AXILOG

SRS-002

Agent Totality & Health FSM Specification

Locked v0.3 L5 [D2]

Last updated 26 March 2026
Depends on SRS-001 · DVEC-001 · AXIOMA-FRAMEWORK
Changelog 1 entry

Contents

Depends on: SRS-001 v0.3 · DVEC-001 v1.3 · AXIOMA-FRAMEWORK v0.4

---

1. PURPOSE

This document defines the **Agent Totality Contract** for the Axioma framework.

It specifies the formal requirements for:

- deterministic agent behaviour
- health state management
- oracle input admission
- replay equivalence
- fail-closed execution

Objective: Every agent is a total, bounded, deterministic function over admitted inputs and internal state.

---

2. DEFINITIONS

2.1 Agent

An **agent** is a deterministic state machine executing on top of the Axioma audit substrate (L6).

2.2 Admitted Observation (AX:OBS:v1)

An **admitted observation** is:

- canonicalised input
- committed to the ledger
- immutable once admitted

2.3 Transition Record (AX:TRANS:v1)

A **transition record** is:

- the canonical representation of a state transition

- committed prior to state mutation

2.4 Health State

The **health state** represents the operational status of the agent.

2.5 Terminal State

A **terminal state** is a state from which no further transitions are permitted without external reset.

2.6 No-Op Transition

A **No-Op transition** is a deterministic transition that preserves the current state while maintaining ledger continuity.

2.7 Fault Accumulator

A **fault accumulator** is a deterministic counter tracking fault events for threshold-based state transitions.

2.8 Layer Terminology

Term	Layer	Meaning
`FAILED`	L6 (Ledger)	Substrate cannot accept commits
`STOPPED`	L5 (Agent)	Agent terminal state

NOTE: `L6 FAILED ≠ L5 STOPPED`, BUT: `L6 FAILED ⇒ L5 STOPPED` (mandatory propagation).

---

3. DETERMINISM MODEL

3.1 Determinism Class

The agent SHALL operate under:

D2 — Constrained Deterministic

3.2 Determinism Definition

SRS-002-SHALL-001

For identical:

- initial state
- ordered sequence of admitted `AX:OBS:v1` inputs

the agent SHALL produce:

- identical sequence of `AX:TRANS:v1` records
- identical resulting state

3.3 Oracle Boundary

SRS-002-SHALL-002

All external inputs MUST be admitted as `AX:OBS:v1` before use.

Direct system calls (time, IO, randomness) are **FORBIDDEN**.

---

4. HEALTH STATE MACHINE

4.1 State Set

The agent SHALL define the following states:

```
UNINIT → INIT → ENABLED → ALARM → DEGRADED → STOPPED
```

4.2 Terminal State

SRS-002-SHALL-003

`AX_HEALTH_STOPPED` SHALL be a terminal state.

4.3 Terminal Behaviour

SRS-002-SHALL-004

If `state == STOPPED`:

- no state mutation SHALL occur
- all transition attempts SHALL be rejected
- a violation SHALL be raised

4.4 Runtime Fault Coupling

SRS-002-SHALL-005

If the underlying ledger enters `FAILED`:

- agent health MUST transition to `STOPPED`

4A. INITIAL STATE BINDING

4A.1 Genesis Binding Requirement

SRS-002-SHALL-026

An agent SHALL only transition:

```
UNINIT → INIT
```

if:

- ledger context is initialised
- `genesis_hash` matches system golden reference

4A.2 Violation Behaviour

If mismatch occurs:

- violation SHALL be raised
- agent SHALL transition to `STOPPED`

---

5. TRANSITION PRE-COMMIT INVARIANT

5.1 Pre-Commit Requirement

SRS-002-SHALL-006

No state transition SHALL occur without a preceding `AX:TRANS:v1` commitment.

