Hermes structured runs and approvals

Use these five endpoint-qualified capabilities when an application needs a Hermes run ID, pollable status, structured lifecycle events, a provider approval continuation, or an explicit stop request.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/hermes-agent/capabilities/durable-runs-and-approvals.md

The run is asynchronous, but its provider state is not durable. The pinned Hermes implementation keeps run status, event queues, active-task references, and approval queues in one process. A restart loses them. Treat `run_id` as a temporary handle, not as a durable AIP execution record.

Operations

Replace `` with the normalized endpoint ID present in AIP discovery.

Operation	Provider request	Risk	AIP approval	Key	Retry
`run_start`	`POST /v1/runs`	Medium	No	Required	Unsafe
`run_status`	`GET /v1/runs/{run_id}`	Low	No	Optional	Safe
`run_events`	`GET /v1/runs/{run_id}/events`	Low	No	Optional	Safe read
`run_approval`	`POST /v1/runs/{run_id}/approval`	Medium	Yes	Required	Unsafe
`run_stop`	`POST /v1/runs/{run_id}/stop`	Medium	Yes	Required	Unsafe

All five are stable AIP Workflow capabilities. Every provider request requires the selected endpoint's Bearer token. The endpoint comes from the capability ID; input cannot select another endpoint or base URL.

`run_events` is a safe read in the mutation sense. Reconnecting does not replay events already removed from the provider's single live queue. Do not interpret the retry marker as an event-delivery guarantee.

Choose the run lifecycle

Need	Capability	Completion evidence
Obtain a handle immediately and let Hermes execute in the background	`run_start`	A `202` receipt with a new `run_id`
Learn the latest provider state	`run_status`	Current status object
Observe message, tool, approval, and terminal events	`run_events`	A terminal SSE event or an explicit incomplete result
Answer a tool approval that this exact run is waiting for	`run_approval`	Approval response plus later status or terminal event
Request interruption of an active run	`run_stop`	A `stopping` receipt plus later status or terminal event

Use a normal Chat or Responses capability when the application needs only one HTTP model turn. Use the governed operator capability when Hermes must delegate child AIP actions and retain a recoverable AIP-to-provider binding.

Shared AIP contract

Operation	Execution modes	Data contract	Provider timeout	AIP timeout
`run_start`	Sync receipt; async supported and expected	Confidential; PII; redact	60 s	60 s
`run_status`	Sync read	Internal; no PII marker; redact	30 s	60 s
`run_events`	Async, stream, cancel; streaming expected	Confidential; no PII marker; redact	No connector HTTP timeout	300 s
`run_approval`	Sync mutation	Restricted; PII; redact	30 s	60 s
`run_stop`	Sync mutation	Restricted; PII; redact	30 s	60 s

Start, approval, and stop publish `read`, `write`, and `external_network` side effects. Status and events publish `read` and `external_network`. None declares a transaction. Every compensation contract says rollback is not supported.

The AIP contract also publishes a 10-second maximum queue-delay hint and a `99.9%` availability target. These are declared values, not measurements or qualification evidence.

The status schema understates the possible data. A completed status can contain model output, and a failed status can contain a redacted error. Events can contain message text, tool names, previews, and approval detail. Handle both as confidential application data even though the published read contracts do not mark them as PII-bearing.

Start a run

The connector publishes an open object schema. Hermes still requires a truthy `input` value. A string is the least ambiguous form. A message array is also accepted when its final item has usable `content`.

Use the `body` object for provider input and keep transport controls at the AIP level:

```
JSON
{
  "body": {
    "input": "Prepare a concise incident summary.",
    "instructions": "Do not perform external mutations.",
    "model": "hermes-agent",
    "session_id": "incident-review-2030-01"
  },
  "session_key": "support:incident-review"
}
```

The provider also accepts `conversation_history` and `previous_response_id`. An explicit, non-empty history takes precedence. When no explicit history is present, Hermes can load the history and session ID stored for a previous Responses object. A missing previous response does not fail the start; Hermes continues without that stored history.

For a multi-message `input`, Hermes turns all usable items except the last into history. The final item's `content` becomes current input. Supply valid message objects and end with the visible user request.

A successful start returns immediately:

```
JSON
{
  "run_id": "run_0123456789abcdef0123456789abcdef",
  "status": "started"
}
```

The provider HTTP status is `202`. This receipt proves only that Hermes created the in-process run structures and scheduled background work. It does not prove that the model started, a tool succeeded, or the run reached a terminal state.

Keep run and continuity identities separate

Identity	Owner	Purpose
AIP Action ID	AIP runtime	Governs the invocation, result, evidence, and cancellation
AIP idempotency key	AIP runtime contract	Detects replay of the same capability action
Hermes `run_id`	Selected Hermes process	Addresses status, events, approval, and stop
Hermes `session_id`	Hermes conversation path	Selects task and conversation continuity
Hermes `session_key`	Hermes memory path	Scopes long-term memory through `X-Hermes-Session-Key`
Provider approval queue key	Hermes process	Is always the new `run_id`, never a client session ID

If `session_id` is absent, Hermes uses the new `run_id` as the conversation task ID. Several runs may intentionally share one explicit session ID or memory key. They still receive isolated approval queues, so approval for one run cannot release another run's pending command.

