Human–AI Governed Engineering (HAGE)

Abhinav Tripathi

Abstract—**Software engineering methods such as Agile, DevOps, DevSecOps, and capability-maturity models were established in environments where accountable humans performed most engineering reasoning and deterministic automation executed predefined tasks. Generative AI and agentic AI change this operating assumption. They can interpret intent, create architecture and code, invoke tools, test systems, modify repositories, prepare deployments, and participate in operations. Yet many enterprises continue to insert these capabilities into the same human-centric lifecycle, preserving old queues, ceremonies, responsibility models, and delivery timelines. This paper proposes Human–AI Governed Engineering (HAGE), a governance-centric methodology for allocating work, authority, evidence, and accountability across human participants, generative models, autonomous agents, deterministic automation, and assurance mechanisms. HAGE introduces intent contracts, qualified context, risk-based autonomy, evidence packages, trust gates, continuous operational learning, and a five-level enterprise maturity model. It does not replace Agile or DevOps; it provides an execution and governance layer that allows those approaches to operate safely and efficiently when non-human participants perform material lifecycle responsibilities. HAGE is presented as a research framework requiring empirical validation. Its claimed contribution is not the first use of AI agents in software development, but an integrated enterprise methodology that connects end-to-end responsibility allocation, bounded autonomy, verification, accountable approval, operational learning, and measurable adoption.**

Index Terms—**Agentic AI, AI governance, evidence-based engineering, Generative AI, human–AI collaboration, software engineering methodology.**

I. EXECUTIVE SUMMARY

Enterprises have adopted GenAI assistants and coding agents, but adoption frequently remains additive rather than transformative. AI accelerates isolated activities while work still waits for conventional handoffs, sprint boundaries, reviews, environment queues, and governance approvals. This creates an "acceleration without redesign" problem: artifact generation becomes faster, but trusted delivery does not improve proportionally.

HAGE addresses this gap by redesigning the engineering operating model around five questions:

- What is the intended outcome and what constraints must never be violated?
- Which participant—human, AI, agent, or deterministic system—should perform each activity?
- How much autonomy is justified by risk, evidence, reversibility, and blast radius?
- What proof is required before an AI-produced change can advance?
- Who remains accountable for the decision and the production outcome?

A. The central proposition

Software engineering must evolve from human-centric collaboration with tool assistance to governed collaboration among human and non-human engineering participants.

B. HAGE at a glance

Dimension	Traditional assumption	HAGE position
Primary actor	Human team	Humans, AI models, agents, automation, and assurance systems

Unit of work	Story, task, code change	Intent contract plus required evidence
Control mechanism	Process compliance and peer review	Risk-based autonomy and evidence gates
Completion	Definition of Done	Definition of Trusted
Productivity	Velocity and throughput	Trusted throughput and verification-adjusted lead time
Accountability	Human role assignment	Human accountability with traceable non-human contribution
Learning	Retrospective and process improvement	Operational feedback updates context, evaluations, policies, and autonomy

II. MOTIVATION AND PROBLEM DEFINITION

A. The maturity paradox

A highly mature organization can possess optimized, measured, and continuously improved processes while still operating a lifecycle designed around human production capacity. CMMI emphasizes capability, performance, quality, predictability, and process improvement. Those strengths remain valuable, but maturity in an existing process does not by itself redesign responsibility when a new class of participant can perform substantial engineering work. HAGE calls this the maturity paradox: an organization may be excellent at executing yesterday's operating model while under-realizing the benefits of today's engineering capabilities.

B. Acceleration without lifecycle redesign

When GenAI is added only as a coding or documentation tool, local tasks become faster but overall lead time remains constrained by unchanged handoffs. The organization may generate more artifacts without increasing verified delivery. Review queues, context collection, architecture approval, security assessment, test execution, environment access, and release governance become the new bottlenecks.

