



KubeCon



CloudNativeCon

Europe 2026

#KubeCon #CloudNativeCon

Cadence Project Pavilion

Ender Demirkaya



Outline

- Intro
- Adopters
- Demos
 - Agentic Workflow Orchestration (OpenAI Demo)
 - Custom Workflow Controls (Cadence Hosted UI, DevOps, ...)
 - E2E Data Encryption
 - Concurrency Controls
- 2025
- Roadmap

Intro

What is Cadence?

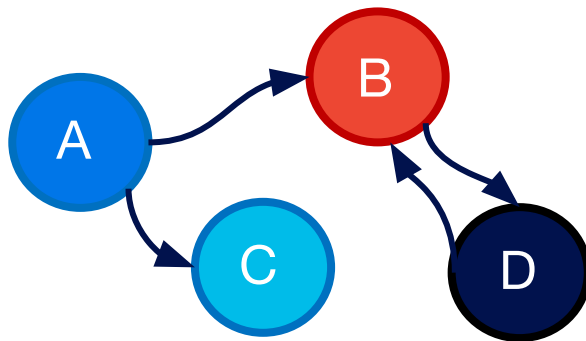
Cadence is a
code-driven
workflow
orchestration
engine

Workflow?

Orchestrated and repeatable pattern of activities.

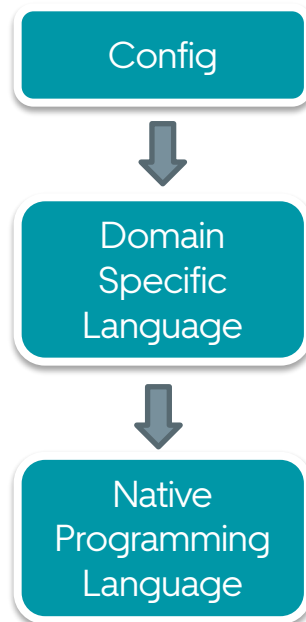
Evolution of Workflows

- Data Pipelines (DAG)
- State Machines
- Schedules / Cron Operations
- Microservice Orchestration
- Batch Processing
- Event Driven Processing
- APIs / External Invocations
- Signals/Clicks (Human-in-the-loop)
- Triggers
- Timers
- Async Processing
- Even single lambda functions
- ...



TLDR;
Any program
is a workflow

Development



Orchestration?

- State Machine Transitions
- Load distribution / balance (among workers)
- Retries (vs crashes, restarts, failures...)
- Execution Resets / Replays
- Durability (saving state up to years)
- Multi-actor operations (admin, dev, ops)
- Data archival (retention management)
- Logic Versioning
- Disaster Recovery (region failovers)
- High Availability / Low Latency
- Scalable (billions of executions)
- Adaptive
- Built-in UI with available customizations
- Observability (metrics, logs ...)

TLDR

Common overheads
while building
Distributed services

Code-driven? Why?

Build any business logic, and only worry about the business logic
(don't think about its overheads: scale, reliability, ...)

No need to learn a new config / DSL format
(just use the programming language you know the best)

Not just background / low pri workloads; but also high pri production scenarios

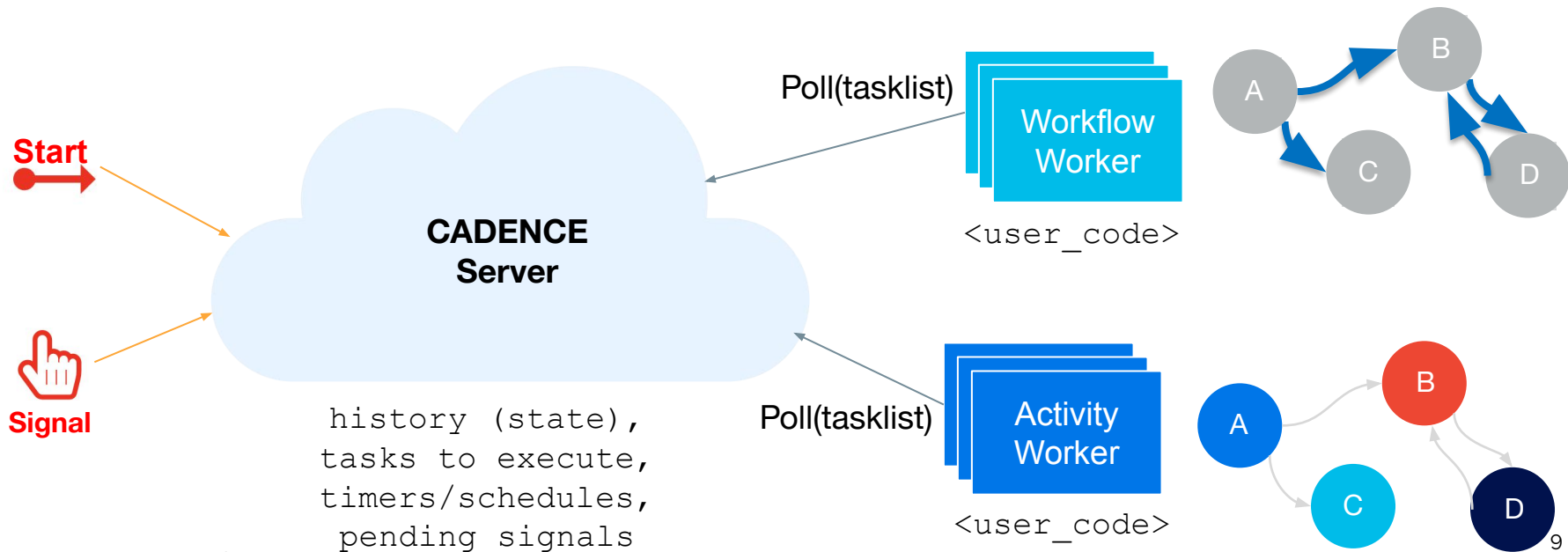
Your service will be ***orchestrated*** with