Trusted execution adds AIP principal, tenant, issuer, trace, delegation, session, idempotency, and deadline headers. The connector verifies an endpoint's configured tenant binding before it sends the request.

Poll run status

Call `run_status` with the exact provider handle:

```
JSON
{
  "run_id": "run_0123456789abcdef0123456789abcdef"
}
```

The connector percent-encodes the path segment and rejects carriage return, line feed, and NUL characters. The provider returns `404 run_not_found` when the handle is absent from that process.

The current status object can contain these fields:

Field	Meaning
`object`, `run_id`	`hermes.run` and the provider handle
`status`	Current lifecycle state
`created_at`, `updated_at`	Provider Unix timestamps
`session_id`, `model`	Effective conversation ID and selected model
`last_event`	Latest recorded lifecycle event name
`output`, `usage`	Final text and token counters after completion
`error`	Redacted provider error after failure

Interpret the states as follows:

State	Meaning	Next action
`queued`	Run structures exist; execution has not entered the agent body	Poll with backoff or subscribe once
`running`	Agent work is active, or an approval was answered	Continue observing
`waiting_for_approval`	This run has at least one pending provider approval	Review the event and use governed `run_approval`

State	Meaning	Next action
`stopping`	Hermes accepted a stop request	Keep polling; interruption is not yet proved
`completed`	Agent returned without its structured failure marker	Retain output and usage with AIP evidence
`failed`	Agent returned a failure marker or raised	Inspect the redacted error and side effects
`cancelled`	The run task processed cancellation	Treat prior tool effects as potentially committed

Terminal status is retained in memory for one hour after its last update, then becomes eligible for the periodic sweep. There is no run-list operation. A missing status can mean an unknown ID, retention expiry, or process restart.

Consume run events

`run_events` opens `GET /v1/runs/{run_id}/events` and aggregates provider SSE frames. It accepts `run_id` and an optional positive `max_events`. The connector defaults to 4096 parsed events.

The provider can emit:

- `message.delta` for generated text;
- `tool.started` and `tool.completed` for tool progress;
- `reasoning.available` for exposed reasoning previews;
- `approval.request` and `approval.responded` for provider approval flow;
- `run.completed`, `run.failed`, or `run.cancelled` as terminal events.

Hermes sends a keepalive comment after 30 seconds without an event. Comments do not enter the connector's event array. The provider sends `: stream closed` after its queue sentinel; that comment is not a terminal event by itself.

The queue is live, in memory, and consumed destructively. There is no cursor, history store, fan-out, or supported resume. Use one subscriber per run. A second subscriber can split events with the first. When a subscriber closes, Hermes removes the public queue handle, so a reconnect can return `404` while the underlying run continues.

If a stream ends without a recognized terminal event, the connector marks the AIP action incomplete rather than completed. If `max_events` truncates the aggregate, `terminal` is false. Poll `run_status` before deciding what happened.

Detailed frame parsing, AIP stream publication, disconnects, and cancellation are owned by the streaming guide and cursor reference.

Resolve a pending provider approval

An `approval.request` event contains a redacted command where applicable and the provider choices `once`, `session`, `always`, and `deny`. Review that event before submitting a decision.

There are two different approval gates:

Gate	What it authorizes
AIP approval on the `run_approval` Action	Sending one protected mutation to the selected Hermes endpoint
Hermes provider approval choice	Releasing or denying pending tool work inside that exact run

The first does not automatically decide the second. After tenant policy grants the AIP action, submit the intended provider choice:

```
JSON
{
  "run_id": "run_0123456789abcdef0123456789abcdef",
```

```
  "choice": "once",
  "resolve_all": false
}
```

The published schema also accepts `approve` and `allow`; Hermes normalizes them to `once`. Prefer the four canonical choices so stored intent matches provider events. Set `all` or `resolve_all` only when the reviewer has assessed every currently queued approval for this run. Resolution never crosses into another run, even when both share a session ID.

The success object reports `run_id`, normalized `choice`, and the number of resolved queue entries. Hermes returns `409 approval_not_active` when the run no longer owns an active approval session, or `409 approval_not_pending` when the queue has no pending item.

The AIP capability requires tenant-policy approval with reason, input snapshot, and policy-decision evidence. The approval expires after 15 minutes. It is a separate record from the provider's in-memory queue.

Stop active work

Call `run_stop` only for a run that is still active:

```
JSON
{
  "run_id": "run_0123456789abcdef0123456789abcdef"
}
```

The capability requires AIP approval. Hermes sets status to `stopping`, calls the active agent's interrupt hook, cancels the asynchronous wrapper task, and waits at most five seconds for that task. It then returns:

```
JSON
{
  "run_id": "run_0123456789abcdef0123456789abcdef",
  "status": "stopping"
}
```

This receipt is not a terminal cancellation guarantee. The agent runs in an executor thread that task cancellation cannot preempt directly. A current tool call can finish, and effects that already reached another system are not rolled back. Poll status and retain a terminal event when available.

