

TMPA Implementation and Case Report

Zhu Wei - joinwell52 Research Center

2026-08-11 - I1.0 - TMPA V1.0

Contents

TMPA Implementation and Case Report	2
TMPA Core S1.0, FCoP, CodeFlowMu V1.8.0, and retained field evidence	2
Abstract	2
1. Scope and Research Questions	2
2. Architecture and Evidence Boundaries	3
3. Fixed Sources and Evidence Design	3
3.1 Locked identities	3
3.2 Fixed machine inputs	3
3.3 Evidence construction	4
4. Executed Test Baseline	4
5. C01–C14 Product Results	4
6. S0.6 to S1.0 Engineering Delta	5
6.1 Stable machine identity	5
6.2 Product-level projection	5
6.3 Regression alignment without criterion weakening	5
7. Regression and Reproducer Results	5
8. Retained WP-13 Evidence-Gating Case	5
9. Three-Valued Governance Interpretation	6
10. Evidence Integrity and Publication Audit	6
11. Limitations	6
12. Claim Ledger	7
13. Engineering Conclusion	7
Artifact Availability	7
References	8

TMPA Implementation and Case Report

TMPA Core S1.0, FCoP, CodeFlowMu V1.8.0, and retained field evidence

Document Version: I1.0

Status: Official implementation and case report

Normative Target: TMPA Core Specification S1.0, frozen candidate commit 942cbb097eb3d662662f96a2269818ec9d7ca2ed

Product Under Test: CodeFlowMu V1.8.0, evidence commit c1e1f724293e8048fc3a956b6f6df8cf83f54830

Evidence Capture: 2026-08-11, Asia/Shanghai

Formal Evidence Package: tmpa-i1.0-codeflowmu-v1.8.0-s1.0-evidence-20260811.zip

Package SHA-256: 9b92b06c3e46c8019362f55075be4ed066cb7a9c0d9859945b6c3b7de8840d04

Abstract

I1.0 evaluates CodeFlowMu V1.8.0 against all fourteen mandatory criteria in TMPA Core S1.0 using the exact bytes frozen by the normative candidate commit. The CodeFlowMu product path reports **14 PASS, 0 PARTIAL, 0 NOT RUN, and 0 FAIL**. The runner calls GovernanceReader.readSync; it does not call the TMPA Reference Reader. The input-bundle digest is sha256:f98764987760cdc8ac356b1265fc98485f33345e7d6ffc8575ccb059ddd34daa, and the result digest is sha256:0f0f642449db1853371861751a7a8ea36dce00013f53e32012a5e4dae45f4c39.

The evidence package fixes the S1.0 schemas, profiles, fixtures, product Adapter and Reader source, CodeFlowMu and FCoP revisions, dependency locks, raw command logs, pre-fix failures, final regression results, a reduced clean-machine reproducer, and a 889-entry SHA-256 manifest. The final regression record contains 24/24 TMPA Runtime tests, 1,522 passed / 0 failed / 1 skipped Runtime tests, 791/791 Shell tests, and 1,210 passed / 2 skipped FCoP reference-implementation tests.

The strongest supported conclusion is **author-run demonstrated product behavior for one exact implementation and one exact S1.0 bundle**. The result is not independent validation, third-party certification, universal conformance, proof of TMPA theory, proof of semantic truth, proof that hallucinations have been eliminated, or independent adoption.

1. Scope and Research Questions

I1.0 asks:

1. Does the CodeFlowMu V1.8.0 product Reader satisfy S1.0 C01–C14 against the exact frozen input bundle?
2. Can the source revision, product revision, input bundle, per-criterion results, regression runs, and evidence archive be traced without substituting the Reference Reader?
3. Does the evidence package preserve pre-fix results and demonstrate a clean reproducer without weakening the S1.0 criteria?
4. Which findings are demonstrated, and which conclusions remain unsupported?

