-ae04-4d00-8290-861ebcbbf9b1-conclusion",
  committed: true
}
```

identity	activities-B
scheduledEventId	195



Report saved

Step completed

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision
- ✓ Approval
- ✓ Report saved

Demo 5 | Source recording at 368.09s | Event 197

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

The same Workflow Run completes

DEMO 5/5 | Worker recovery | Step 6/6

The same Run completes. Recovery added zero model calls.



Workflow completed

Same Run, saved progress

Status	Completed
Workflow ID	keynote-9cce44f7
Workflow type	IncidentTriageWorkflow
Run ID	1c41e5a8-ae04-4d00-8290-861ebcbbf9b1
Start time	01 Nov, 23:20:47 PST
End time	01 Nov, 23:27:01 PST
Duration	6m 14s 479ms
History events	201
Task list	keynote-9cce44f7

Input | Result

```
{
  outcome: "conclusion_uploaded",
  conclusion: {
    call_path: [
      "api-gateway",
      "checkout",
      "payments",
      "ledger"
    ],
    root_cause: "unknown",
    confidence: "low",
    summary: "The observed error chain is a 503 from t
    he upstream service, followed by errors in the che
    ckout, payments, and ledger services. The WARN mes
    sage from the inventory service indicates slow sto
    ck checks but does not contribute to the ERROR cha
    in. The trigger for the ledger connection exhausti
    on is unknown.",
    revision_request: "The observed ERROR chain is che
    ckout: payments authorize failed (503); payments:
    ledger connection pool exhausted and ledger call t
    imed out; ledger: max connections reached for clie
```

Investigation so far

- ✓ Trace saved
- ✓ Graph saved
- ✓ Gateway logs
- ✓ Checkout logs
- ✓ Inventory logs
- ✓ First draft
- ✓ Revision
- ✓ Guidance
- ✓ Checkout again
- ✓ Payments logs
- ✓ Ledger logs
- ✓ New draft
- ✓ Revision
- ✓ Approval
- ✓ Report saved

Clip 5 at 49.76s | Event 201

Recorded local demo. Synthetic logs. Reviewer Signals automated. Screenshot cropped for readability.

Half the tokens in this recovery comparison

Qwen2.5:7b. Same requests. Python restarts without a checkpoint.

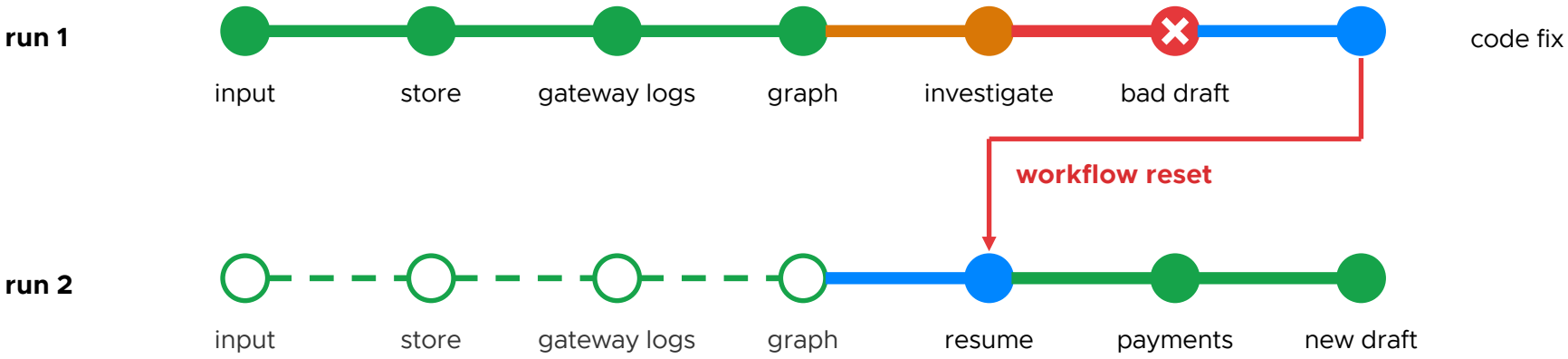
Recorded comparison	Python restart	Cadence Run
Model responses	**18**	**9**
Input tokens	**7,802**	**3,901**
Output tokens	**1,502**	**751**
Total logical tokens	**9,304**	**4,652**
New calls during recovery	**9**	**0**

50% fewer logical tokens

Recovery at approval wait. Local inference. Infrastructure cost unpriced.

Ctrl+Z for agents

ILLUSTRATION



Your Activity boundary decides what repeats

Whole agent

model tool model tool

Separate Activities

model | tool | model | Review wait

Coordinator → Child Workflows → nested children
Await results · Set timeouts · Handle failures · Validate outputs

830,720 AES runs ended in September

Michelangelo AI

On-demand runs. One PHX domain. September 2026 (UTC).



66.74%

Completed

554,400 runs

32.93%

Timed out

273,547 runs

0.13%

Failed

1,080 runs

Terminated: 1,693 (0.20%). Canceled: 0.

2,932 sampled histories underpin the next slides.

Completion describes execution, not diagnosis quality.

Failures inside completed runs

~32,000
completed AES runs had an earlier recorded failure.

Same Run ID.
Earlier Activity results retained.

Cadence retried Activities or redispatched Workflow tasks.

Sample estimate: 5.8% of completed runs. 95% range: 2% to 11%.

Most retried Activities completed

9 in 10 retried Activities completed

The Activity boundary decides how much work repeats. Earlier completed results stay recorded.

September 2026 weighted estimate. Excludes Activities still running at Workflow timeout.

A timeout tells us where to look

Share of recorded failure events, not runs

50%

Workflow task

No response before timeout

Worker response

32%

Activity pickup

Did not start before timeout

Queue and pollers

17%

Activity heartbeat

Heartbeat stopped arriving

Activity and dependencies

Weighted from 2,932 histories. Harness-handled errors excluded. Other events: about 1%.

Cadence narrows where to investigate

>99%

of recorded timeouts were in one Worker pool

Filter by Task List.
Inspect those histories.

AES on-demand runs closed in September 2026. Timeouts grouped by Task List.

Retried Activities that completed

Activity timeout	Completed	Retried	Completion
Pickup	136	138	99%
Heartbeat	93	135	69%
Execution	6	6	100%

Pooled sample counts. The overall 91.6% estimate uses population weighting.

Excludes retries still running when the Workflow timed out.

Your agent with Cadence

Your agent



With Cadence



Your agent logic stays yours.

Your functions

Wrap model and tool calls as Activities.
Coordinate them in a Workflow.

Retries and approval

Set bounded retry policies.
Use Signals and timers for review.

Workflow history

Inspect failures and scheduling waits.
Reset at a supported history boundary.

Start with one model or tool call. Test recovery at its boundary.

KubeCon CloudNativeCon North America 2026

Cadence Workflow ContribFest: Build Durable AI Agent Examples

Salt Lake City, Utah, United States





Diana Zawadzki Shijie Sheng

Build durable agents with us

CONTRIBFEST

Retries and recovery

Handle a temporary tool failure.
Verify the recovery in history.

Human review

Send guidance or approval.
Bound the wait with a timer.

Research workflows

Validate research plans and sources.
Coordinate parallel researchers.

MCP tools

Give tool calls recovery points.
Check their outputs.

Facilitators: Diana Zawadzki, Shijie Sheng, Kevin Burns

github.com/cadence-workflow/cadence-ai-samples

Try Incident Triage in the Samples Explorer

