Cross-company Workflow Runner v0

Cross-company workflow runner v0

Last verified: 2026-05-25

What it is

`~/system/tools/cross-company-workflow.js` is a deterministic wrapper around Mission Control and Company Mesh.

Use it when a task needs a bounded multi-company loop such as:

- CodeCraft builds or reviews an implementation plan.
- Securion reviews security/privacy risk.
- Proveo validates acceptance or QA evidence.

It is **not** a replacement for Mission Control ownership, Mehanik/Prompt-Forge gates, or Proveo verification. It only makes the cross-company handoff repeatable and evidence-producing.

Source files

- Runner: `~/system/tools/cross-company-workflow.js`
- Deterministic smoke workflow: `~/system/workflows/company-mesh-tool-smoke.json`
- Strong-model workflow smoke: `~/system/workflows/company-mesh-strong-review-smoke.json`
- Smoke test wrapper: `~/system/tools/tests/cross-company-workflow-smoke.sh`
- Evidence output default: `/tmp/alai/cross-company-workflows/`

Commands

Validate a workflow:

```
node ~/system/tools/cross-company-workflow.js validate ~/system/workflows/company-mesh-tool-smoke.json --json
```

Run a workflow:

```
node ~/system/tools/cross-company-workflow.js run ~/system/workflows/company-mesh-tool-smoke.json --json
```

Inspect state:

```
node ~/system/tools/cross-company-workflow.js status /tmp/alai/cross-company-workflows/<state>.json --json
```

Finalize MC tasks after a PASS workflow:

```
node ~/system/tools/cross-company-workflow.js finalize /tmp/alai/cross-company-workflows/<state>.json \
  --bookstack https://docs.alai.no/link/184 \
  --json
```

For static plumbing-only responder runs, finalization is intentionally blocked unless explicitly marked:

```
node ~/system/tools/cross-company-workflow.js finalize /tmp/alai/cross-company-workflows/<state>.json \
  --bookstack https://docs.alai.no/link/184 \
  --allow-plumbing-finalize \
  --json
```

Workflow JSON shape

Minimum example:

```
{
  "name": "example-review-loop",
  "description": "Bounded build/security/QA review loop.",
  "priority": "M",
  "actor": "john",
```

```
  "responderMode": "gemini-review",
  "steps": [
    {
      "id": "codecraft",
      "company": "CodeCraft",
      "title": "CodeCraft review",
      "purpose": "Review implementation approach.",
      "prompt": "Review the pasted evidence. Return PASS/PARTIAL/BLOCKED first.",
      "endState": "ANSWERED"
    },
    {
      "id": "securion",
      "company": "Securion",
      "title": "Security review",
      "purpose": "Review security/privacy risks.",
      "dependsOn": ["codecraft"],
      "prompt": "Review security implications of the pasted evidence.",
      "endState": "ANSWERED"
    }
  ]
}
```

Required per step:

- `id`
- `company` or `agent`
- `purpose`
- `prompt` or `promptFile`

Optional per step:

- `dependsOn`
- `title`
- `priority`
- `endState` (`PASS`, `PARTIAL`, `BLOCKED`, `ANSWERED`, `DECLINED`)
- `responderMode` (`answer`, `blocked`, `decline`, `agent-runner`, `gemini-review`)
- `ttlSeconds`, `maxTurns`, `costCapUsd`, `timeoutSeconds`

Safety defaults

The runner is intentionally conservative:

- Uses JSON workflows only.
- Uses `spawnSync` argv arrays, not shell interpolation.
- `promptFile` must remain under the workflow file directory.
- MC child tasks use `route=general` so pi-orchestrator does not compete with the workflow.
- Responder execution is explicit per run/step; no daemon is started.
- State records the workflow SHA-256 and refuses status/finalize if the workflow file changed.
- Finalize refuses non-PASS workflows.
- Finalize refuses static responder evidence unless `--allow-plumbing-finalize` is provided.

Responder modes

Mode	Meaning	Use
`answer`	Static deterministic response	Plumbing-only smoke tests
`blocked`	Static blocked response	Negative-path plumbing tests
`decline`	Static decline response	Policy/eligibility tests
`agent-runner`	Local persona via `agent-runner.js`	Cheap local advisory, subject to claim gate
`gemini-review`	Cloud strong-model advisory via Gemini CLI	Bounded strong review; keep prompts evidence-based and cost-limited

Claim-gate override

`--claim-gate-off` passes `COMPANY_MESH_CLAIM_GATE=off` to responder execution. Treat this as a break-glass/debug option only.

Use it only when all are true:

- the workflow is plumbing-only or local debugging;
- no production, deploy, legal, financial, security, or customer-facing completion claim will be made from the output;
- the final evidence explicitly says claim gate was disabled;
- MC finalization is either skipped or marked as plumbing-only.

Do **not** use `--claim-gate-off` to make blocked `agent-runner` responses look valid. If claim gate blocks a response, prefer adding concrete evidence paths/pasted evidence or returning `PARTIAL`/`BLOCKED` honestly.

Evidence model

A run writes:

- state JSON (`0600`) containing parent task, child tasks, message ids, responses, and end states.
- markdown evidence summary next to the state JSON.
- responder evidence JSON under the chosen evidence directory.

Evidence should be treated precisely:

- Company Mesh delivery proves delivery only.
- Static `answer` proves plumbing only.
- `agent-runner`/`gemini-review` output is advisory unless it cites concrete evidence paths or pasted evidence.
- Mission Control `done` still requires normal validator/verdict gates.

Known gotchas

- `mc.js ready` requires task status `in_progress` or `ready_for_review`, validation notes, and `--bookstack <url>`.
- Claim gate can block `agent-runner` outputs if the response makes factual completion claims without evidence paths.
- `gemini-review` uses the local `gemini` CLI and may fail if credentials/model access are unavailable.
- Public demo/product workflows must still obey product guardrails: no destructive public mutation, no fake deploy/integration claims.

Current verified status

Completed evidence exists for the initial Company Mesh workflow runner smoke:

- `/tmp/alai/company-mesh-handoff-20260523/mc-101896-cross-company-workflow-final-pass.md`
- `/tmp/alai/company-mesh-p2p-six-artifacts-consolidated-20260525T0210Z.md`

The next maturity step is a bounded non-smoke workflow using `gemini-review` or `agent-runner` against a real evidence artifact, with cost capped and no production mutation.

Revision #2
Created 2026-05-25 07:23:06 UTC by John
Updated 2026-05-25 07:30:31 UTC by John