- a simple SDK import and
- function annotations

Code-driven: Concepts

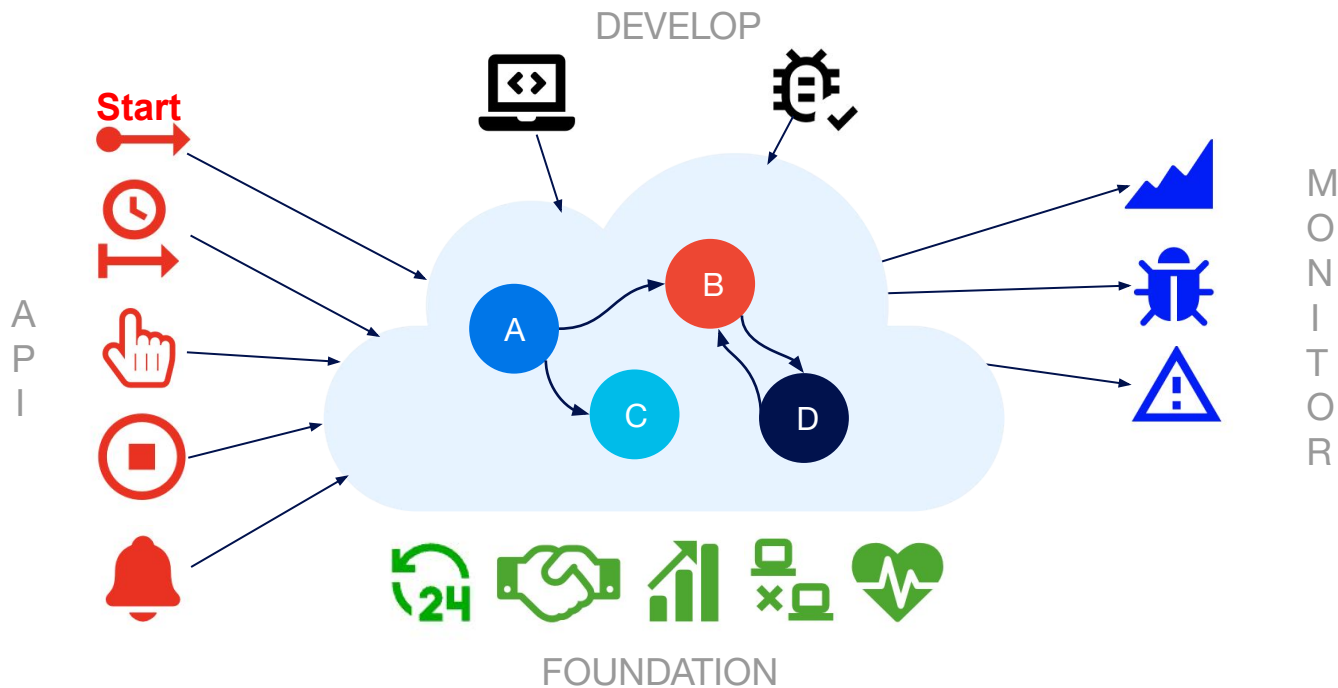


Orchestration is managed by a Cadence server while your service only contains your code



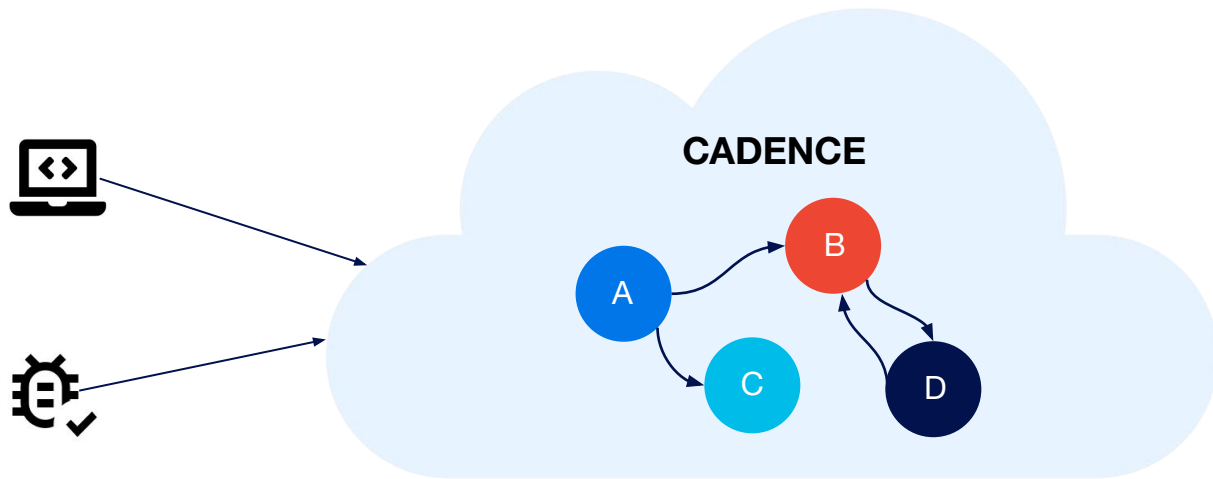
Wrap-up

What's a Well-Supported Distributed Service?



... with Cadence

Develop tools & services without having to worry about its overheads



Some Concrete Use Cases

Multi-bank transactions; Fintech

Records of exactly what happened,
where the system failed if it did

Deployment, Provisioning, Infra

Automatic retries, policies

Data Pipelines, Model Training

Crons/Schedules, Large Data Handling

Agentic Workflows

Context retention, Resume, Durability

Batch Work, Bulk / Monthly Ops

Concurrency controls, Queuing

Human-in-the-Loop

Product services, web / mobile apps

Efficient Lambda-like Functions

Allows overprovisioning, elastic, adaptive
highly multitenant, no cold start issues

Internal Tooling, DevOps, Migrations

Avoids downsides of scripting, efficient
fully built-in customized controls

Do I need it?

Almost* certainly yes!

*** The Trade-off:**

- Every execution/step is recorded:
 - storage
 - several ms latency
- Async (queued) execution

For example; not ideal for building video games

Why Cadence?

Most powerful orchestration engine

No vendor-lock, fully open source, with open governance

No private logic to sell a managed solution

90%+ maintainers are “users”, not vendors

but we support and welcome any vendor who would like to sell

Experience oriented (vs profit oriented)

Deployed with Kubernetes, not tied to any cloud provider

4K+

Domains

~500B

Actions a Month

450K

Actions a Second

40%

Code deleted after
switching

~150

Companies in Slack®

>99.9

Availability

Adopters?

Public Adopters (Adopters.md):

The Uber logo, consisting of the word "Uber" in a black, sans-serif font.The NetApp logo, featuring a purple square icon with a white 'N' shape inside, followed by the word "NetApp" in a black, sans-serif font.The Doordash logo, featuring a red stylized 'D' icon above the word "DOORDASH" in a red, bold, sans-serif font.

Cadence is a 9+ year old project. Its ecosystem has ~150 companies

We became a CNCF Sandbox project last year; we weren't collecting logos before.

Publicly announcing adopters is a challenge (legal).

Use it and share your logos to get more support at Incubating/Graduated levels!

Demos

Visit

github.com/cadence-workflow/cadence-samples

for more

Deploying with Helm Charts

[Watch on YouTube!!](#)

Codelab **Deploy Cadence with Helm on GKE**



Kevin Burns

Developer Advocate, Uber

Deploying with Helm Charts

```
brew install --cask google-cloud-sdk
```

```
brew install kubectl
```

```
brew install helm
```

```
gcloud init
```

Or configure manually:

```
gcloud auth login
```

```
gcloud config set project <YOUR_GCP_PROJECT_ID>
```

```
gcloud config set compute/region us-central1
```

```
gcloud config set compute/zone us-central1-a
```

Get Cadence Helm Charts

This codelab uses the local charts approach. Clone the repository:

```
git clone https://github.com/cadence-workflow/cadence-charts.git
```

```
cd cadence-charts
```

Alternative approach: Use the remote Helm repository (not used in this codelab):

```
helm repo add cadence https://cadence-workflow.github.io/cadence-charts
```

```
helm repo update
```

Deploying with Helm Charts

```
kubectl create namespace cadence-postgres-os2
```

```
cd cadence-charts
```

```
helm repo add bitnami https://charts.bitnami.com/bitnami
```

```
helm repo add opensearch https://opensearch-project.github.io/helm-charts/
```

```
helm repo update
```

```
helm dependency build ./charts/cadence
```

```
helm upgrade --install cadence-release ./charts/cadence \
  --namespace cadence-postgres-os2 \
  -f charts/cadence/examples/values.postgres-os2.yaml \
  --wait
```

You should see pods for:

- cadence-release-frontend, cadence-release-history, cadence-release-matching, cadence-release-worker
- cadence-release-web
- cadence-release-postgresql
- cadence-release-kafka-controller, cadence-release-kafka-broker
- cadence-release-opensearch-master-0
- Schema jobs (may show as Completed)

Deploying with Helm Charts

```
kubectl port-forward -n cadence-postgres-os2 svc/cadence-release-frontend 7833:7833
```

```
kubectl port-forward -n cadence-postgres-os2 svc/cadence-release-web 8088:8088
```

```
POD=$(kubectl get pods -n cadence-postgres-os2 \
  -l app.kubernetes.io/component=frontend \
  -o jsonpath='{.items[0].metadata.name}')
```

```
kubectl exec -n cadence-postgres-os2 -it "$POD" -- \
  cadence --address cadence-release-frontend:7833 --transport grpc \
  --do cadence-samples domain register -rd 1
```