The unit of judgment is a criterion-bound claim over a fixed bundle. Evidence maturity and conformance verdicts are reported separately:

Evidence level	Meaning in I1.0	Reached?
Specified	S1.0 clauses, schemas, profiles, and C01–C14 define the required behavior	Yes

Evidence level	Meaning in I1.0	Reached?
Implemented	CodeFlowMu V1.8.0 contains the corresponding Adapter, Reader, protocol, and governance mechanisms	Yes
Demonstrated	Author-run product and regression executions produced inspectable evidence	Yes, for the fixed bundle and revision
Independently Adopted	An unrelated organization adopted and independently validated the mechanism	No

2. Architecture and Evidence Boundaries

The current guidance and implementation relation is:

```

TMPA Architecture Paper A1.0
  ↓ provides the architecture theory and design direction
TMPA Core Specification S1.0
  ↓ fixes normative object, Reader, and conformance behavior
FCoP protocol
  ↓ supplies the file-based coordination and evidence profile
CodeFlowMu V1.8.0
  ↓ implements and consumes the Adapter/Reader result
bounded cases, governance gates, recovery, and audit views

```

TMPA is the theory and normative governance layer. FCoP is a reusable coordination protocol, not an application program. The `fcop` and `fcop-mcp` packages are reference implementations of FCoP rather than the protocol itself. CodeFlowMu is the engineering system guided by TMPA: it emits and consumes coordination evidence, projects FCoP artifacts into TMPA candidates, reconstructs a governance view, and lets workflow, review, recovery, and audit components consume the result.

Historical co-evolution is reported separately. XiaoDian AI, FCoP, CodeFlowMu, and TMPA share an author-controlled lineage; engineering feedback helped refine the present formalization. That lineage does not make an implementation a proof of the theory. WP-13 remains a bounded example of role-separated evidence admission and fact checking; it does not prove hallucination elimination and is not substituted for the S1.0 C01–C14 run.

3. Fixed Sources and Evidence Design

3.1 Locked identities

Item	Fixed identity	Role in this report
TMPA Core	S1.0 candidate commit 942cbb097eb3d662662f96a2269818ec9d7ca2ed	Normative input
CodeFlowMu	V1.8.0 evidence commit c1e1f724293e8048fc3a956b6f6df8cf83f54830	Product under test
CodeFlowMu protocol surface	V1.2.0	Product schema and validator surface
FCoP reference implementation	commit da79dfefd99f597c9e422ce9edec22157f915a21	Locked dependency regression only
Product Reader	GovernanceReader.readSync	Required execution path
Product Adapter	FcopSourceAdapter.projectFcopToTmpa	FCoP-to-TMPA projection path

The product run used an isolated, tracked-clean evidence worktree. The original CodeFlowMu mother worktree was observed as dirty and changing, so it was not used as the evidentiary execution root. The V1.8.0 evidence commit was local-only at capture time: it was not pushed, tagged, or released. The complete source snapshot and V1.7.0-to-V1.8.0 patch are therefore included in the evidence archive. This supports inspection of the tested source but does not turn the commit into a public CodeFlowMu release.

3.2 Fixed machine inputs

The run locks four S1.0 JSON Schemas plus the lifecycle Profile, canonicalization Profile, fixtures, and input bundle identity. The four published Schema hashes are:

Schema	SHA-256
Governance object	a2829cd7149c3054a52886365f2293a23106b636b0c52799739bfabdab1ff4fa
Lifecycle Profile	481a61ac2485bbaf15d90e9c5a255ad9ce6a55971190f0fe404856be4b10f993
Reader result	4527df7096fe840b85b245e50d5cea576ff359d50a54d17c8873a7b4f458d431
Conformance result	4b1ecef83e62d2aa1aff0e79a0cd0ea0a85fbc14a426d5fe873ab40aefdc2fe

3.3 Evidence construction

For each criterion, the product runner records a manifest, explicit mandatory assertions, product input invocations, canonical Reader output, assertion outcomes, a manifest digest, and a result digest. The aggregate result is validated against the S1.0 conformance-result Schema. Raw stdout, stderr, exit status, command, working directory, environment, dependency-lock digest, and remediation history are retained.

