GENERAL REASONING, INC.

STANDARD

Agent-Native Development Maturity

ANDM v1.0

A conformance standard for organizations building software with AI agents as primary executors. Establishes the infrastructure requirements, artifact governance model, and audit architecture that make CMMI Level 5 process maturity economically viable for the first time outside mission-critical government contexts.

Issued by

General Reasoning, Inc.

Birmingham, Alabama · genreason.com

2026

DXMachine is the reference implementation of this standard. Chandra is the audit substrate on which ANDM conformance rests.

BACKGROUND

The problem CMMI Level 5 solved — and why it failed

The Capability Maturity Model Integration (CMMI) Level 5 defines the highest degree of process maturity in software development: quantitatively managed, continuously optimized, causally analyzed. It is not a description of good intentions. It is a structural guarantee — the organization can measure its own performance, trace defects to root causes, and demonstrably improve over time. It is the answer to the question regulators, auditors, and mission-critical buyers have always wanted to ask: not whether your software works today, but whether your process guarantees it will keep working.

CMMI Level 5 failed in commercial software development for one reason: the governance overhead was the product. To operate at Level 5 with human teams required process archaeologists, measurement specialists, review boards, and documentation infrastructure that consumed a significant fraction of the engineering capacity it was governing. The only organizations that could absorb this cost were those for whom the alternative — a failed mission — was measured in lives and billions. NASA Mission Control operated at Level 5. Commercial software settled at Level 2 or 3 in practice, called Level 5 academic, and moved on. The market punished governance latency before the quality dividend arrived.

> **The industry did not reject CMMI Level 5 because the goal was wrong. It rejected it because the human execution cost made it irrational at commercial velocity. That constraint no longer exists.**

THE FORCING FUNCTION

What agents change

AI agents collapse the governance overhead cost toward zero. The comprehension gate a senior engineer would spend days applying — reading generated code, verifying architectural intent, cross-referencing dependencies — now takes minutes. The measurement infrastructure, the artifact generation, the audit trail, the causal analysis: these are precisely the class of work agents perform as a byproduct of the development operation itself. The governance overhead that made Level 5 uneconomical for human teams is the overhead agents eliminate.

But agents introduce a new risk that CMMI Level 5 was not designed to address: dark code. Code generated by AI, passing automated checks, shipping without the comprehension step ever occurring. Not buggy code. Not spaghetti code. Code that was never understood by any human at any point in its lifecycle — because the process no longer required it. Dark code is not a tooling problem or a security problem. It is an organizational capability problem. And it is compounding.

The forcing function is therefore bilateral. Agents make Level 5 maturity economically viable for the first time. Agents simultaneously make the consequences of failing to achieve it more severe than ever. An organization that deploys agents without Level 5 infrastructure is not moving fast. It is accumulating dark code at agent velocity — a liability that grows faster than any human team could previously generate.

This standard defines the infrastructure requirements that resolve the tension: maximum agent velocity, zero dark code, full organizational reconstructibility.

THE STANDARD

Agent-Native Development Maturity defined

Agent-Native Development Maturity (ANDM) is a conformance standard for organizations in which AI agents function as primary code executors. It establishes the minimum infrastructure, artifact governance, and audit architecture required to achieve CMMI Level 5 process maturity at agent velocity. ANDM does not modify CMMI's requirements. It provides the first viable implementation path for organizations that could not previously afford the human overhead Level 5 demanded.

Three invariants

Invariant	Statement
Dark factory, not dark code	The organization's internal development process may be opaque to external parties. The code produced by that process must not be opaque to the organization itself. Every artifact must be comprehensible, traceable, and reconstructible from its genesis record.
Auditability is authorization	The audit record is not a receipt applied after work completes. It is the gate token that authorizes the next unit of work. No governed operation proceeds without a preceding attestable record. This is the mechanism, not a policy.
Reconstruction over recovery	The goal is not backup. It is the ability to reconstruct the organization's complete digital state — and the decision lineage that produced it — from first principles. Any catastrophic event that destroys artifacts must leave the reconstruction roadmap intact.

FOUNDATIONAL PRINCIPLE

Rigor and velocity are not in tension

Niklaus Wirth's design philosophy, embodied in Modula-2 and its successors, held that the separation of interface from implementation, explicit dependency declaration, and formal module boundaries were not constraints on programmer productivity — they were the source of it. The industry rejected this thesis not because it was wrong, but because human execution speed made the overhead of rigorous structure feel like friction. Wirth was right. The execution environment was the limiting factor, not the principle.

Agents remove the execution bottleneck. The ANDM standard is Wirth's thesis applied at organizational scale: explicit interface declarations, separation of specification from implementation, immutable dependency graphs, and formal change records — all generated and maintained by agents as a byproduct of the work itself. The spec spoke is the definition module. The code spoke is the implementation module. The commit record is the formal change manifest. The architecture is forty years old. The execution environment that makes it economical is new.