C. Changed engineering assumptions

- AI output is probabilistic and may be plausible but wrong.
- Agents can act, not merely advise; they may invoke tools and alter state.
- AI can produce dependent artifacts faster than humans can manually inspect them.
- Prompt, context, model, tools, and policy jointly influence the result.
- A model cannot carry legal, regulatory, fiduciary, or organizational accountability.
- The cost of generation may fall while the cost of verification, attention, and governance rises.

III. RELATIONSHIP TO EXISTING METHODS AND EMERGING RESEARCH

HAGE is deliberately positioned as an extension layer rather than a declaration that Agile, DevOps, or CMMI are obsolete. The Agile Manifesto prioritizes individuals and interactions, working software, customer collaboration, and responsiveness to change. DevOps integrates development and operations through automation and feedback. CMMI develops measurable organizational capability and process performance. These foundations continue to matter.

Emerging research already recognizes agent-centric software engineering. Recent proposals describe Agentic SDLC architectures, agent-based development methodologies, verification-first lifecycles, and graduated human oversight. Therefore, HAGE does not claim to be the first agent-oriented software-development concept. Its intended contribution is the integration of enterprise lifecycle governance with explicit responsibility allocation, risk-calibrated autonomy, evidence-based progression, accountable human authority, operational feedback, and maturity measurement.

Approach	Primary focus	Non-human roles	Governance depth	HAGE relationship
Agile/Scrum	Human collaboration and iterative delivery	Tools are secondary	Team/process governance	HAGE can operate inside or across Agile increments
DevOps/DevSecOps	Automated delivery and operational feedback	Deterministic automation	Pipeline, security, operations	HAGE adds probabilistic actors and autonomy control
CMMI	Capability and process maturity	Not central to role model	Institutionalized process governance	HAGE can become a measurable capability domain
Agentic SDLC / Agentsway	Agents as software-development collaborators	Explicit AI-agent roles	Agent workflow and oversight	HAGE broadens to enterprise authority, evidence, risk, and operations
AI risk frameworks	Trustworthy AI risk management	AI system is governed object	Risk governance	HAGE applies risk governance to engineering execution

IV. HAGE DEFINITIONS AND SCOPE

A. Definition

Human–AI Governed Engineering (HAGE) is a software-engineering methodology that assigns activities, authority, constraints, evidence obligations, and accountability across human and non-human participants according to risk, capability, and operational context.

B. Scope

HAGE applies from initiative conception through architecture, construction, verification, deployment, operation, support, and retirement. It can govern product development, modernization, migration, maintenance, incident response, platform engineering, and regulated technology delivery.

C. Non-goals

- HAGE is not a specific AI model, agent framework, coding assistant, or orchestration product.
- HAGE does not permit accountability to be transferred to an AI system.
- HAGE does not assume that every activity should be automated.
- HAGE does not eliminate Agile ceremonies or DevOps controls where they remain useful.
- HAGE is not yet a certified standard or empirically proven universal method.

V. HAGE PRINCIPLES

1. Human accountability remains identifiable. Every material decision and production outcome must map to an accountable human authority.
2. Autonomy is earned, bounded, and reversible. AI authority increases only with evidence, limited permissions, observability, and rollback capability.
3. Intent precedes execution. Business outcome, constraints, prohibited actions, and acceptance evidence must be explicit before delegated execution.
4. Context is governed engineering infrastructure. Context sources require ownership, access control, freshness, provenance, and quality management.
5. Verification precedes trust. Fluency or confidence is not proof. Advancement requires independent evidence proportionate to risk.
6. Evidence travels with the change. Every material AI contribution must carry provenance, tests, policy results, assumptions, and unresolved uncertainty.
7. Least privilege applies to agents. Agents receive only the tools, data, environments, duration, and actions required for the assigned intent.