Agentic Workflows (OpenAI example)

Workflow: BookFlightAgentWorkflow

Status	Completed
Workflow ID	620d948f-92af-41a1-aba1-3a35fe18b69c
Workflow type	BookFlightAgentWorkflow
Run ID	e44dd68c-edb5-4b83-94eb-8c9b83eea5ec
Start time	22 Mar, 20:57:21 GMT+1
End time	22 Mar, 20:57:30 GMT+1
Duration	8s 926ms
History events	23
Task list	<u>agent-task-list</u>

Input

```
[  
  "Book a flight from New York to London on March 20th 2026  
  at 10:00 AM and return on March 25th, 2026 at 10:00 AM"  
]
```

Result

```
"Your flight has been successfully booked! Here are the details: - **From:** New York - **To:** London - **Departure Date:** March 20, 2026, at 10:00 AM - **Return Date:** March 25, 2026, at 10:00 AM - **Airline:** United - **Flight Number:** 123456 - **Seat Number:** 12A - **Price:** $100.00 If you need any more assistance, feel free to ask!"
```

Agentic Workflows (OpenAI example)

Workflow history

[Export JSON](#)**Grouped**[Ungrouped](#)[Filters](#)[Timeline](#)

Event group	Status	Time	Duration	Details
 Workflow Started	 Started	22 Mar, 20:57:21 GMT+1		<code>["Book a flight from New York to London on March...</code>  10m
 Decision Task	 Completed	22 Mar, 20:57:21 GMT+1	53ms	 Reset
 Activity 0: OpenAIActivities.invo ke_model	 Completed	22 Mar, 20:57:27 GMT+1	5s 501ms	<code>["gpt-4o-mini",null,["content":"Book a flight f...</code>  10m <code>{"output":["arguments":{"departure_date":"2...</code>
 Decision Task	 Completed	22 Mar, 20:57:27 GMT+1	57ms	 Reset
 Activity 1: book_flight	 Completed	22 Mar, 20:57:27 GMT+1	8ms	<code>["New York","London","2026-03-20T10:00:00+00:00"...</code>  10m <code>{"from_city":"New York","to_city":"London","depa...</code>
 Decision Task	 Completed	22 Mar, 20:57:27 GMT+1	24ms	 Reset
 Activity 2: OpenAIActivities.invo ke_model	 Completed	22 Mar, 20:57:30 GMT+1	3s 212ms	<code>["gpt-4o-mini",null,["content":"Book a flight f...</code>  10m <code>{"output":{"id":"msg_01da76e617ca26cc0169c049a8...</code>
 Decision Task	 Completed	22 Mar, 20:57:30 GMT+1	71ms	 Reset
 Workflow Completed	 Completed	22 Mar, 20:57:30 GMT+1		<code>"Your flight has been successfully booked! Here ...</code>

Agentic Workflows (Definition)

Agent
Definition

```
@cadence_registry.activity(name="book_flight")
async def book_flight(from_city: str, to_city: str, departure_date: datetime, return_date: datetime) -> Flight:
    """
    Book a flight tool
    """
    return Flight(from_city=from_city,
                  to_city=to_city,
                  departure_date=departure_date,
                  return_date=return_date,
                  price=100,
                  airline="United",
                  flight_number="123456",
                  seat_number="12A")
```

Workflow
Definition

```
@cadence_registry.workflow(name="BookFlightAgentWorkflow")
class BookFlightAgentWorkflow:
    @cadence.workflow.run
    async def run(self, input: str) -> str:
        agent = Agent(
            name = "Book Flight Agent",
            model = "gpt-4o-mini",
            tools = [
                function_tool(book_flight),
            ],
        )
        result = await Runner.run(agent, input, run_config=RunConfig(
            tracing_disabled=True,
        ))
        return result.final_output
```

Agentic Workflows (Call)

Workflow
call with
a prompt

Waiting
for the
response

```
async def main():
    set_default_agent_runner(CadenceAgentRunner())
    worker = cadence.worker.Worker(...)
)

async with worker.run() as worker:
    execution = await worker.client.start_workflow(
        "BookFlightAgentWorkflow",
        "Book a flight from New York to London on March 20th 2026 at 10:00 AM and return on March 25th, 2026 at 10:00 AM",
        task_list=worker.task_list,
        execution_start_to_close_timeout=timedelta(minutes=10),
    )

    print(f"cadence workflow started: http://localhost:8088/domains/default/cluster0/workflows/{execution.workflow\_id}/{execution.run\_id}/summary")

    # get workflow result
    response: GetWorkflowExecutionHistoryResponse = await worker.client.workflow_stub.GetWorkflowExecutionHistory(
        GetWorkflowExecutionHistoryRequest(
            domain=worker.client.domain,
            workflow_execution=execution,
            wait_for_new_event=True,
            history_event_filter_type=EventFilterType.EVENT_FILTER_TYPE_CLOSE_EVENT,
            skip_archival=True,
        )
    )

    return response.history.events[-1].workflow_execution_completed_event_attributes.result.data.decode()
```

Custom Workflow Controls

Demo Custom Workflow Controls



Kevin Burns
Developer Advocate, Cadence



Kevin S. Burns

[Watch it on YouTube!](#)

Custom Workflow Controls (Definitions)

Example: Basic Start

```
{% start
  workflowType="MyWorkflow"
  label="Start Workflow"
  domain="my-domain"
  cluster="my-cluster"
  taskList="my-task-list"
/%}
```

Start Workflow

Example: Start with Options

```
{% start
  workflowType="MyWorkflow"
  label="Start with Config"
  domain="my-domain"
  cluster="my-cluster"
  taskList="my-task-list"
  wfId="custom-workflow-id"
  input={key: "value"}
  timeoutSeconds=300
  sdkLanguage="GO"
/%}
```

Start with Config

Signal Workflow

Send a signal to a running workflow using the `signal` tag.

Example:

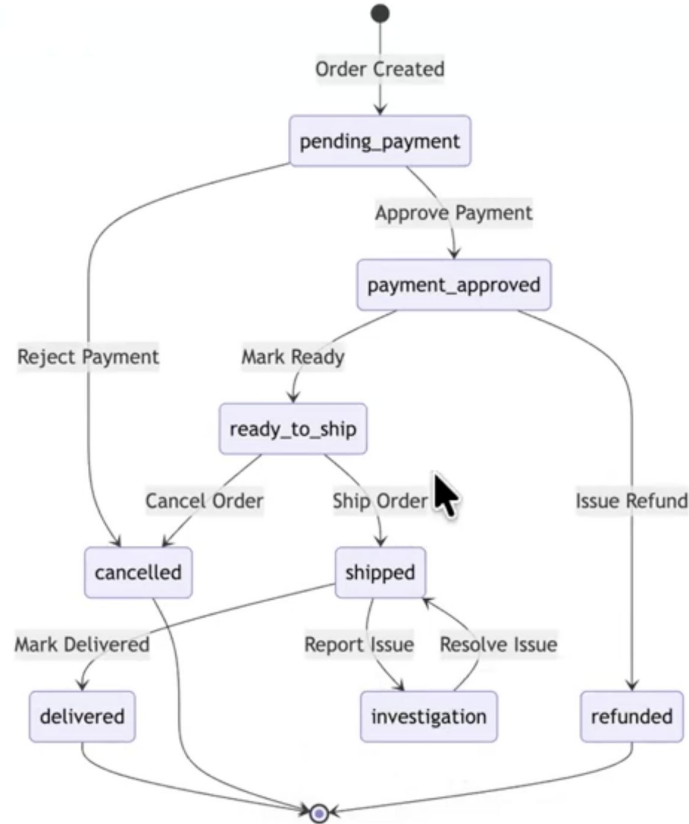
```
{% signal
  signalName="approve"
  label="Approve"
  input={status: "approved", user: "john"}
  domain="my-domain"
  cluster="my-cluster"
  workflowId="workflow-123"
  runId="run-456"
/%}
```

Note that input field is optional and can be omitted. For different value types, you can use the following syntax:

- For booleans, use `input=true` or `input=false`
- For json objects, use `input={"key": "value"}`
- For strings, use `input="string"`

Signal Workflow

Custom Workflow Controls (Order Flow Example)



Custom Workflow Controls (Order Flow Example)

Cadence

cadence-samples > d02ec22d-a56b-46f5-9915-7de66be6ede5 > 2be878b6-3611-4710-b4d6-174bd95c09bb Running

Summary History Queries Stack Trace

CLI commands Workflow Actions

__open_sessions

Add input

Run

dashboard



Add input

Run



Order Dashboard

Your admin panel - manage orders directly from Cadence Web.



