Fallback Architecture Blueprint

Specifies the reliability state machine every GROW agent must implement. The blueprint enumerates deterministic fallbacks, retry caps with backoff, recovery paths per failure mode, terminal-failure handling, and the HITL fallback contract. The state machine below is the canonical reference shape; implementations may add states but may not remove or rename these.

1. Reliability principles

- Deterministic fallback first, retry second. Before any retry, the agent attempts a deterministic fallback path (cached prior result, rules-based classifier, prior-shift snapshot, last-known-good template). Retries without a fallback are the primary cause of looping-retry.
- Bounded retries with jittered exponential backoff. Default cap is 3 retries per step, 6 retries per run, with backoff min(60s, base 2^attempt) + jitter(0, base/2), base=2s. Connector-specific overrides allowed.
- Every state transition emits an event. Events feed S3 provenance and S2 eval scoring. Silent transitions are a bug.
- Terminal failure is preferable to wrong success. A clear failed-terminal state with evidence is better than a succeeded state with a hallucinated post-condition.

2. State diagram

```
stateDiagram-v2
    [*] --> Intake
    Intake --> Plan: input valid
    Intake --> Quarantined: schema-drift-input / prompt-injection-detected
    Plan --> Execute: confidence_band in {high, medium} AND no boundary
    Plan --> AwaitingHITL: boundary OR confidence_band=low
    Plan --> Halted: confidence_band=unknown
    Execute --> Verify: tool result returned
    Execute --> Fallback: tool-timeout / connector-auth-failure / rate-limit-exceeded
    Execute --> Halted: unsafe-action-attempted / pii-leak-risk
    Fallback --> Verify: deterministic path produced result
    Fallback --> Retrying: no deterministic path available
    Retrying --> Execute: retry budget remaining
    Retrying --> Escalated: retry cap reached / looping-retry
    Verify --> Succeeded: post-condition passed
    Verify --> Fallback: false-success-report OR hallucinated-citation
    Verify --> Escalated: verification ambiguous
    AwaitingHITL --> Execute: reviewer approves (permit-with-modification)
    AwaitingHITL --> Halted: reviewer refuses
    AwaitingHITL --> Escalated: HITL SLA exceeded
    Quarantined --> Escalated: always
    Escalated --> Succeeded: reviewer resumes with override
    Escalated --> FailedTerminal: reviewer terminates OR Tier-3 timeout
    Halted --> FailedTerminal: no resume path
    Succeeded --> [*]
    FailedTerminal --> [*]
```

State definitions:

- Intake: input schema validation, injection scan, idempotency check.
- Plan: model produces a candidate plan and a routing_confidence with confidence_band.

- Execute: tool calls run, side effects staged.
- Verify: post-condition check (write-confirm, downstream ack, citation resolution).
- Fallback: deterministic alternative path. Logged as decision_origin=fallback.
- Retrying: bounded retry with backoff, only after fallback exhausted.
- AwaitingHITL: reviewer queue with evidence pointer.
- Quarantined: isolated input, no execution permitted until cleared.
- Escalated: routed up the tier ladder per threshold spec.
- Halted: clean stop, recoverable via reviewer action.
- Succeeded / FailedTerminal: terminal states; emit final provenance frame.

3. Retry policy (defaults)

```
per_step_cap: 3
per_run_cap: 6
backoff:
  base_seconds: 2
  exponent: 2
  max_seconds: 60
  jitter_seconds: 1
classes_excluded_from_retry:
  - unsafe-action-attempted
  - pii-leak-risk
  - prompt-injection-detected
  - false-success-report (until fallback re-grounds)
classes_with_immediate_retry_zero:
  - connector-auth-failure (refresh-then-retry once)
loop_detector:
  same_step_hash_threshold: 2
  stagnant_state_window_seconds: 30
```

Loop detector triggers looping-retry and forces transition to Escalated regardless of remaining retry budget.

4. Recovery paths by failure mode

Every entry in the Failure-Mode Register has a recovery_path field. The blueprint canonicalizes the shapes:

- fallback -> bounded-retry -> escalate: tool-timeout, rate-limit-exceeded
- escalate-to-operator -> credential-refresh -> resume: connector-auth-failure
- terminate-run -> HITL gate -> postmortem: unsafe-action-attempted
- HITL-review -> resume-with-human-decision: low-confidence-routing, ambiguous-intent
- rollback-if-possible -> escalate -> regression-seed: false-success-report
- quarantine -> HITL-review -> add to adversarial suite: prompt-injection-detected, schema-drift-input
- re-ground -> HITL-if-needed -> regression-seed: hallucinated-citation
- block -> redact -> HITL + compliance review: pii-leak-risk
- refresh -> re-evaluate -> escalate-if-stale: stale-data

- rehydrate -> retry-once -> escalate: state-corruption
- freeze -> investigate -> rollback-or-rerelease: silent-degradation

If a new failure mode is added to the register without a canonical recovery path, the default is fallback -> escalate -> FailedTerminal.

5. Terminal failure handling

On entering FailedTerminal:

1. Persist the full event chain and evidence bundle to the S3 provenance store.

2. Emit a structured incident with failure_id, severity, decision_origin=fallback|escalation, and the canvas-declared escalation target.

3. If reversibility was partially-reversible or irreversible, run the rollback playbook from the agent's canvas; do not auto-resume.

4. Tag the run for inclusion in the next S2 regression sweep if severity in {critical, high}.

5. Open a hardening task in the HITL Policy's hardening backlog when the same failure_id fires twice in 7 days on the same agent.

6. HITL fallback contract

When the system cannot resolve via deterministic fallback or bounded retry, HITL is itself a fallback. The contract:

- The reviewer queue is treated as a tool with a timeout (the SLA from the threshold spec).
- If the SLA expires, the agent does not invent a decision. It escalates one tier up or transitions to FailedTerminal.
- The reviewer's decision becomes a first-class event with decision_origin=human-override or escalation and is logged per the HITL Policy schema.
- Recurring HITL interventions on the same failure_id graduate to hardening tasks: a new deterministic fallback, a tightened threshold, or a model/prompt change.

This blueprint is intentionally conservative. Performance tuning happens after the state machine is honored end-to-end and S2 evals are green.