8. Generation and approval remain separated. The same mechanism should not be the sole generator, verifier, and approver of a high-risk change.
9. Human attention is a constrained resource. The methodology optimizes review quality and exception handling, not merely artifact volume.
10. Production outcomes govern future autonomy. Defects, incidents, overrides, and successful execution update evaluations, policies, and autonomy thresholds.
11. Risk determines process depth. Controls are proportionate to impact, uncertainty, reversibility, data sensitivity, and regulatory exposure.
12. Trusted outcomes outrank implementation effort. Performance is measured by verified value and operational success, not lines of code or raw AI usage.

VI. ENGINEERING PARTICIPANTS AND ACCOUNTABILITY

Participant	Primary responsibility
Human authority	Owns business, architectural, security, compliance, or operational decisions and accepts accountability.
Human practitioner	Performs engineering work, reviews evidence, resolves ambiguity, and intervenes when required.
Generative AI participant	Produces analysis or artifacts but does not independently alter governed environments unless wrapped in an agent.
Agentic participant	Plans and performs multi-step actions through approved tools within bounded authority.
Deterministic automation	Executes predefined repeatable controls, builds, tests, scans, policies, and deployments.
Assurance participant	Independently challenges outputs through tests, formal rules, simulations, reviews, or separate models.
Governance authority	Defines policies, risk classes, approval rules, audit requirements, and autonomy boundaries.

A.
HAGE responsibility notation

HAGE extends conventional RACI by distinguishing performance, authority, assurance, and evidence. A working notation is P-A-V-E:

- P — Performer: executes the activity.
- A — Accountable Authority: owns the decision and outcome.
- V — Verifier: independently validates the result.
- E — Evidence Custodian: ensures traceability and retained proof.

A non-human participant may be P or contribute to V, but an accountable authority must be an identified human role or legally accountable organizational body.

VII. THE HAGE LIFECYCLE

Phase	Purpose	Primary artifact
1. Intent Engineering	Convert the initiative into an intent contract containing outcome, scope, constraints, non-functional requirements, prohibited actions, acceptance evidence, and accountable authority.	Intent Contract
2. Context Qualification	Identify and approve the repositories, standards, policies, data, historical decisions, and	Qualified Context Pack

	knowledge sources that may influence execution.	
3. Risk and Responsibility Allocation	Classify impact and assign P-A-V-E roles for each activity.	Risk–Responsibility Map
4. Autonomy Design	Set the allowed tools, permissions, environment, time, cost, action boundaries, escalation triggers, and rollback rules.	Autonomy Contract
5. Governed Execution	Humans, models, agents, and automation perform assigned work while retaining prompts, model versions, tool calls, outputs, and decisions.	Execution Trace
6. Evidence Engineering	Generate proof through compilation, tests, static analysis, security scans, policy checks, simulations, traceability, and independent critique.	Evidence Package
7. Trust Decision	An authorized gate evaluates evidence, residual uncertainty, risk, and reversibility before progression.	Trust Decision Record
8. Controlled Release	Release through approved progressive-delivery, rollback, monitoring, and separation-of-duty controls.	Release Assurance Record
9. Operational Learning	Connect production outcomes to the responsible intent, context, AI contributions, controls, and approvals.	Outcome Learning Record
10. Knowledge and Autonomy Evolution	Update context, evaluation sets, prompts, policies, tests, and autonomy thresholds using validated outcomes.	Governance Change Record

VIII. RISK-BASED AUTONOMY MODEL

HAGE rejects one universal human-in-the-loop rule. Oversight must be calibrated to risk and demonstrated capability. The following provisional levels describe increasing delegated authority.