5.2 Ordering Constraint

SRS-002-SHALL-007

The transition sequence SHALL be:

```
determine transition
→ commit AX:TRANS:v1
→ mutate in-memory state
```

5.3 Commit Failure Behaviour

SRS-002-SHALL-008

If `AX:TRANS:v1` commitment fails:

- state mutation SHALL NOT occur
- agent SHALL transition to `STOPPED`

5A. SUBSTRATE FAILURE HANDLING

5A.1 Ledger Failure Override

SRS-002-SHALL-025

If the L6 substrate returns:

- `ledger_fail`
- `io_error`

the agent SHALL:

- immediately transition to `STOPPED`
- set local terminal state
- prohibit further mutation

5A.2 Fail-Safe Priority Rule

Safety SHALL take precedence over audit continuity.

5A.3 Invariant

Even if `AX:TRANS:v1` cannot be committed:

- agent MUST still enter `STOPPED`

---

6. TIME ORACLE MONOTONICITY

6.1 Admission Requirement

SRS-002-SHALL-009

All timestamps MUST be admitted as `AX:OBS:v1`.

6.2 Monotonicity Constraint

SRS-002-SHALL-010

For timestamps:

```
T_new > T_last MUST hold
```

6.3 Violation Behaviour

SRS-002-SHALL-011

If:

```
T_new ≤ T_last
```

then:

- violation SHALL be raised
- agent SHALL transition to `STOPPED`

---

7. INPUT ALPHABET

7.1 Admitted Input Classes

The agent SHALL support the following input classes:

Input Class	Description
`AX_INPUT_TIME_OBS`	Admitted timestamp observation
`AX_INPUT_LLM_OBS`	Admitted LLM response observation
`AX_INPUT_POLICY_TRIGGER`	Policy evaluation trigger
`AX_INPUT_FAULT_SIGNAL`	Fault condition signal
`AX_INPUT_RESET_REQUEST`	Reset/recovery request

7.2 Closure Requirement

SRS-002-SHALL-012

The input set SHALL be closed.

No undeclared input types are permitted.

7.3 Input Ordering Guarantee

SRS-002-SHALL-028

The agent SHALL process inputs strictly in ledger order:

```
ORDER BY ledger_sequence ASC
```

No alternative ordering source is permitted.

7A. UNKNOWN INPUT HANDLING

7A.1 Semantic No-Op Transition

SRS-002-SHALL-023

For any admitted input that is:

- syntactically valid (`AX:OBS:v1` committed)
- semantically irrelevant to the current state

the agent SHALL:

- produce a deterministic No-Op transition
- preserve current state
- commit an `AX:TRANS:v1` record

7A.2 No-Op Invariant

The No-Op transition SHALL satisfy:

```
State(t+1) = State(t)
```

while still producing:

- `AX:TRANS:v1` (No-Op witness)

8. TRANSITION TOTALITY

8.1 Total Function Requirement

SRS-002-SHALL-013

The transition function SHALL be total:

```
F : (State × InputClass) → (NewState × Evidence)
```

8.2 Completeness

SRS-002-SHALL-014

Every `(state, input_class)` pair MUST map to exactly one outcome.

8.3 No Hidden States

SRS-002-SHALL-015

No undeclared states SHALL exist.

8.4 Deterministic Outcome

SRS-002-SHALL-016

Given identical inputs and state, the transition result SHALL be identical.

8A. FAULT ACCUMULATION

8A.1 Fault Budget Requirement

SRS-002-SHALL-024

The agent SHALL maintain a deterministic fault accumulator.

8A.2 Budget Constraints

The accumulator SHALL be:

- fixed-width integer (uint32_t)
- zero-initialised
- deterministic across platforms

Thresholds SHALL be:

- hardcoded constants

- identical across all builds

8A.3 Transition Rules

Fault accumulation SHALL govern transitions:

Condition	Result
`fault_count < threshold_alarm`	Remain `ENABLED`
`fault_count ⩾ threshold_alarm`	Transition to `ALARM`
`fault_count ⩾ threshold_stop`	Transition to `STOPPED`

8A.4 Determinism Requirement

Fault accumulation SHALL be:

- order-dependent
- deterministic under identical input sequence

8A.5 Fault Accumulator Reset

SRS-002-SHALL-029

The fault accumulator SHALL:

- reset to zero on transition to `INIT`
- remain unchanged across `ENABLED` / `ALARM` / `DEGRADED`
- never decrease except via reset to `INIT`

---

9. TRANSITION TABLE (CLOSED SET)

9.1 Complete Transition Matrix

Current State	Input Class	New State	Evidence
`UNINIT`	`RESET_REQ`	`INIT`	Genesis binding witness
`UNINIT`	other	`STOPPED`	Invalid init violation
`INIT`	`TIME_OBS`	`ENABLED`	Temporal sync witness
`INIT`	`FAULT_SIGNAL`	`STOPPED`	Init failure witness
`INIT`	other	`INIT`	No-Op
`ENABLED`	`LLM_OBS`	`ENABLED`	Decision/action witness
`ENABLED`	`FAULT_SIGNAL`	`ALARM`	Fault threshold witness
`ENABLED`	`TIME_OBS`	`ENABLED`	Time progression witness
`ENABLED`	`POLICY_TRIGGER`	`ENABLED`	Policy evaluation witness
`ENABLED`	other	`ENABLED`	No-Op
`ALARM`	`POLICY_TRIGGER`	`DEGRADED`	Mitigation witness
`ALARM`	`FAULT_SIGNAL`	`STOPPED`	Critical failure witness
`ALARM`	other	`ALARM`	No-Op
`DEGRADED`	`RESET_REQ`	`INIT`	Recovery witness
`DEGRADED`	`FAULT_SIGNAL`	`STOPPED`	Escalation witness
`DEGRADED`	other	`DEGRADED`	No-Op
`STOPPED`	ANY	`STOPPED`	Terminality violation witness

9.2 Closure Guarantee

SRS-002-SHALL-017

Any transition not defined above SHALL:

- raise violation
- transition to `STOPPED`

The transition table SHALL be:

- complete
- closed
- deterministic

No undefined `(state, input)` pairs SHALL exist.

---

10. BOUNDEDNESS

10.1 Execution Bound

SRS-002-SHALL-018

Each transition SHALL:

- process exactly one input
- execute in constant or bounded time
- perform no recursion
- perform no unbounded iteration
- not allocate unbounded memory

---

11. REPLAY EQUIVALENCE

11.1 Replay Requirement

SRS-002-SHALL-019

Given:

- identical initial state

- identical ordered `AX:OBS:v1` sequence

the system SHALL reproduce:

- identical `AX:TRANS:v1` sequence

11.2 Scope

Replay equivalence SHALL apply to:

- state transitions
- health state progression
- violation behaviour

---

12. TRANSITION EVIDENCE

12.1 Evidence Canonicality

SRS-002-SHALL-027

Every `AX:TRANS:v1` record SHALL:

- be canonicalised per RFC 8785 (JCS)
- be bit-identical for identical transitions

12.2 Required Evidence Fields

Each `AX:TRANS:v1` SHALL include:

Field	Type	Description
`prev_state`	enum	State before transition
`input_class`	enum	Input that triggered transition
`next_state`	enum	State after transition
`violation`	enum/null	Violation type if any
`fault_count`	uint32	Fault accumulator value
`ledger_seq`	uint64	Ledger sequence number

12.3 Forbidden Fields

The following SHALL NOT appear in `AX:TRANS:v1`:

- wall-clock timestamps (unless admitted as `AX:OBS:v1`)
- random values
- process IDs or thread IDs
- memory addresses

---

13. VIOLATION HANDLING

13.1 Violation Types

The system SHALL define:

Violation	Description
`TIME_ROLLBACK`	Timestamp monotonicity violation
`POLICY_BREACH`	Policy constraint violation
`FAULT_BUDGET_EXCEEDED`	Fault threshold exceeded
`PROTOCOL_VIOLATION`	State machine protocol error
`GENESIS_MISMATCH`	Ledger binding failure
`COMMIT_FAILURE`	L6 commit failed

13.2 Violation Behaviour

SRS-002-SHALL-020

On violation:

If commit succeeds:

- violation SHALL be recorded in `AX:TRANS:v1`
- state SHALL transition deterministically
- if critical → `STOPPED`