THE AUDIT SUBSTRATE

Chandra: the organizational reconstruction protocol

ANDM conformance requires an append-only, cryptographically sealed audit substrate. Chandra is the reference implementation. A conformant substrate must satisfy the following properties:

Property	Requirement
Append-only	No CU may be modified or deleted after publication. Corrections are new appends whose content is the corrected state. The error remains in the chain permanently.
Cryptographic sealing	Each CU carries a hash of its canonical content plus the hash of the preceding CU. Chain integrity is verifiable at any point without a trusted third party.
Attribution invariant	Every CU must carry a human author identifier. Agent-authored CUs must identify both the agent and the human principal who authorized the operation.
Agent-readable primary	The substrate's native audience is agents. Header fields are indexed for O(1) agent lookup. Payload content is agent-parseable without human mediation.
Reconstruction completeness	The substrate must be sufficient to reconstruct the organization's complete governed artifact set from genesis. The chain IS the roadmap.

ARCHITECTURE

Domain / Hub / Spoke / Commit Record

The ANDM artifact governance model organizes all governed artifacts into a four-tier hierarchy. Each tier has a defined scope, a manifest, and a commit record chain. The commit record chain at each tier is a structural primitive — auto-created, non-deletable, and not user-configurable.

Tier	Scope	Manifest	Commit record	User-created
Domain	A coherent organizational unit — a reconstructible package that could be handed to a recovery team intact.	Purpose, hub inventory, owner, genesis date.	Domain commit record chain. One entry per batch that touched any hub within the domain. References hub commit record ids.	Yes
Hub	A governed artifact cluster — one logical subject with multiple artifact dimensions (code, spec, tests, etc.).	Hub type, domain membership (immutable), spoke inventory, Level 5 conformance status.	Hub commit record chain. One entry per batch operation. References spoke CU ids. cu-correlation-id on spoke CUs points back to the hub commit record.	Yes

Spoke	A single artifact dimension within a hub. One append chain per spoke. The subject_id is the spoke identifier.	Spoke type, hub membership, creation date.	Each CU on the spoke IS the commit record for that artifact version. cu-correlation-id links upward to the hub commit record.	Yes (within Level 5 c onstraints)
Commit record	The governance layer. Not a spoke. Not a hub. A parallel structural primitive that runs vertically through all tiers.	N/A — the commit record IS the manifest of what changed.	The commit record chain at each tier is the auditable history of that tier's evolution. Agents navigate up and down via id references in each record's payload.	No — aut o-created

Commit record payload schema

Commit records at hub and domain level carry a fixed JSON manifest. The schema is fixed — agents read it without negotiation:

Field	Type	Hub commit record	Domain commit record
hub_id / domain_id	string	Hub identifier	Domain identifier
batch_id	string	Unique batch identifier	Unique batch identifier
timestamp	ISO 8601	Operation timestamp	Operation timestamp
actor	string	Human author + agent id	Human author + agent id
operations	array	Spoke-level ops: spoke_subject_id, spoke_type, op (created/updated/soft-deleted), cu_id	Hub-level ops: hub_id, op, hub_commit_record_id

> *Domain membership of a hub is declared at creation and immutable thereafter. A hub belongs to exactly one domain for its lifetime. This is a hard architectural constraint, not a policy. Without it the domain commit record cannot be written atomically with the hub commit record, and the domain-level audit guarantee collapses.*

The Level 5 Module Hub Standard

A module hub is a hub whose subject is a software module — a discrete, deployable unit of code with a defined interface and explicit dependencies. The Level 5 Module Hub Standard defines the minimum spoke configuration required for a module hub to be ANDM-conformant. This is the anti-dark-code guarantee made structural.

Spoke	Type token	Origin	Required	Conformance requirement
Commit record	"commit-record"	Auto-created with hub	Always	Structural primitive. Not governed by this standard. Exists at hub tier independent of content spokes.
Code	"code"	Level 5 standard	Yes	Canonical, versioned source. The authoritative implementation artifact. Must be co-versioned with the spec spoke — a code update without a corresponding spec update is a conformance violation.
Spec	"spec"	Level 5 standard	Yes	Markdown. Must be sufficient to reconstruct the module without the code spoke. Required content: purpose statement, public interface, all defparameters and exported functions with contracts, data store interactions, JS output description (if applicable), known invariants, failure modes. This is the Wirth definition module. Not a README.
Test results	"test-results"	Level 5 standard	Yes	Structured output from the testing framework against the current code spoke. Linked to spec assertions that generated the tests. The spec is the eval source. Stale test results (results that postdate a code update) are a conformance violation.
Additional spokes	any	Domain discretion	No	Enumerated in hub manifest. Examples: dependency graph, usage graph, agent attestation log, performance telemetry, changelog.

The spec spoke requirement

The spec spoke is the most critical and most demanding requirement in this standard. The test is simple and binary: if the code spoke were deleted, could an agent regenerate it from the spec spoke alone? If the answer is no, the spec is not conformant.