Level	Operating mode	Agent authority	Human control	Typical example
A0	Human-only	No AI participation	Human performs and approves	Restricted policy decision
A1	Assistive	Suggest or draft	Human executes and approves	Requirement summary
A2	Supervised execution	Act in sandbox or branch	Human reviews before merge or promotion	Code and test generation

A3	Conditional autonomy	Execute and progress after automated gates	Human approves defined high-impact gates	Low-risk service change
A4	Monitored autonomy	Operate within policy and rollback envelope	Human monitors exceptions and audits	Routine dependency remediation
A5	Adaptive bounded autonomy	Adjust plans within approved objectives	Human governs policy, thresholds, and exceptions	Mature internal platform optimization

A. Autonomy assignment factors

- Impact and blast radius
- Reversibility and rollback quality
- Data sensitivity and regulatory exposure
- Novelty and ambiguity of the task
- Evidence coverage and historical performance
- Agent permissions and tool reach
- Observability and incident-detection capability
- Human availability and escalation latency

IX. DEFINITION OF TRUSTED

HAGE extends the Definition of Done into a Definition of Trusted. A change is trusted only when the organization possesses sufficient evidence to justify its advancement at the assigned risk level.

- Intent traceability is complete.
- Source context and model/tool provenance are retained.
- Required deterministic tests and policy checks pass.
- Security and privacy obligations are satisfied.
- Known assumptions and residual uncertainty are visible.
- Approval is issued by the designated accountable authority.
- Rollback and operational monitoring are ready.
- Post-release learning linkage is established.

X. HAGE OPERATING CADENCE AND CEREMONIES

Ceremony	Purpose
Intent Review	Clarify outcomes, constraints, acceptance evidence, and accountable authority.
Context Readiness Review	Confirm that the AI-accessible context is authorized, current, complete, and relevant.
Autonomy Allocation Review	Select risk class, responsibility allocation, permissions, and human checkpoints.
Evidence Review	Evaluate proof and unresolved uncertainty rather than manually rereading every generated artifact.
Trust Gate	Authorize progression, require remediation, reduce autonomy, or reject the change.
Outcome Learning Review	Examine production outcomes and adjust evaluations, context, controls, and autonomy.

These ceremonies may be synchronous, asynchronous, or policy-automated. HAGE does not require a fixed sprint duration. Work may advance continuously when evidence and trust gates are satisfied.

XI. MEASUREMENT MODEL

HAGE measures whether AI-enabled engineering produces trusted outcomes rather than merely more output.

Metric	Definition
Trusted Throughput	Number or value of changes reaching production with complete required evidence and no defined trust failure.
Verification-Adjusted Lead Time	Elapsed time from approved intent to trusted release, including review and evidence latency.
Human Attention per Trusted Change	Qualified human review and decision time consumed per accepted production change.
AI Contribution Acceptance Rate	Proportion of material AI contributions accepted after required verification.
Human Override Rate	Frequency with which authorized humans alter, block, or reverse AI decisions.
Evidence Completeness Index	Percentage of mandated evidence items present and valid at trust gate.
Autonomy Utilization	Share of eligible work performed at each autonomy level.
Autonomy Regression Rate	Frequency at which incidents or failures require autonomy reduction.
AI-Linked Defect Escape Rate	Production defects attributable to AI-produced or AI-modified artifacts.
Context Quality Score	Freshness, relevance, provenance, authorization, and contradiction status of supplied context.
Recovery Readiness	Measured ability to detect, contain, and reverse an AI-influenced failure.
Outcome Realization	Degree to which the released change achieves the intent's business and operational objectives.

XII. HAGE ENTERPRISE MATURITY MODEL

Level	Characteristics	Primary control objective
Level 0 — Human-Centric	AI is absent or prohibited. Existing SDLC and deterministic automation dominate.	Establish use policy and baseline metrics.
Level 1 — AI-Assisted	Individuals use AI for isolated drafting, coding, explanation, or testing.	Control data exposure and require human review.
Level 2 — Workflow-Integrated	AI is embedded in approved lifecycle activities and connected to enterprise context.	Standardize context, provenance, evaluations, and tool access.
Level 3 — Governed Human–AI Engineering	Risk classes, P-A-V-E roles, autonomy contracts, and evidence gates are institutionalized.	Measure trusted throughput and operational outcomes.
Level 4 — Agentic Engineering	Specialized agents execute multi-step work across lifecycle stages under bounded authority.	Strengthen separation of duties, simulation, observability, and rollback.
Level 5 — Adaptive Governed Autonomy	Autonomy and controls adapt using validated performance and production learning.	Prevent self-reinforcing failure through independent governance and periodic recertification.