Available Actions

Payment Review:

✓ Approve Payment

✗ Reject: Policy Violation

✗ Reject: Fraud Suspected

✗ Reject: Customer Request



Order Details

Field

Value

Custom Workflow Controls (Order Flow Example)

 **Order Details**

Field	Value
Order ID	ORD-2024-001234
Customer	Alice Johnson
Email	alice.johnson@example.com
Created	2026-03-22 22:20:13
Status	● Pending Payment

Refund: \$0.00

 **Order Items**

Item	Qty	Price	Subtotal
Wireless Headphones	2	\$79.99	\$159.98
Phone Case	1	\$19.99	\$19.99

Total: \$179.97

 **Action History**

Timestamp	Action	Operator	Details
22:20:13	Order Created	System	Order ORD-2024-001234 created for Alice Johnson

 **Available Actions**

Payment Review:

✓ Approve Payment ✗ Reject: Policy Violation ✗ Reject: Fraud Suspected

✗ Reject: Customer Request

 **Order Details**

Field	Value
Order ID	ORD-2024-001234
Customer	Alice Johnson
Email	alice.johnson@example.com
Created	2026-03-22 22:20:19
Status	● Pending Payment

Refund: \$0.00

Custom Workflow Controls (Order Flow Example)

 **Order Dashboard**

Your admin panel - manage orders directly from Cadence Web.

 **Available Actions**

Fulfillment Actions:

 **Mark Ready to Ship**

Refund Options:

 **Full Refund (\$179.97)**

 **Partial Refund (50%)**

 **Order Details**

Field	Value
Order ID	ORD-2024-001234

 **Order Dashboard**

Your admin panel - manage orders directly from Cadence Web.

 **Available Actions**

Delivery Confirmation:

 **Mark as Delivered**

Custom Workflow Controls (Order Flow Example)

Action History

Timestamp	Action	Operator	Details
22:20:19	Order Created	System	Order ORD-2024-001234 created for Alice Johnson
22:21:45	Payment Approved	ops-user	Payment of \$179.97 approved
22:21:54	Marked Ready to Ship	ops-user	Order prepared and ready for shipping
22:22:01	Order Shipped	ops-user	Carrier: UPS, Tracking: 1Z999AA10123456784
22:22:07	Order Delivered	ops-user	Package confirmed delivered to customer

E2E Data Converters (Encryption, Compression, ...)

```
// NewCompressedJSONDataConverter creates a new compressed JSON data converter.
func NewCompressedJSONDataConverter() encoded.DataConverter {
    return &compressedJSONDataConverter{}
}

func (dc *compressedJSONDataConverter) ToData(value ...interface{}) ([]byte, error) {
    var jsonBuf bytes.Buffer
    enc := json.NewEncoder(&jsonBuf)
    for i, obj := range value {
        err := enc.Encode(obj)
        if err != nil {
            return nil, fmt.Errorf("unable to encode argument: %d, %v, with error: %v",
                i, obj, err)
        }
    }

    var compressedBuf bytes.Buffer
    gzipWriter := gzip.NewWriter(&compressedBuf)

    _, err := gzipWriter.Write(jsonBuf.Bytes())
    if err != nil {
        return nil, fmt.Errorf("unable to compress data: %v", err)
    }

    err = gzipWriter.Close()
    if err != nil {
        return nil, fmt.Errorf("unable to close gzip writer: %v", err)
    }

    return compressedBuf.Bytes(), nil
}
```

```
// Create compressed JSON data converter
dataConverter := NewCompressedJSONDataConverter()

// Show compression statistics on startup
printCompressionStats()

workerOptions := worker.Options{
    Logger:      logger,
    MetricsScope: tally.NewTestScope(TaskListName, nil),
    DataConverter: dataConverter, // Custom DataConverter for compression
}

w := worker.New(
    cadenceClient,
    Domain,
    TaskListName,
    workerOptions)
```

Concurrency Controls

```
// BatchWorkflowInput configures the batch processing parameters.
type BatchWorkflowInput struct {
    Concurrency int // Maximum number of activities running in parallel
    TotalSize   int // Total number of activities to process
}

// BatchWorkflow demonstrates processing large batches of activities with controlled
// concurrency using workflow.NewBatchFuture. This pattern is useful for:
// - Processing thousands of records without overwhelming downstream services
// - Respecting the 1024 pending activities limit per workflow
// - Automatic error handling and retry management
func BatchWorkflow(ctx workflow.Context, input BatchWorkflowInput) error {
    // Create activity factories for each task (not yet executed)
    factories := make([]func(workflow.Context) workflow.Future, input.TotalSize)
    for taskID := 0; taskID < input.TotalSize; taskID++ {
        taskID := taskID // Capture loop variable for closure
        factories[taskID] = func(ctx workflow.Context) workflow.Future {
            // Configure activity timeouts
            aCtx := workflow.WithActivityOptions(ctx, workflow.ActivityOptions{
                ScheduleToStartTimeout: time.Minute * 1,
                StartToCloseTimeout:    time.Minute * 1,
            })
            return workflow.ExecuteActivity(aCtx, BatchActivity, taskID)
        }
    }

    // Execute all activities with controlled concurrency
    batch, err := workflow.NewBatchFuture(ctx, input.Concurrency, factories)
    if err != nil {
        return fmt.Errorf("failed to create batch future: %w", err)
    }

    // Wait for all activities to complete
    return batch.Get(ctx, nil)
}
```

More Demos?