Stopping a completed run returns `404` because Hermes has already removed its active agent and task references. Runtime cancellation dispatches the same provider stop operation only when the cancelled AIP action input contains a `run_id`; cancelling `run_start` before its receipt has no provider handle to stop.

Understand retention and process loss

Provider state	Storage	Current retention boundary
Active agent and task references	Process memory	Removed on completion or cancellation
SSE queue	Process memory	Removed after subscription closes or by the orphan sweep
Approval routing	Process memory	Removed after the run settles, stops, or is swept
Terminal status	Process memory	Eligible for sweep 3600 seconds after update
Non-terminal status	Process memory	No terminal TTL rule; lost on restart

Run streams become eligible for the orphan sweep 300 seconds after creation. The sweep runs once per minute. It removes active references and provider approval routing but does not prove that an executor thread stopped. A run that exceeds this window can become unmanageable through events, approval, or stop even while underlying work is still settling.

Subscribe promptly and design runs to finish within the admitted deadline. Do not use the run API as a durable job store. Preserve AIP lifecycle and execution evidence outside this provider memory boundary.

Apply idempotency and retry rules

Start, approval, and stop require a capability-scoped AIP idempotency key. The published TTL is 24 hours, and collision behavior is ReturnOriginalResult. The Hermes run handlers do not enforce the forwarded `Idempotency-Key` header.

Consequently, durable replay protection depends on the AIP runtime's configured idempotency backend. Repeating `run_start` directly at Hermes creates another run. Repeating an approval or stop after provider success commonly returns a conflict or not-found response rather than the original result.

Use these recovery decisions:

Uncertain operation	Safe next check	Do not assume
Start returned no `run_id`	Inspect the AIP action record and runtime idempotency result	That a replay will find the first provider run
Approval lost its response	Poll status and inspect the remaining live event stream	That resending the choice cannot release another pending item
Stop lost its response	Poll status and look for a terminal event	That `stopping` or transport close means cancellation
Status read failed	Retry with bounded backoff	That absence after restart means the run never executed
Event stream disconnected	Poll status and preserve the evidence gap	That reconnect or connector retry can recover missed frames

The provider has no endpoint for listing runs or reconciling a lost start receipt. Do not auto-retry a mutating call after an unknown outcome.

Account for shared limits

Hermes applies one concurrency gate across Chat, Responses, and structured runs. The default is 10 in-flight agent runs; zero disables the cap. Saturation returns `429` with `Retry-After: 1`, although the connector's typed failure does not retain that response header.

For structured runs, the gate counts entries in the live run-stream map. A completed run whose event queue was never consumed can therefore occupy a slot until a subscriber drains it or the orphan sweep removes it. Consume the event stream even when the application also polls status.

The provider rejects request bodies over 10,000,000 bytes. The connector does not publish a separate response-size bound. Trusted AIP execution can end any operation at its action deadline, even when the connector or provider would otherwise keep waiting.

Handle failures

Symptom	Meaning	Response
Local schema or missing-ID rejection	The action cannot form a valid route or approval body	Correct the input; do not retry unchanged
Provider `400`	Invalid JSON, missing input, invalid history, or invalid approval choice	Correct the provider body
Provider `401`	Endpoint Bearer authentication failed	Verify the admitted endpoint credential revision
Provider `404`	Run state or active stop target is absent from this process	Check retention and restart history before concluding it never ran
Provider `409` on approval	No active approval session or no pending item	Refresh status and current events; do not force a replay
Provider `429`	Shared agent concurrency is saturated	Apply bounded backoff to a new start only when prior outcome is known
Transport or decode failure on mutation	Provider outcome can be unknown	Reconcile through AIP state, status, and events
Stream closes without terminal evidence	Event delivery is incomplete	Poll status; preserve the gap in execution evidence

The connector reports provider non-success bodies through a typed failure and marks mutating failures as non-retryable. Authentication, tenant mismatch, expired trusted deadlines, and runtime cancellation are separate failures.

Verify an integration

For each admitted endpoint, verify and retain evidence that:

1. discovery exposes exactly the five expected capability IDs and contracts;
2. one approved start returns a unique `run_id` and status becomes visible;
3. the single event consumer observes ordered frames and a terminal condition;
4. two runs sharing a session ID still have isolated approval queues;
5. approval requires AIP evidence and changes only the selected run;
6. stop is followed until `cancelled`, `failed`, or another explained outcome;
7. a disconnected stream is reconciled through status without claiming replay;
8. process restart and retention expiry are reported as evidence loss, not as

proof that no external effect occurred.

Record the AIP source revision, Hermes revision, connector and host artifact identities, endpoint ID, tenant binding, credential revision, timestamps, and retained results. Source presence alone is not qualification or live-product evidence.

Related documentation

- Hermes Agent capabilities
- Chat and Responses capabilities
- Actions and sessions
- Approvals and policy
- Errors and recovery