CrewAI connector changelog

This changelog records source-confirmed CrewAI connector baselines and the operator action required by each change. Artifact digests and retained release evidence, not the version string alone, identify a deployable release.

SECTION	Connectors
TYPE	Connector
PROTOCOL	AIP 1.0
SOURCE	docs/connectors/crewai/changelog.md

The reviewed commits remain in the workspace `1.0.0` line. Intermediate source commits are not presented as separately published semantic releases.

Current reviewed baseline

Property	Value
Connector version line	`1.0.0`
AIP protocol	`1.0`
Connector profile	`aip.connector.crewai.v1`
Reviewed AIP source	`97be86e9efedf07ecf1783b03800f683f107fb04`
Reviewed source time	`2026-07-26T10:41:19Z`
Sidecar package	`aip-crewai-sidecar` `1.0.0`
Default CrewAI package set	`crewai`, `crewai-core`, and `crewai-cli` `1.15.5`
Method-verification source	`bfa652a7be8637562cc9b0833f75d927a64552d1`
Public operations per fully admitted crew	10
Durable job operations	7
Control operations	3

This table identifies source behavior. A production record must also identify the Rust host digest, sidecar digest, crew-code digest, admission package, configuration revision, and qualification evidence.

1.0.0 reviewed hardening baseline

Source revision `97be86e9efedf07ecf1783b03800f683f107fb04` is the documentation baseline. It confirms the following surfaces.

Added

- standalone `aip-host-crewai` using the common connector-host lifecycle;
- immutable type, version, instance, and replica identity;
- operation-specific capabilities for status, events, cancel, batch, replay,

training, testing, knowledge, and memory;

- a frozen ten-operation catalogue verified against one CrewAI source commit;

- bounded input and output schemas for every operation;
- a lockfile-frozen Python sidecar artifact;
- durable job records keyed by AIP Action ID;
- canonical request hashing with collision rejection;
- ordered replayable sidecar events and cursor reads;
- explicit archive confirmation for terminal records;
- product-fleet image, execution, restart, outage, and recovery evidence paths.

Hardened

- sidecar Bearer authentication is required outside explicit loopback development;
- host-to-sidecar transport requires HTTPS outside loopback and disables redirects;
- token, descriptor, response, input, output, event, state, and retention bounds fail closed;
- journal admission persists before provider task scheduling;
- one exclusive nonblocking lock fences each journal to one process;
- journal replacement uses a same-directory temporary file and durability flushes;
- restored running records become failed with explicit uncertainty;
- runtime storage, connector health, lease, and drain state gate routing readiness;
- credential revision policy is checked before provider side effects;
- sidecar logs use Action identity while terminal events avoid raw exception text.

Changed

- one descriptor now includes an explicit operation allowlist;
- one crew can expose up to ten distinct capabilities instead of one base run capability;
- startable operations use `/jobs` and `/jobs/stream` as the canonical private routes;
- `/runs` aliases remain only for legacy run compatibility;
- status and events target a durable Action through `run_action_id`;
- cancellation support is advertised only for run, batch, replay, train, and test;
- sidecar health must prove a durable journal, every configured crew, and every

descriptor-admitted operation;

- horizontal capacity uses independent connector instances instead of shared

sidecar state.

Known limitations

- provider exactly-once effects are not guaranteed;
- cancellation does not prove termination of CrewAI worker threads or external

tools;

- `remote_stop_confirmed: true` proves only sidecar acknowledgement;
- the sidecar has no Prometheus metrics route;
- the deterministic qualification crew does not call an external model or

tool provider;

- the pinned fleet procedure does not execute every operation end to end;
- workspace version `1.0.0` does not distinguish every source hardening commit.

Earlier 1.0.0 source baseline

The AIP Core `1.0.0` source commit `034a520608eea44e6b14e61c34734bd402d989c0` contained an earlier CrewAI connector shape.

That state had:

- one descriptor with ID, name, and optional description;
- one base capability for each configured crew;
- run, streamed run, and cancellation mapping through the sidecar;
- optional sidecar Bearer configuration;
- a simpler HTTP client and error inventory;
- no standalone `aip-host-crewai` crate;
- no fixed ten-operation catalogue;
- no descriptor operation allowlist;
- no reviewed durable multi-operation journal and process fence;
- no CrewAI entry in the production product-fleet topology.

The comparison describes two source states with the same workspace version. It does not establish that an earlier container was publicly released.

Migration from the earlier source shape

Apply these actions before moving an existing integration to the reviewed baseline.

Area	Required action
Artifact identity	Build or select separately admitted host and sidecar digests
Deployment	Run the Rust host and Python sidecar as distinct processes
Descriptors	Add exact operation allowlists and review generated capability IDs

Area	Required action
Routing	Register type, version, instance, replica, tenant, account, endpoint, and lease identity
Sidecar URL	Use private HTTPS or explicit loopback HTTP; remove credentials, query, and fragment
Authentication	Project one owner-controlled token to both processes
State	Provision one protected persistent volume per sidecar instance
Training	Add protected training storage before enabling `train`
Identity	Preserve Action ID and idempotency key across recovery
Operations	Update callers from a single run capability to operation-specific IDs
Events	Resume durable events by target Action ID and cursor
Cancellation	Treat route success as sidecar evidence, not provider termination
Qualification	Re-run source, image, execution, restart, and failure gates

Do not migrate by attaching the new sidecar to an unreviewed old host. The private route, request model, operation metadata, health contract, and failure semantics must remain one compatible pair.

Compatibility rules

The reviewed connector remains compatible with AIP `1.0` at the native wire boundary. The CrewAI-specific profile and private sidecar contract have their own compatibility requirements.

A change requires a new artifact identity and requalification when it affects:

- a capability ID, kind, schema, risk, approval, idempotency, or side-effect

contract;

- a sidecar route, request, response, event, status, auth, or timeout rule;
- CrewAI package versions or the method-verification source revision;
- crew factory code, model, tool, knowledge, memory, or training behavior;
- request hashing, journal format, locking, persistence, or archive behavior;
- host signing, registry identity, credential policy, storage, or lifecycle;
- an image base, runtime user, entrypoint, dependency graph, or trust evidence.

Documentation-only clarification can retain the artifact identity when it does not change code, schemas, manifests, examples that define behavior, or qualification claims.

Upgrade acceptance record

For each adopted baseline, record:

```
TEXT
Connector version:
AIP source revision:
Host image digest:
Sidecar image digest:
Sidecar lockfile digest:
CrewAI package versions:
Crew source digest:
Admission package digest:
Configuration revision:
Qualification run and evidence digest:
Migration reviewer:
Accepted at:
Rollback pair:
```

Never record only `1.0.0`. That string can describe multiple source and image states in the reviewed history.

Changelog maintenance

Add a new entry only after the corresponding source and artifact identities are fixed. Each entry must list added, changed, deprecated, removed, fixed, security, migration, compatibility, qualification, and known-limitation information when applicable.

Keep planned work outside this file. Mark a surface deprecated only when a committed artifact defines the replacement and lifecycle.

Related documentation

- CrewAI connector overview
- Deploy the CrewAI connector
- CrewAI sidecar contract and lockfile
- CrewAI idempotency, cancellation, and process fencing
- Qualify the CrewAI connector