Visit

github.com/cadence-workflow/cadence-samples

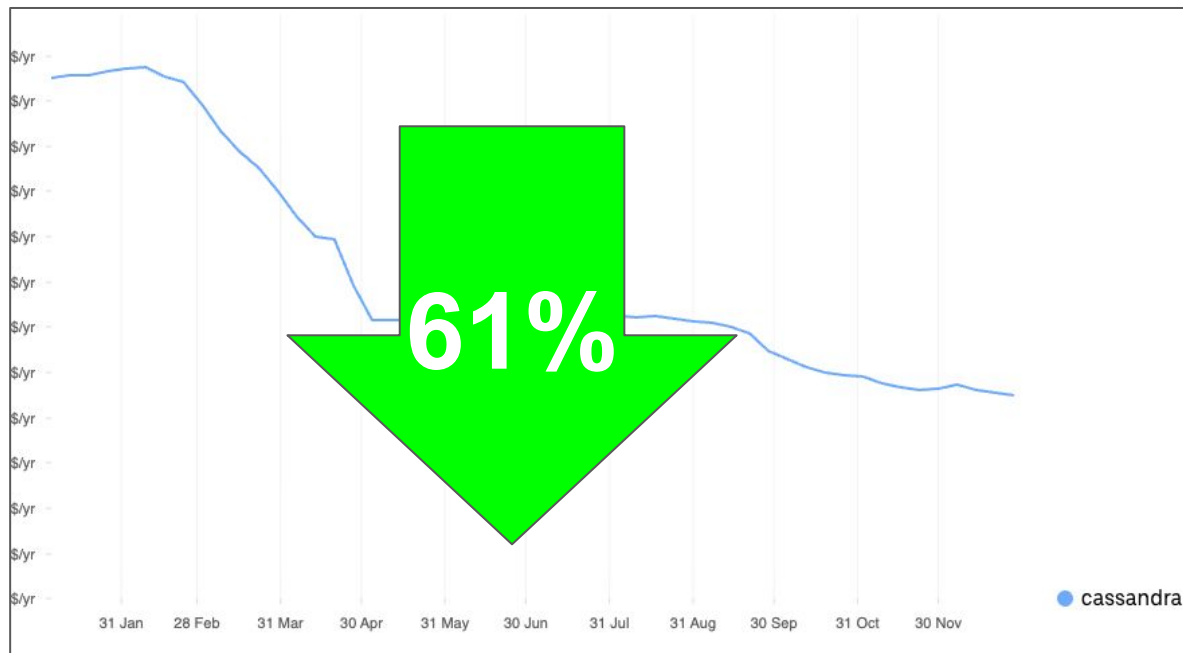
2025

Became a CNCF Sandbox Project



Cadence joined CNCF in 2025

Cadence Cassandra Cost



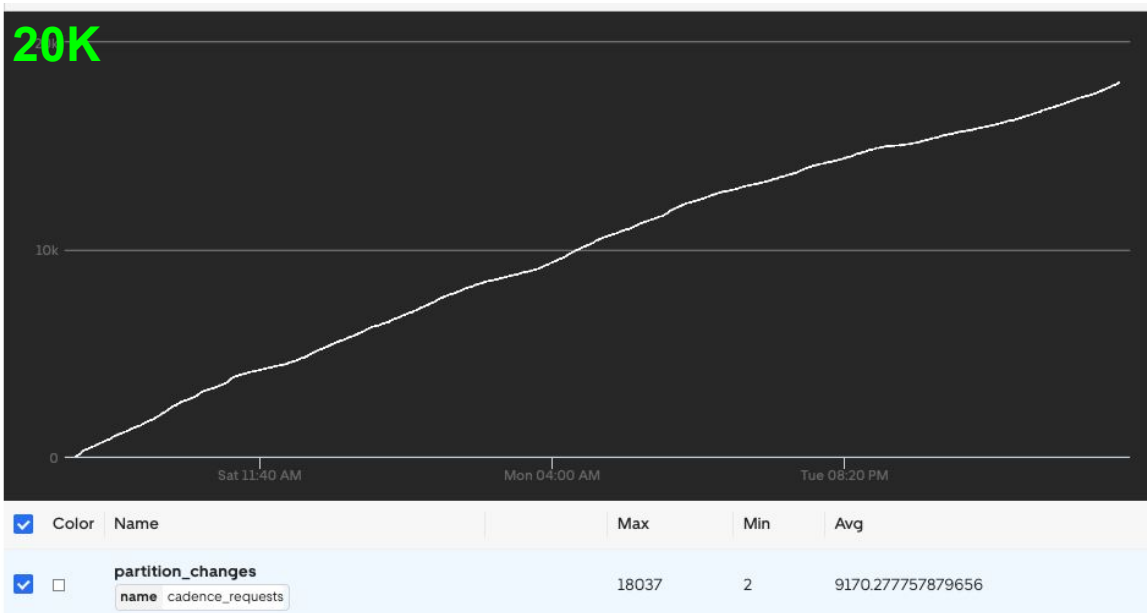
Metric showing total Cadence Cassandra cost between Jan 1st and December 31st in 2025

What Happened?

- Eliminated DB Traffic Bursts
- Adaptive tasklist partitioning
- Proactive task cleanup in standby region
- Increased in-memory matching
- Improved history queues
- Async workflows
- Replication Cache

Cadence is 80% cheaper compared to its competitors!
More to come in 2026!

Adaptive TaskList Partitioning



Metric showing total number of tasklist partition changes

Scales tasklist partitions based on user traffic

Takes action within seconds (instead of manual partitioning)

~20k upscales + downscales per month

History Queues v2



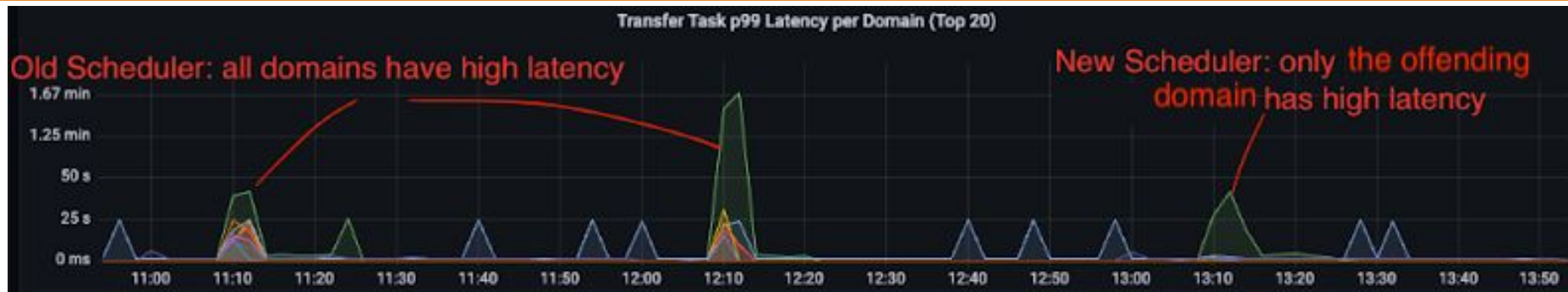
Metric showing Cassandra latency change at p99 number after the new History service queues were enabled.

Provides isolation by splitting tasks into virtual queues

Improves the task queue latency

Automatically isolates domains with an outlier traffic

Eliminating Noisy Neighbor Effect



Metric showing task processing latencies per domain. On the left side (before new schedulers are enabled), bursts impact other domains, while in the right side (after the feature was enabled) bursts are automatically isolated, without impacting other domains.

Task queueing latency w noisy neighbor presence (p99): **~60s => milliseconds**

Noisy neighbors with sudden high traffic are automatically isolated

Essential for highly multi-tenant clusters; some clusters hosts **2000+ domains**

Active-Active Domains

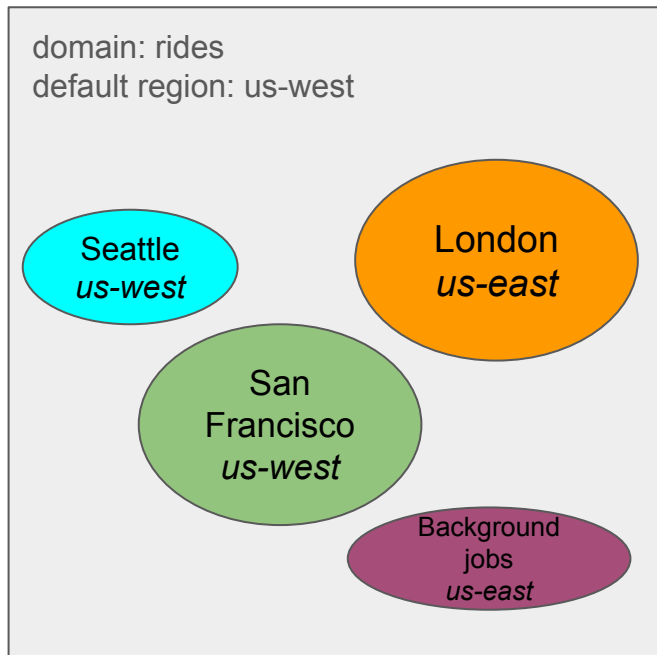


Figure showing a domain with a default active region "us-west" and some its workflows grouped by cluster attributes choosing their own active/default regions

