

# Continuous Research Integration (CRI) — Specification (MVP)

**Status:** Draft (MVP)

**Scope:** Defines how this repository executes **Continuous Research Integration** (CRI) runs: *ingest* → *machine validation* → *human review* → *commit*.

**Goal:** Make research updates reproducible, auditable, and falsifiable by construction.

---

## 1. Goals & Non-Goals

**Goals - Reproducibility:** every run writes a timestamped, immutable folder under `runs/`.

- **Falsifiability:** claims must include predictions with tolerances or explicit falsification tests.
- **Human-in-the-loop:** two gates (Scope & Merge) with checklists and recorded approvals.
- **Minimal vendor lock:** local fallbacks work in CI; providers are pluggable later.

**Non-Goals (MVP)** - No external LLM providers required (local fallbacks only).

- No dataset downloads or networked replication (air-gapped default).
- 

## 2. Lifecycle (Happy Path)

1. **Trigger:** GitHub Actions workflow `awo-run.yml` (manual or on push).
  2. **Run:** `scripts/awo_run.py workflows/<name>.json`
  3. **Machine checks:**
    - `scope_validate` enforces testability.
    - `assert_contains` applies content guardrails.
  4. **Scope Gate:** Environment `awo-scope` renders preview summary for human approval.
  5. **Merge Gate:** Environment `awo-audit` presents final run facts for approval.
  6. **Publish:** Full run directory is committed to `runs/run_YYYY-MM-DDTHH-MM-SSZ/`.
-

### 3. Roles (Logical)

- **Proposer**: produces candidate outputs (e.g., simulated fan-out).
  - **Auditor-Static**: deterministic checks (`scope_validate`, `assert_contains`).
  - **Consensus**: reduces multiple outputs to a representative result.
  - **Human Gate**: scope + merge approvals recorded.
  - **Planned**: Refuter (adversarial rounds), Replicator (containerized re-run).
- 

### 4. Required Artifacts (Per Run)

Each run directory (`runs/run_<UTC_ISO>/`) includes:

**Always** - `index.json` — run metadata { `run_id`, `started_at`, `status`, `finished_at?` }.  
- `workflow_frozen.json` — exact workflow used.  
- `report.md` — human-readable step summary.  
- `steps/*.json` — structured logs of each step.  
- `scope/summary.json` — scope validation results.  
- `scope/claims/*.json` — frozen claim inputs.

**Conditional** - `gate_decision.yml` — only written if `audit_gate` triggers.  
- `approval.json` — only written after merge approval.

---

### 5. File / Folder Contract

```
workflows/
    multimodel.json      # example workflow
scripts/
    awo_run.py          # CRI runner (stdlib only)
.github/workflows/
    awo-run.yml        # CI runner
runs/
    run_YYYY-MM-DDTHH-MM-SSZ/
        index.json
        workflow_frozen.json
        report.md
        steps/
        scope/
            summary.json
            claims/*.json
```

```
gate_decision.yml      (conditional)
approval.json         (conditional)
```

---

## 6. Step Operations (MVP)

### 6.1 scope\_validate

- **Purpose:** enforce testability.
- **Output:** scope/summary.json, scope/claims/\*.json.

Schema (summary.json):

```
{  
    "claims_checked": 1,  
    "overall_ok": true,  
    "details": [  
        { "id": "claim-hello-001", "ok": true, "problems": [] }  
    ],  
    "notes": [],  
    "ts": "2025-09-12T01-31-09Z"  
}
```

Claim requirements:

```
{  
    "id": "string",  
    "statement": "string",  
    "predictions": [  
        { "name": "...", "function": "...", "tolerance": {"metric": "RMSE", "max": 0.05} }  
    ],  
    "falsification_tests": [  
        { "name": "...", "must_pass": true }  
    ]  
}
```

---

### 6.2 fanout\_generate

- Simulates multiple models (echo, upper, reverse).

---

### 6.3 consensus\_vote

- Normalizes outputs, counts agreement, emits consensus\_text.

#### **6.4 assert\_contains**

- Guards required content.
  - Fails the run if missing.
- 

#### **6.5 audit\_gate**

- Creates `gate_decision.yml`, halts run with status `pending_review`.
- 

### **7. Gates & UX**

- **Scope Gate (awo-scope)**: Preview + approve scope summary.
  - **Merge Gate (awo-audit)**: Preview + approve final run facts.
  - After approval, the run is auto-committed.
- 

### **8. Versioning & Invariants**

- Runner always writes `workflow_frozen.json` and `index.json` at start.
  - All timestamps UTC, format `YYYY-MM-DDTHH-MM-SSZ`.
  - Adding ops = backward compatible.
  - Breaking changes → bump minor version + ADR.
- 

### **9. Security & Privacy**

- No external network calls (MVP).
  - No secrets required.
  - Future adapters must handle secrets via GitHub encrypted secrets.
- 

#### **## 10. Roadmap**

- **Refuter**: adversarial rounds; halt if unaddressed failures remain.
- **Replicator**: containerized re-run + tolerance diff.

- **Provider adapters (future):** pluggable connectors to external LLM APIs.

*Status:* **not implemented in the MVP.** When added, adapters will:

- run behind a provider interface,
- log {provider, model, api\_version, params, seed, cost\_estimate} to steps/\*.json,
- respect repo-wide reproducibility settings (seeds/temp), and
- be optional—local fallbacks remain the default.

- **Consensus with dissent ledger:** record unresolved objections.
- 

## Appendix A — Example Workflow

```
{
  "name": "Multi-model consensus demo (CRI)",
  "steps": [
    {
      "id": "scope_1",
      "op": "scope_validate",
      "args": {
        "claim": {
          "id": "claim-hello-001",
          "statement": "A demo claim asserting the summary mentions battery life and confus",
          "predictions": [
            {
              "name": "mentions-battery",
              "function": "text-contains('battery lasts all day')",
              "tolerance": {"metric": "boolean", "max": 1}
            }
          ],
          "falsification_tests": [
            {"name": "mentions-ui-confusing", "must_pass": true}
          ]
        }
      }
    },
    {
      "id": "fanout_1",
      "op": "fanout_generate",
      "prompt": "Summarize: The battery lasts all day but the UI is confusing.",
      "models": ["echo", "upper", "reverse"],
      "params": { "seed": 123 }
    },
    {
      "id": "consensus_1",
    }
  ]
}
```

```
        "op": "consensus_vote",
        "inputs_from": "fanout_1"
    },
    {
        "id": "assert_consensus_quality",
        "op": "assert_contains",
        "args": {
            "from_step": "consensus_1",
            "field": "consensus_text",
            "must_include": [
                "battery lasts all day",
                "ui is confusing"
            ]
        }
    },
    {
        "id": "gate_review",
        "op": "audit_gate",
        "args": { "checklist": "templates/audit-checklist.md" }
    },
    {
        "id": "emit_consensus",
        "op": "write_text",
        "args": { "path": "notes/consensus.txt", "from_step": "consensus_1", "field": "consensus_text" }
    }
]
```