This is not documentation best practice. It is the organizational reconstruction guarantee made concrete at the module level. An organization whose every module hub has a conformant spec spoke can lose its entire codebase and reconstruct it. An organization without conformant spec spokes cannot make that claim regardless of what other backup infrastructure it has.

> *The spec spoke is co-versioned with the code spoke. They are not independent documents. A batch operation that updates the code spoke must update the spec spoke in the same commit record or the hub is in a non-conformant state.*

The spec-as-eval flywheel

The spec spoke is also the source of truth for the testing framework. Behavioral assertions in the spec become test cases. Test results in the test-results spoke are linked to the spec assertions that generated them. This closes the loop: spec -> code -> tests -> results -> spec update. The flywheel continuously improves both the artifact and the specification without additional governance overhead. This is the CMMI Level 5 continuous optimization loop operating at agent velocity.

ANDM to CMMI Level 5

ANDM does not extend CMMI Level 5. It provides the infrastructure through which an agent-native organization satisfies CMMI Level 5 requirements — and in several areas exceeds them, because the agent execution model enables continuous audit guarantees that human-mediated processes could only approximate.

CMMI Level 5 Practice Area	CMMI Requirement (summary)	ANDM Satisfaction
Causal Analysis and Resolution (CAR)	Identify causes of selected outcomes and take action to prevent recurrence.	The commit record hierarchy at domain, hub, and spoke level provides complete causal lineage for every artifact change. Root cause identification is a read operation on the chain, not a reconstruction exercise. Agent-mediated causal analysis runs against the chain without human review overhead.
Organizational Performance Management (OPM)	Proactively manage organizational performance to meet business objectives.	Domain commit records provide a complete timeline of organizational-level artifact motion. Performance metrics derive from the chain without instrumentation overhead. The domain manifest declares the organizational unit's objectives; the commit record chain measures progress against them.
Quantitative Project Management (QPM)	Quantitatively manage the project to achieve quality and process performance objectives.	Hub commit records provide per-module change frequency, defect introduction rate (via test-results spoke), and recovery time metrics. All quantitative management data is a query against the Chandra substrate — no separate measurement infrastructure required.
Continuous Process Improvement	Continuously improve processes based on quantitative understanding of common causes.	The spec-as-eval flywheel is the implementation of this practice area at agent velocity. Spec updates driven by test results, test results driven by spec assertions, code driven by both — the loop runs without human mediation at each cycle.
Configuration Management (CM)	Establish and maintain integrity of work products using configuration management.	Cryptographic sealing of the spoke chain provides integrity guarantees stronger than any human-mediated CM process. Every artifact version is immutably sealed. No configuration item can be modified without producing a new chain entry. Rollback is a new append, not a rewrite — the error and its correction are both auditable evidence.

Process and Product Quality Assurance (PPQA)	Provide staff and management with objective insight into processes and work products.	The Level 5 Module Hub Standard provides the structural definition of a conformant work product. Conformance is a query, not an audit exercise. The hub manifest carries conformance status. Non-conformant hubs are immediately visible without review board intervention.

Where ANDM exceeds CMMI Level 5: the traditional SOX and CMMI audit model assumes periodic snapshot reviews of a process that runs between reviews. In a continuous human-agent pipeline, the audit record IS the process. There is no gap between the event and the evidence. ANDM's commit record hierarchy provides a stronger compliance position than any snapshot-based audit model — not as a policy, but as an architectural guarantee.

CHANDRA CORE ENTRY

ANDM standard registration

The following constitutes the normative Chandra core entry for the ANDM standard. This entry lives in the core domain of any Chandra instance implementing ANDM. It is itself subject to the standard — the entry is a module hub with code, spec, and test-results spokes.

Field	Value
`Standard identifier`	ANDM-v1.0
`Domain`	core
`Hub type`	standard-definition
`Issued by`	General Reasoning, Inc.
`Effective date`	2026
`Supersedes`	None (inaugural version)
`CMMI mapping version`	CMMI-DEV v2.0
`Reference implementation`	DXMachine (genreason.com)
`Audit substrate reference`	Chandra MVP whitepaper (gr-chandra-mvp.pdf)
`Level 5 hub conformance check`	Hub manifest carries andm-conformant: true/false. Evaluated on every commit record write. Non-conformant state is a blocking condition in ANDM-enforcing implementations.

CONCLUSION

The day CMMI Level 5 was waiting for

CMMI Level 5 was always the right answer. The industry was not wrong to want quantitatively managed, continuously optimized, causally analyzed software development. It was wrong to think human teams could sustain it at commercial velocity. That constraint is gone.

An organization that deploys agents without ANDM-conformant infrastructure is not moving fast. It is accumulating dark code at agent velocity — organizational liability growing faster than any human team could previously generate it. The governance overhead that made Level 5 irrational for human teams is the overhead agents perform as a byproduct of the work itself.

The dark factory is now viable. The dark code is now inexcusable. The reconstruction guarantee is now achievable. This standard defines what achieving it requires.

Auditable. Haltable. Accountable.

General Reasoning, Inc. · Birmingham, Alabama · genreason.com