- Running workflows from the same domain in multiple regions.
- Users define which workflow is active in which region.
- Workflows can be grouped by aliases (e.g. cities)
- Avoids having to create multiple domains for the same service
- Aligns worker and Cadence processing regions

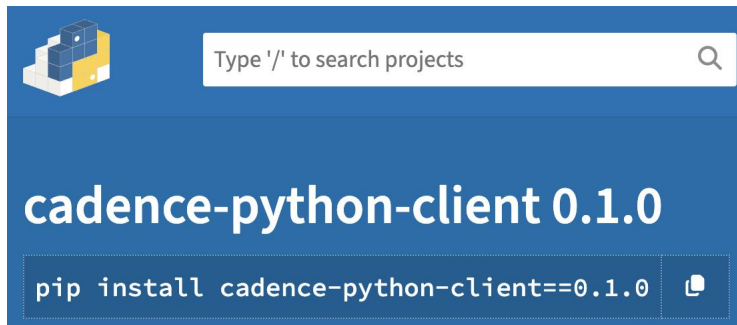
Cadence Python SDK

What's supported:

- Deterministic event loop
- Workflow engine
- Workflow/Activity registry
- Activity Executor

Parity with Golang SDK by April 2026

Groundwork for **Agent Orchestration, Data Pipelines, Model training**



Official Cadence PyPi package

```
@registry.activity()
async def echo(message: str) -> str:
    return message

@registry.workflow()
class SingleActivity:
    @workflow.run
    async def run(self, message: str) -> str:
        echo_activity = echo.with_options(
            schedule_to_close_timeout=timedelta(seconds=10)
        )
        return await echo_activity(f"Hello {message}!")
```

test-domain > 08995

Summary History

Workflow history

Type

All

- Workflow Started
- Decision Task
- Activity echo_1: echo
- Decision Task
- Workflow Completed

Workflow Started

Started at 01 Dec, 15:31:57 PST

Started

eventid 1

timestamp 01 Dec, 15:31:57 PST

workflowType.name SingleActivity

taskList test_single_activity

input

```
[
  "hello world"
]
```

Figure showing the first workflow written with the official Python SDK and its web view

Cadence Worker Auto Scaler

Scales worker pollers up/down

Makes workers CPU bound

Enables CPU based scaling in cloud
provides with no human in the loop

- Saved thousands of CPU cores & instances after rollout
- Reduced scheduled-to-start latencies



Figure showing the change in worker service CPU utilization, decision task latency and # of instances after the worker auto-scaling feature was turned on

Replication Cache

Retain a cache for replication messages in source cluster replicate directly from memory

- **71%** reduction in reads to history_node
- **23+%** reduction in all Cassandra reads
- **13%** reduction in CPU

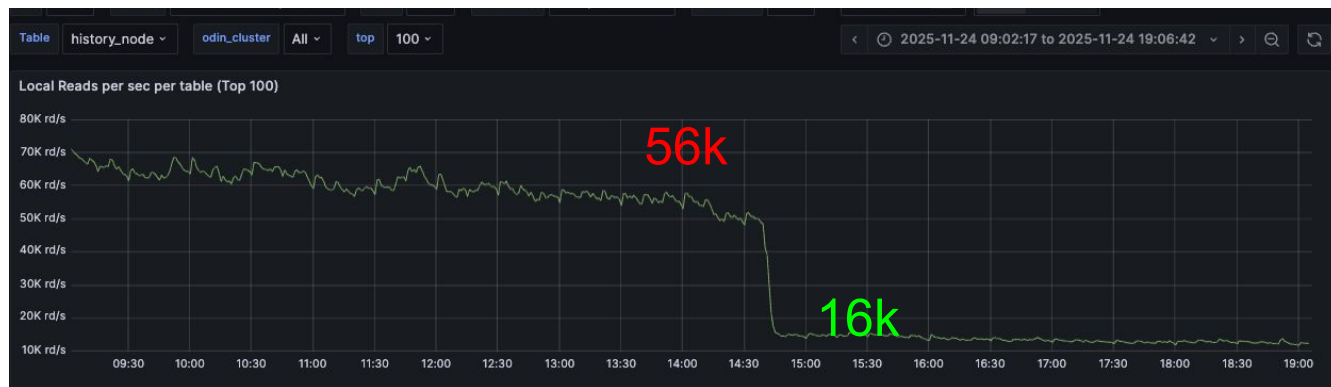


Figure showing the drop in read volume to the history_node (biggest) table in one of the Cassandra instances after replication cache was enabled

Replication Lag Improvements

Series of improvements in replication helped reduce the cross region replication lag