XIII. ENTERPRISE IMPLEMENTATION BLUEPRINT

A. Entry point

An enterprise applies HAGE when a new initiative, change request, modernization program, incident, or operational improvement is proposed. The initiative enters an intent intake rather than immediately becoming a conventional backlog item.

B. Minimum viable adoption

Select one bounded product or engineering value stream.

Baseline current lead time, review effort, defects, and rework.

Define two or three risk classes and permitted autonomy levels.

Create intent-contract and evidence-package templates.

Connect AI only to approved context and sandboxed tools.

Introduce one trust gate before merge or environment promotion.

Measure outcomes for at least several comparable changes.

Expand autonomy only after evidence demonstrates stable performance.

C. Operating architecture

Human Governance	Intent, authority, exceptions	Business / Architecture / Security / Operations
HAGE Control Plane	Risk classification, policy, identity, autonomy, traceability	Governance services
Engineering Agents	Planning, architecture, code, testing, release, support	Model and agent layer
Assurance Plane	Tests, scans, simulation, policy checks, independent review	Verification systems
Engineering Platforms	Repositories, CI/CD, cloud, observability, ITSM, knowledge	Enterprise toolchain

XIV. ILLUSTRATIVE EXAMPLE: PAYMENT-SERVICE CHANGE

A bank requests a new payment-status API. Under a conventional process, requirements, design, implementation, testing, security review, deployment, and support readiness may proceed through separate queues. Under HAGE:

Intent: Define consumer outcome, latency target, data classification, authentication requirements, failure behavior, and evidence required.

Risk: Classify as high due to payment domain and sensitive data.

Allocation: AI drafts API contract and threat scenarios; an agent scaffolds code and tests; humans own architecture and business correctness.

Autonomy: Agent may create a branch, execute tests, and open a pull request, but cannot merge, access production secrets, or deploy.

Evidence: Contract tests, authorization tests, dependency scan, threat-model checks, performance tests, and traceability are assembled automatically.

Trust gate: Architect, security owner, and product authority review evidence and unresolved risks.

Release: Progressive deployment with rollback, telemetry, and transaction anomaly monitoring.

Learning: Defects, review findings, and production telemetry update the evaluation suite and future autonomy eligibility.

XV. RESEARCH HYPOTHESES AND VALIDATION PLAN

Hypothesis	Statement
H1	HAGE reduces verification-adjusted lead time relative to an unchanged AI-assisted Agile/DevOps process.
H2	Risk-based autonomy reduces human attention per trusted change without increasing defect escape.

H3	Intent contracts and evidence packages improve traceability and review decision quality.
H4	Production-linked autonomy adjustment improves reliability over static autonomy assignments.
H5	Trusted throughput correlates more strongly with operational value than raw AI usage or code-generation volume.

A. Proposed study design

- Select comparable teams, products, or work-item categories.
- Establish a baseline using the organization's existing AI-assisted process.
- Introduce HAGE controls incrementally to avoid confounding all variables at once.
- Measure lead time, attention, evidence completeness, rework, defects, incidents, and outcomes.
- Conduct qualitative interviews on trust, accountability, cognitive load, and role change.
- Repeat across at least one regulated and one non-regulated environment.