If commit fails:

- agent SHALL still transition deterministically to `STOPPED`
- violation SHALL be marked in local state (non-persistent)
- `local_violation_flag` SHALL be set

---

14. RESET SEMANTICS

14.1 Reset Requirement

SRS-002-SHALL-021

Recovery from `STOPPED` SHALL require:

- explicit reset input
- new `AX:OBS:v1` admission

14.2 Reset Behaviour

Reset SHALL:

- reinitialise state
- preserve ledger history
- reset fault accumulator to zero

---

15. TRACEABILITY

15.1 Requirement Mapping

SRS-002-SHALL-022

Every transition SHALL be traceable to:

- an SRS requirement
- an `AX:TRANS:v1` record

16. PHASE 2 CLOSURE CRITERIA

Phase 2 is complete when:

16.1 Replay Verification

- replay produces identical transitions

16.2 Monotonicity Enforcement

- time rollback triggers `STOPPED`

16.3 Totality Proof

- all `(state, input)` pairs covered

16.4 Traceability

- all transitions linked to SRS

16.5 Fault Budget

- threshold transitions are deterministic

16.6 Genesis Binding

- agent lifecycle bound to ledger identity

16.7 Evidence Canonicality

- all `AX:TRANS:v1` records bit-identical for identical transitions

---

17. REQUIREMENT SUMMARY

ID	Requirement	Section
SRS-002-SHALL-001	Determinism definition	3.2
SRS-002-SHALL-002	Oracle boundary	3.3
SRS-002-SHALL-003	Terminal state	4.2
SRS-002-SHALL-004	Terminal behaviour	4.3
SRS-002-SHALL-005	Runtime fault coupling	4.4
SRS-002-SHALL-006	Pre-commit requirement	5.1
SRS-002-SHALL-007	Ordering constraint	5.2
SRS-002-SHALL-008	Commit failure behaviour	5.3
SRS-002-SHALL-009	Timestamp admission	6.1
SRS-002-SHALL-010	Monotonicity constraint	6.2
SRS-002-SHALL-011	Monotonicity violation	6.3
SRS-002-SHALL-012	Input closure	7.2
SRS-002-SHALL-013	Total function	8.1
SRS-002-SHALL-014	Completeness	8.2
SRS-002-SHALL-015	No hidden states	8.3
SRS-002-SHALL-016	Deterministic outcome	8.4
SRS-002-SHALL-017	Illegal transitions	9.2
SRS-002-SHALL-018	Execution bound	10.1
SRS-002-SHALL-019	Replay requirement	11.1
SRS-002-SHALL-020	Violation behaviour	13.2
SRS-002-SHALL-021	Reset requirement	14.1
SRS-002-SHALL-022	Traceability	15.1
SRS-002-SHALL-023	No-Op transition	7A.1

ID	Requirement	Section
SRS-002-SHALL-024	Fault accumulator	8A.1
SRS-002-SHALL-025	Substrate failure	5A.1
SRS-002-SHALL-026	Genesis binding	4A.1
SRS-002-SHALL-027	Evidence canonicality	12.1
SRS-002-SHALL-028	Input ordering	7.3
SRS-002-SHALL-029	Fault accumulator reset	8A.5

Total: 29 SHALL requirements

---

18. FINAL STATEMENT

The Axioma agent SHALL:

> Operate as a total, bounded, deterministic state machine whose behaviour is fully defined, replayable, and anchored to cryptographic evidence.

System Property:

> The agent cannot behave differently without producing different evidence.

---

19. REVISION HISTORY

Version	Date	Author	Changes
0.1-draft	2026-03-26	William Murray	Initial draft
0.2	2026-03-26	William Murray	Added SHALL-023 to SHALL-026, complete transition matrix
0.3	2026-03-26	William Murray	Added SHALL-027 (evidence canonicality), SHALL-028 (input ordering), SHALL-029 (fault reset), tightened violation semantics, collapsed boundedness section, added layer terminology

20. DOCUMENT APPROVAL

Role	Name	Date	Signature
Author	William Murray	2026-03-26	
Reviewer			
Approver			