```
cd cadence-ai-samples
./scripts/explore.sh
```

Choose Incident Triage, then mock or live mode. Follow the setup guide.

A failed lookup

Retry a temporary error.
Stop on a permanent 404.

A draft that needs more evidence

Compare it with the logs.
Send guidance to investigate further.

github.com/cadence-workflow/cadence-ai-samples

Integrating your agent framework

Google ADK

```
registry.register_activities(
    GoogleADKActivities()
)
```

Model calls use Activities.
Wrap your tools too.

OpenAI Agents SDK

```
registry.register_activities(
    OpenAIActivities()
)
```

Keep Runner.run.
Register Activities.

Workflow guidance: cadence-workflow/ai-skills

Smarter agents still need reliable systems

Failure is normal

Keep the evidence you gathered.

Keep a way to question the result.

Build durable AI examples

github.com/cadence-workflow/cadence-ai-samples

KubeCon CloudNativeCon
North America 2026

Agents fail. Workflows recover.

Thank you. Questions?

Diana Zawadzki
Cadence team, Uber

Shijie Sheng
Cadence team, Uber

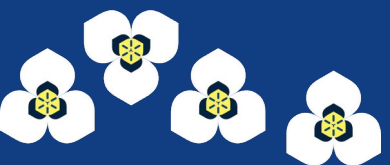
Session feedback

AI samples

Run an example

Python SDK

SDK & integrations





Primary references

Cadence service and worker topology

Workflows and Activities

Signals, timers and approval waits

Cadence CLI

Python SDK and integrations

Replay and shadower testing