The frozen S1.0 candidate corpus also contains a historical product NOT RUN baseline. I1.0 does not rewrite it. The V1.8.0 product result is registered as a later external exact-version run so that candidate history and later evidence remain distinguishable.

4. Executed Test Baseline

The formal product command was `npm run test:tmpa:s1.0`. It executed the CodeFlowMu product Reader against the fixed bundle and produced the following aggregate:

```

TMPA Core: S1.0
Implementation: CodeFlowMu V1.8.0
Product Reader called: true
Reference Reader called: false
PASS: 14
PARTIAL: 0
NOT RUN: 0
FAIL: 0
Aggregate: PASS

```

The S1.0 Reference Reader separately reports 14/14 PASS. Its result validates the author-produced reference path; it is not counted as a CodeFlowMu product result.

5. C01–C14 Product Results

Criterion	Tested behavior	Mandatory assertions	Result
C01	Schema validation and rejection of invalid shapes	3	PASS
C02	Primary-carrier and single-writer immutability	5	PASS
C03	Duplicate identity handling with source provenance	5	PASS
C04	Per-stream continuity and asynchronous progress	4	PASS
C05	Role authority evaluation	5	PASS
C06	Lifecycle legality and state preservation	9	PASS
C07	Separation of duties and human approval authorization	10	PASS
C08	Integrity tampering detection	3	PASS
C09	Missing-reference treatment	4	PASS
C10	Prohibited-cycle detection	4	PASS
C11	Aggregation and reconstruction determinism	4	PASS
C12	Conflict preservation and explicit resolution	5	PASS
C13	Recovery behavior	5	PASS
C14	Terminal-history preservation	5	PASS

The 71 recorded assertions were recomputed during publication review. All manifest digests, actual-result digests, input-bundle digest, and the aggregate conformance-result digest matched the package records.

6. S0.6 to S1.0 Engineering Delta

6.1 Stable machine identity

S1.0 reissues the reviewed S0.6 behavior under stable S1.0 schema identifiers, Profile identities, canonicalization identity, and executable corpus paths. CodeFlowMu V1.8.0 binds its validator and Reader to those identities rather than treating an older-Core result as evidence for S1.0.

6.2 Product-level projection

The product runner imports the exact S1.0 bundle, validates the lifecycle Profile through the CodeFlowMu protocol surface, creates the CodeFlowMu GovernanceReader, and passes FCoP-derived source candidates through the product path. The Reference Reader module is retained in the bundle for traceability but is not imported or called by the product runner.

6.3 Regression alignment without criterion weakening

The retained pre-fix Runtime run reported 1,520 passed, 2 failed, and 1 skipped. The failures were stale expectations for V1.7 wording and for the distinction between QA execution completion and a failing business verdict. Tests were aligned to the already implemented contract; no S1.0 schema, fixture, mandatory assertion, Reader behavior, or pass criterion was weakened.

A later isolated wake-endpoint failure passed five immediate repetitions. One full run also stalled in `TaskDispatcher.test.ts`; the exact child process was terminated, and a bounded audit then reported 29 passed / 0 failed. These records remain in the package rather than being erased.