13s => 2s

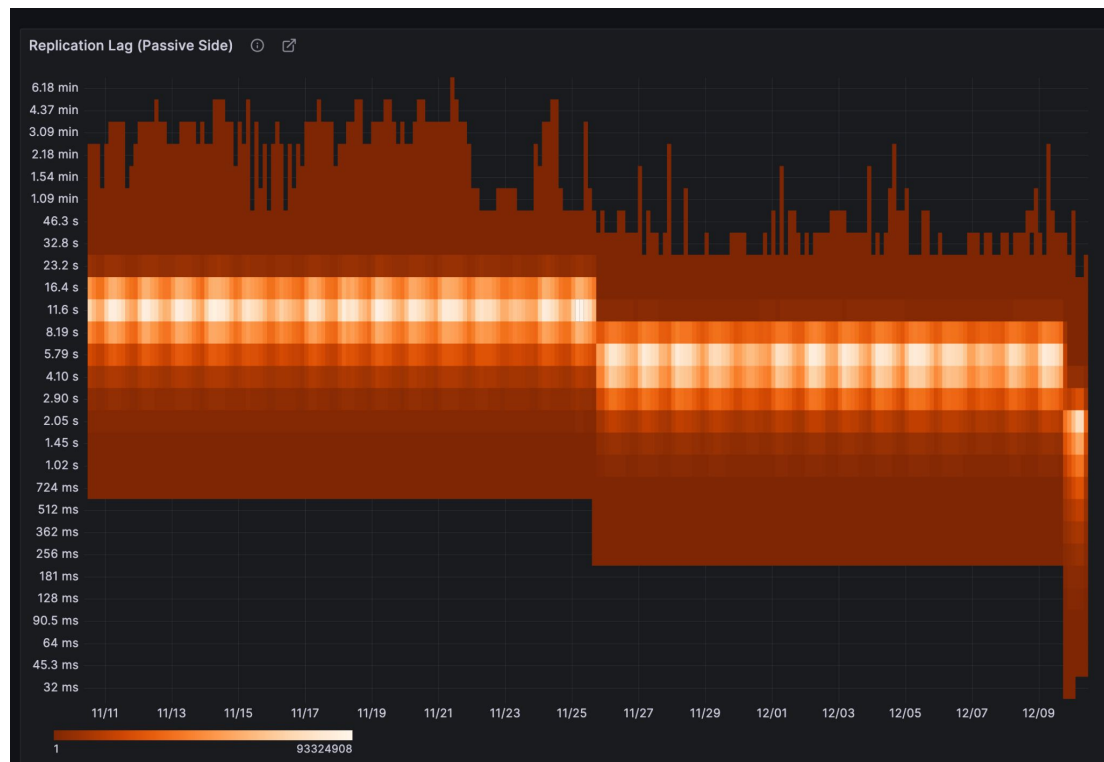
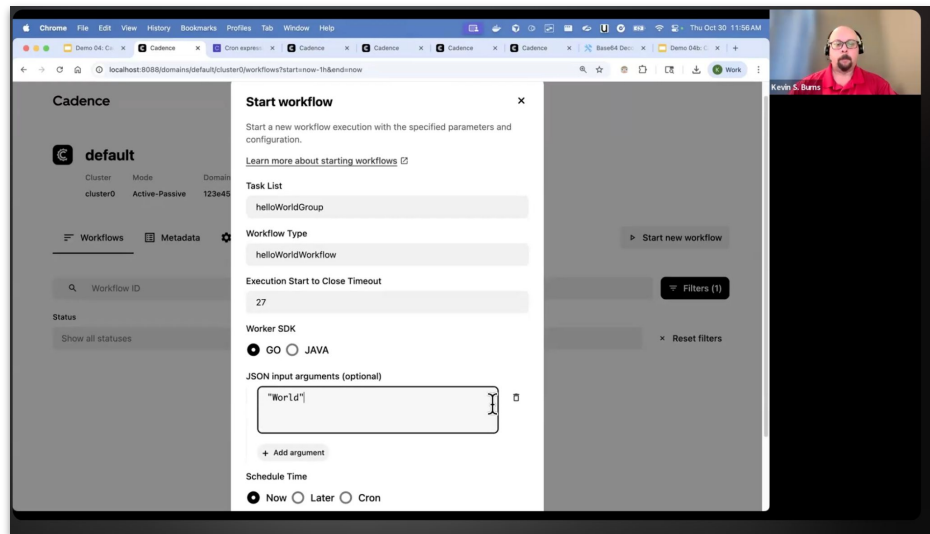
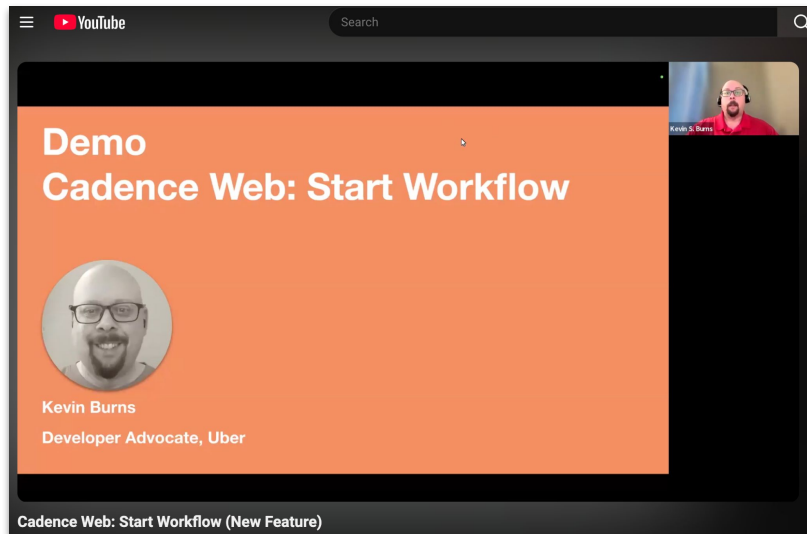


Figure showing replication latency changes over time in a heatmap

New Cadence Codelabs, Demos & Wikis

New series of YouTube® videos showcasing new features.
Codelabs to help users try them and onboard on Cadence



Screenshots from a video showcasing new cadence-web feature allowing users directly start and signal workflows

Workflow Diagnostics

Workflow issues in the summary tab:
Functional:

- Workflow/Activity failures
- Non-determinism
- ...

Non-functional:

- History size violations
- Issues with tasklist setup
- Rate limits
- Timeouts
- ...

Codelab and tutorial video are available!

The screenshot shows the Cadence workflow summary page in a web browser. The workflow ID is 17aae020-bf75-4466-a720-2bd814c2348b, and the status is 'Timed Out'. The summary tab is selected, showing a message: '1 issue was detected on this workflow'. A 'View issue' button is present. The workflow details section shows the workflow name 'helloWorldWorkflow', the ID '17aae020-bf75-4466-a720-2bd814c2348b', and the type 'helloWorldWorkflow'. The failure details are shown in a code block:

```
decisionTaskFailedEventAttributes: {
  scheduledEventId: 2,
  startedEventId: 3,
  cause: "WORKFLOW_WORKER_UNHANDLED_FAILURE",
  details: "dW5hYmxlIHRvIGZpbmQgd29ya2Zsb3cgdHlwZTogbWVpbi5Xb3JrZmxvdy4gU3VwcG9ydGVkIHR5cGVzOiBbaGVsbG9Xb3JrZFdvcmtmbG93XQ==",
  identity: "14584@kburns2-H7V37R7YJR@helloWorldGroup@d45ac7d7-622f-45c2-8ca0-4c69ad613248",
  reason: "",
  binaryChecksum: "5939a37b7250d691fb096c6155a7062a"
}
```

Figures showing an issue warning on the workflow summary page and the zoomed details to the failure.

Workflow actions

Workflows can now be directly started and signaled from cadence-web service directly

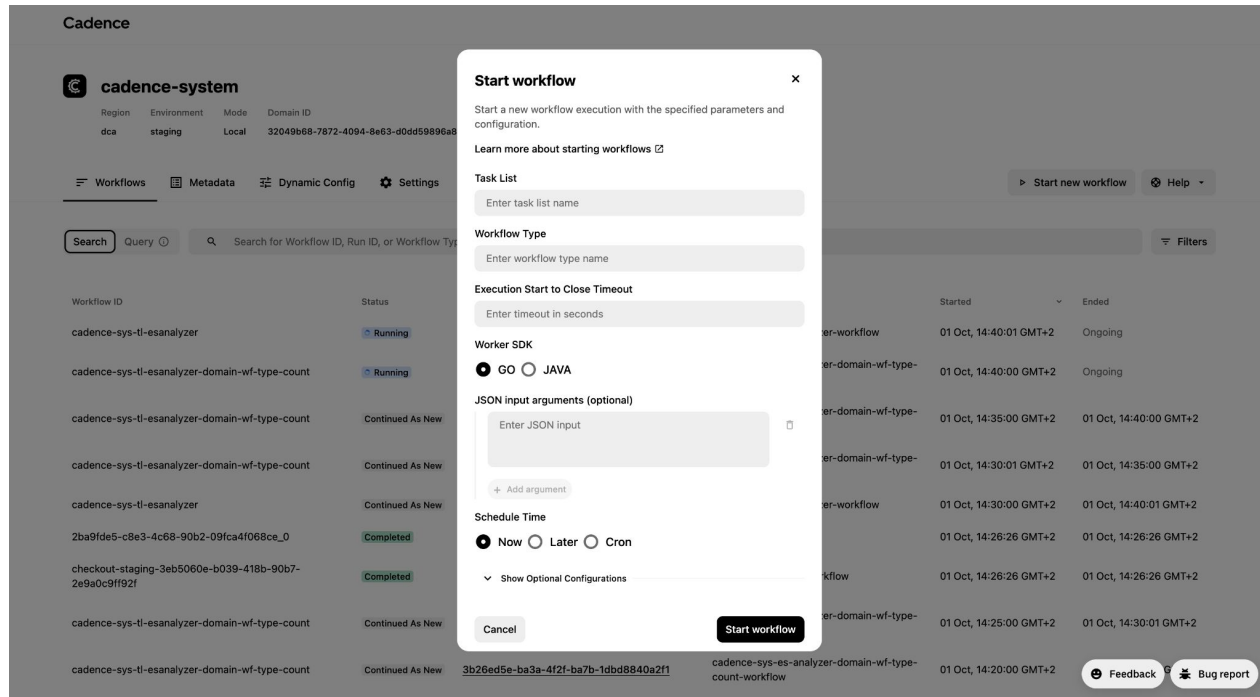


Figure showing a pop-up with start workflow parameters to start a workflow

Cadence-web visibility query autocomplete

Filtering Cadence workflows with advanced queries is a lot easier now!