XVI. LIMITATIONS, RISKS, AND OPEN QUESTIONS

- HAGE is conceptual and has not yet been validated through controlled longitudinal studies.
- The framework may introduce governance overhead if applied indiscriminately to low-risk work.
- Metrics such as AI contribution and defect attribution may be difficult to measure consistently.
- Excessive reliance on AI-generated evidence can create correlated failure unless verification is sufficiently independent.
- High autonomy may weaken skills, ownership, or system understanding if organizational safeguards are inadequate.
- Legal and regulatory accountability varies by jurisdiction and industry.
- Rapid changes in models and agent capabilities may require frequent revision of autonomy assumptions.
- The term "engineering participant" is functional, not a claim of legal agency, personhood, or moral responsibility.

XVII. CONCLUSION

Generative and agentic AI do not simply make existing engineering tasks faster. They introduce non-human systems capable of performing material engineering responsibilities and acting on enterprise environments. The resulting challenge is not only productivity; it is the allocation of authority, verification, accountability, and trust.

HAGE proposes a structured response. It preserves human accountability while enabling bounded non-human execution. It replaces indiscriminate manual review with risk-proportionate evidence and trust gates. It connects engineering actions to production outcomes and uses those outcomes to govern future autonomy. Its intended value is an end-to-end enterprise operating model that converts AI capability into trusted delivery rather than isolated acceleration.

The next stage of this research is empirical: refine definitions, compare HAGE with adjacent methods, pilot the lifecycle in real engineering value streams, test the proposed metrics, and publish the findings for peer review.

APPENDIX A — HAGE MANIFESTO (DRAFT)

We are discovering better ways for humans and intelligent systems to engineer software together.

Governed collaboration over isolated automation

Verified evidence over assumed correctness

Accountable autonomy over unrestricted delegation

Explicit engineering intent over implementation activity

Trusted outcomes over artifact volume

Continuous operational learning over static process compliance

While the items on the right may retain value, HAGE places greater value on the items on the left.

APPENDIX B — WORKING GLOSSARY

Intent Contract: A governed specification of outcome, constraints, prohibited actions, acceptance evidence, and accountable authority.

Qualified Context Pack: Authorized and quality-checked information supplied to AI or agentic participants.

Autonomy Contract: Machine- and human-readable boundaries governing tools, data, environments, actions, budgets, escalation, and rollback.

Evidence Package: The collection of provenance, tests, scans, policy results, assumptions, and uncertainty supporting a trust decision.

Trust Gate: A risk-proportionate decision point that permits, blocks, limits, or escalates lifecycle progression.

Trusted Throughput: Verified changes delivering intended value under required governance and reliability criteria.

Engineering Participant: A human or non-human system that materially contributes to an engineering activity; the term does not imply legal personhood.

Definition of Trusted: The evidence and accountability conditions required for a change to progress or operate.

REFERENCES

[1] Beck, K. et al. “Manifesto for Agile Software Development.” Agile Manifesto, 2001. https://agilemanifesto.org/

[2] Agile Manifesto authors. “Principles behind the Agile Manifesto.” https://agilemanifesto.org/principles.html

[3] CMMI Institute. “CMMI: Global Best Practices for Improving Capability and Performance.” https://cmmiinstitute.com/CMMI

[4] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0), NIST AI 100-1, 2023.

[5] National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile, NIST AI 600-1, 2024.

[6] Bandara, E. et al. “Agentsway — Software Development Methodology for AI Agents-based Teams.” arXiv:2510.23664, 2025.

[7] Bhati, H. “Agentic AI in the Software Development Lifecycle: Architecture, Empirical Evidence, and the Reshaping of Software Engineering.” arXiv:2604.26275, 2026.

[8] Alenezi, M. “Rethinking Software Engineering for Agentic AI Systems.” arXiv:2604.10599, 2026.

[9] “Graduated Human Oversight for Agentic Code Generation.” arXiv:2606.22484, 2026.

[10] Mandl, P. and Mandl, P. “AI-driven Software Development: A Pragmatic Path to Agentic Development Processes.” arXiv:2606.15283, 2026.

[11] “Agentic Software Engineering: Foundational Pillars and a Research Roadmap.” arXiv:2509.06216, 2025.