7. Regression and Reproducer Results

Surface	Final result	Interpretation
CodeFlowMu TMPA Runtime suite	24 passed / 0 failed	Product Reader unit and integration surface
CodeFlowMu Runtime full suite	1,522 passed / 0 failed / 1 skipped	Final full run
Runtime batched coverage	207/207 files; 1,522 passed / 0 failed / 1 skipped	Exact file coverage confirmation
CodeFlowMu Shell batched coverage	791 passed / 0 failed	Exact 8-batch execution
Protocol validation and typecheck	exit 0	Schema and validator surface
FCoP locked reference implementation	1,210 passed / 2 skipped	Dependency regression; not a substitute for product Reader
Reduced clean-machine reproducer	14/14 PASS	npm ci plus exact S1.0 product runner

The reproducer initially excluded an entire Protocol schemas directory and therefore also removed the required S1.0 schemas. That reduced-scope attempt is retained as a failure. The corrected reproducer excludes only unrelated legacy material, retains schemas/tmpa, installs from the lock file, and executes the same product runner successfully.

8. Retained WP-13 Evidence-Gating Case

WP-13 is retained as a field-oriented example of multi-agent fact checking with executor/reviewer separation, evidence admission, audit records, and explicit lifecycle boundaries. Its evidence package supports the bounded conclusion that development completed and role-separated QA passed for the captured task state. It also records that later task snapshots were still review and pending, that a scheduled date was advanced without a separately located approval artifact, and that runtime binding and signed checksums were outside the package boundary.

WP-13 illustrates why TMPA separates execution evidence, review evidence, authorization, lifecycle state, and publication claims. It is not one of the S1.0 C01–C14 product fixtures. It does not prove the theory, independently validate CodeFlowMu, or prove that multi-agent systems cannot hallucinate.

9. Three-Valued Governance Interpretation

TMPA keeps semantic judgment separate from view classification:

Judgment	Typical view	Meaning
valid	authoritative	Required evidence and applicable rules establish the conclusion.
invalid	quarantined / rejected	A deterministic violation excludes the affected evidence or action from authoritative use.
undetermined	partial / disputed / pending_human	Evidence is missing or conflicting, or an authorized human decision is still required.

The V1.8.0 run makes this separation observable. A wrong-type, self-issued, or otherwise unauthorized approval remains preserved but cannot satisfy C07. A missing reference in C09 leaves the dependent claim undetermined rather than silently complete. An unauthorized resolution in C12 remains evidence but is invalid as a resolving act. Integrity failure in C08 quarantines the covered content while preserving its source record. These are governance judgments over evidence; they are not semantic truth judgments about the world.

10. Evidence Integrity and Publication Audit

The formal archive contains 891 entries and 889 files. All 889 files are covered by its internal SHA-256 manifest, and the outer ZIP SHA-256 is 9b92b06c3e46c8019362f55075be4ed066cb7a9c0d9859945b6c3b7de8840d04. Publication audit verified:

- safe ZIP paths and structural integrity;
- 889/889 internal SHA-256 entries;
- strict UTF-8 decoding for 884 text files;
- successful parsing of all 190 JSON files;
- byte identity for the four published S1.0 Schemas and the remaining normative inputs;
- product Reader invocation without Reference Reader substitution;
- Schema-valid C01–C14 result envelopes;
- recomputation of all fourteen manifest digests, fourteen result digests, 71 mandatory assertions, the aggregate result digest, and the input-bundle digest;
- preservation of pre-fix failures, remediation notes, raw commands, exit status, dependency locks, source snapshot, and patch.

No recomputed digest or assertion differed from the archive record. This audit establishes internal consistency and traceability of the submitted package; it is not an independent product rerun or certification.

11. Limitations

1. The product and regression evidence is author-run. No independent organization has certified or adopted the implementation.
2. The CodeFlowMu evidence commit was local-only at capture time and was not a public tag or release. The archive carries a complete source snapshot and patch for inspection.
3. The evidence worktree was tracked-clean, but the original mother worktree was dirty and changing. Claims are bound to the isolated evidence worktree and fixed commit.