The screenshot displays the Cadence web interface for workflow visibility. At the top, there are navigation tabs: Workflows, Metadata, Dynamic Config, Settings, and Archival. To the right of these tabs are buttons for 'Start new workflow' and 'Help'. Below the navigation bar, there is a search section with a 'Search' button, a 'Query' button with an information icon, and a query input field. The query input field contains the text '<> WorkflowType = "workflow.sanity" AND C'. To the right of the query input field is a 'Run Query' button. Below the query input field, an autocomplete dropdown menu is open, showing two suggestions: 'CloseTime' and 'CloseStatus'. Below the dropdown menu, there is a table of workflow details. The table has columns for Workflow ID, Status, Name, Workflow Type, Start Time, and End Time. The first row shows a workflow with ID 'cadence.canary.cron', status 'Running', name 'workflow.cron', type 'workflow.cron', start time '30 Sep, 11:00:58 PDT', and end time 'Ongoing'. The second row shows a workflow with ID 'cadence.canary.cron-workflow.sanity-2025-09-30T18:00:53Z/workflow.cancellation', status 'Canceled', name 'workflow.cancellation', type 'workflow.cancellation', start time '30 Sep, 11:00:56 PDT', and end time '30 Sep, 11:00:56 PDT'.

Workflow ID	Status	Name	Workflow Type	Start Time	End Time
cadence.canary.cron	Running	workflow.cron	workflow.cron	30 Sep, 11:00:58 PDT	Ongoing
cadence.canary.cron-workflow.sanity-2025-09-30T18:00:53Z/workflow.cancellation	Canceled	workflow.cancellation	workflow.cancellation	30 Sep, 11:00:56 PDT	30 Sep, 11:00:56 PDT

Figure showing autocomplete suggestions in the query bar while the user is typing their query

Cadence Web v4

Announcing cadence-web v4! A new modern look for Cadence!

- Vue.js => React
- Mobile friendly
- Completely redesigned the UI
- Added multi environment support
- Significantly improved the performance

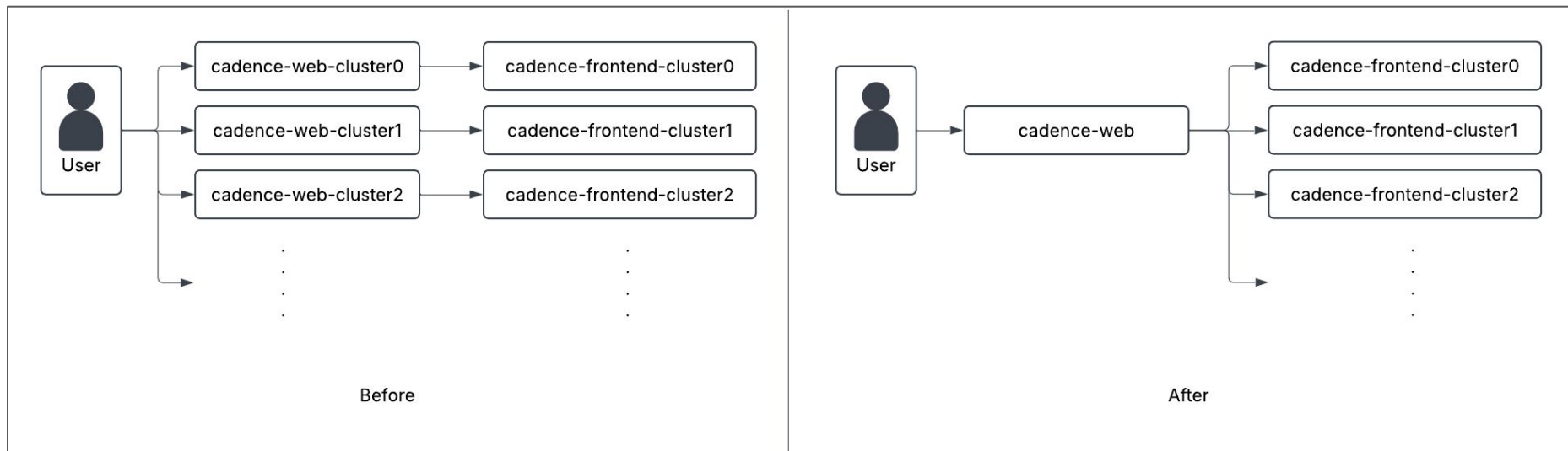


Figure showing single cadence-web service talking to multiple Cadence environments

New History Page

A new more compact and expressive history view!

- **Redesigned Events List:** presented in aligned, color-coded columns for easy scanning
- **Improved Loading Speed:** eager loading, load times reduced 98% for larger workflows
- **Sticky Header & Quick Navigation:** Modify your view without losing your place. Quick actions like expanding/closing all panels, instantly jumping to failed or pending events.

Workflow history		Switch to the legacy History view ⓘ		Export JSON		Grouped
Event group	Status	Time	Duration	Details		
● Workflow Started	✓ Started	28 Jan, 07:41:59 PST		[1769614918980851012,30000000000]		
● Decision Task	✓ Completed	28 Jan, 07:41:59 PST	27ms	⌛ 10s		
● Version Marker: Version	✓ Recorded	28 Jan, 07:41:59 PST				
● Workflow Search Attributes	✓ Upserted	28 Jan, 07:41:59 PST				
● Workflow Signaled	✓ Signaled	28 Jan, 07:42:00 PST		["canary.signal"] signal-name		
● Decision Task	✓ Completed	28 Jan, 07:42:00 PST	65ms	⌛ 10s		
● Workflow Completed	✓ Completed	28 Jan, 07:42:00 PST				

Figure showcasing the new history page

Custom Workflow Controls

Add custom workflow controls using Markdown!

- Supports familiar markdown format
- Allows user to add custom start/signal buttons
- Secure against custom JS code execution
- Use cases: smart runbooks, data pipeline controls, ...

Markdown Query Workflow

You can use markdown as your query response, which also supports starting and signaling workflows.

- Use the Complete button to complete this workflow.
- Use the Continue button just to send a signal to continue this workflow.
- Or you can use the "Start Another" button to start another workflow of this type.

Complete

Continue

Start Another

Figures showing how markdown queries with custom buttons look like in the Cadence UI (left), and how to add those (start and signal) buttons (right).

```
{% start
  workflowType="cadence_samples.HelloWorldWorkflow"
  label="Start a HelloWorld Workflow"
  domain="cadence-samples"
  cluster="cluster0"
  taskList="cadence-samples-worker"
  sdkLanguage="GO"
  input={message: "Cadence"}
  timeoutSeconds=7200
/%}
```

Start a HelloWorld Workflow

```
{% signal
  signalName="update_status"
  label="Update Status"
  domain="my-domain"
  cluster="my-cluster"
  workflowId="my-workflow-123"
  runId="my-run-456"
  input={status: "approved", user: "john"}
/%}
```

Signal with Payload Example

Roadmap

2026 Focus

- **Usability**
 - Time to write First Workflow: 1 minute
 - Special Scenarios
- **Agent Orchestration / Agentic Workflow Infra**
- **Adapting Modern Features**
 - Bringing highly loved features from other solutions
- **Operability**
 - Intuitive Error Modes operable directly from the UI
 - Improved Observability + Verbal Metric Interpretation
- **Maximum Adaptiveness**
 - Path to (almost) zero config
- **Seamless Portability**
 - Moving workloads across cloud providers/on-prem with no downtime