4. The reduced reproducer demonstrates the conformance slice, not every private deployment dependency or operational environment of CodeFlowMu.
5. Runtime retains one skipped test; the FCoP reference implementation retains two skipped tests. Neither skip is counted as a C01–C14 product result.
6. C11 evaluates a fixed fixture set and declared permutations; it is not a formal proof over arbitrary graphs, encodings, filesystems, or hostile platforms.
7. C08 demonstrates governance-object integrity handling, not model truthfulness, actor authentication, installer integrity, or Byzantine resilience.
8. Full-suite performance, representative SME burden, comparative baselines, cross-profile portability, and independent deployment remain open empirical questions.
9. WP-13 is a bounded governance and evidence-admission case, not a hallucination-elimination benchmark.

12. Claim Ledger

Claim	I1.0 disposition
TPMA Core S1.0 defines C01–C14	Specified
CodeFlowMu V1.8.0 contains corresponding product mechanisms	Implemented
The exact product bundle records 14/14	Demonstrated
PASS	
The archive preserves inputs, source, commands, outputs, failures, and hashes	Demonstrated
The reduced conformance slice ran successfully in the captured clean reproducer	Demonstrated
CodeFlowMu is universally conformant for arbitrary inputs and deployments	Not claimed
The result has been independently rerun, certified, or adopted	Not demonstrated
TPMA theory is proved by the implementation	Prohibited conclusion
WP-13 proves hallucination elimination	Prohibited conclusion

13. Engineering Conclusion

I1.0 establishes a release-grade, exact-input engineering baseline for TPMA Core S1.0. CodeFlowMu V1.8.0 passes all fourteen mandatory criteria through its own product Adapter and Reader path, records 71 mandatory assertions, and preserves the regression, remediation, source, dependency, and integrity trail needed to inspect the result. The S1.0 frozen candidate baseline and the later product execution remain separately identifiable.

The result strengthens evidence that TPMA can guide a concrete engineering system. It does not make CodeFlowMu the authority that defines TPMA, and it does not convert engineering success into proof of the theory. The dependency direction remains: A1.0 states the architecture theory; S1.0 defines normative behavior; FCoP supplies the coordination protocol; CodeFlowMu implements and consumes the governance projection; I1.0 reports the bounded evidence.

Artifact Availability

The formal archive is `tmpa-i1.0-codeflowmu-v1.8.0-s1.0-evidence-20260811.zip`. The adjacent file `tmpa-i1.0-codeflowmu-v1.8.0-s1.0-evidence-20260811.zip.sha256` records `9b92b06c3e46c8019362f55075be4ed066cb7a9c0d9859945b6c3b7de8840d04`.

The run is registered in the S1.0 external-run registry. Earlier I0.6–I0.8 packages remain immutable historical evidence at their versioned paths. Git history is the publication history; no parallel paper database has editorial authority.

References

- [1] TMPA Project. “TMPA Core Specification S1.0,” frozen candidate commit 942cbb097eb3d662662f96a2269818ec9d7ca2ed. GitHub, 2026.
- [2] TMPA Project. “TMPA Architecture Paper A1.0.” GitHub, 2026.
- [3] FCoP Project. “FCoP — File-based Coordination Protocol,” reference implementation commit da79dfefd99f597c9e422ce9edec22157f915a21. GitHub, 2026.
- [4] CodeFlowMu Project. “CodeFlowMu V1.8.0 S1.0 Product Conformance,” evidence commit c1e1f724293e8048fc3a956b6f6df8cf83f54830, 2026.
- [5] TMPA Project. “I1.0 CodeFlowMu V1.8.0 S1.0 Evidence,” package tmpa-i1.0-codeflowmu-v1.8.0-s1.0-evidence-20260811, 2026.
- [6] CodeFlowMu Project. “WP-13 Multi-Agent Fact-Check Publication Evidence V3,” 2026.
- [7] CodeFlowMu Project. “TMPA Governance: Theory-to-Engineering Relation,” docs/ TMPA-GOVERNANCE.md. GitHub, 2026. <https://github.com/joinwell52-AI/codeflowmu/blob/main/docs/TMPA-GOVERNANCE.md>.