A METHODOLOGY FOR THE AI AGE

KANAME

A specification-driven, flow-based methodology for AI-augmented software delivery.

v1.0 · May 2026

Purpose of This Guide

This guide contains the definition of Kaname. Each element serves a specific purpose essential to the value and results Kaname delivers. Removing elements, bypassing human gates, or operating without structured markdown artifacts limits the benefits of Kaname. A team that removes any element is not running Kaname.

Kaname governs how AI-augmented work flows through a delivery system. It presupposes that psychological safety, collaborative trust, and quality engineering practices are already present in the team. Kaname does not create these conditions - it requires them.

As Kaname is being used, patterns, processes, and insights that fit it as described here may be found, applied, and devised. Their description is beyond the purpose of this guide because they are context-sensitive and differ widely between implementations. Teams are encouraged to develop, document, and share them.

Kaname Definition

Kaname is a specification-driven, flow-based methodology for AI-augmented software delivery. It is founded on a single shift: artificial intelligence has relocated the primary delivery constraint from implementation capacity to requirements clarity. It helps teams produce software that faithfully reflects human intent by making structured specifications the source of truth and AI agents the implementation mechanism.

In Kaname:

1. A Spec Owner authors a specification that precisely defines system behavior.
2. Spec-Ready Use Cases are selected for the delivery cycle.
3. Human judgment approves the selected Use Cases before any technical planning begins.
4. AI agents generate a technical plan; the Plan Reviewer approves it before tasks are generated.
5. Task Implementers pull tasks and verify AI-generated code against the specification at each merge.
6. The delivered system is verified against all Use Cases in the current delivery cycle before release.
7. Repeat.

Steps 2 through 6 constitute one delivery cycle - the arc from Use Case selection to verified release. A delivery cycle is scope-defined, not time-boxed.

Kaname is purposefully incomplete in its tactical expression. How teams write specifications, configure AI agents, or structure their boards varies by context. This guide defines the elements that must be present. How those elements are operated is left to the team.

Kaname Theory

Kaname is founded on three convictions.

Specification primacy. The specification is the source of truth. Code is a derived, regenerable output. Specifications endure; implementations are replaced. When specification and code conflict, the specification governs.

Flow over cadence. Work moves continuously through defined stages governed by explicit policies - stated rules about how work enters, advances, and exits each stage. Time-boxed iterations are not required. Work is pulled, not pushed. Limiting work in progress - the count of active items in any stage - improves delivery more reliably than increasing effort.

Human governance over AI autonomy. Artificial intelligence generates code at speed and scale. Human judgment governs intent, architecture, and acceptance. No AI output advances through the delivery system without passing a human gate.

Kaname Values

Successful use of Kaname depends on the team living five values:

Specification, Accountability, Transparency, Governance, and Flow.

The team commits to the specification as the single source of truth - not intuition, convenience, or verbal agreement. Each role is accountable for its defined scope without delegation. Work, decisions, and blockers are transparent to all. Every human gate is exercised without exception. And flow is continuous - work is pulled when capacity exists, not batched into artificial cycles.

These values give direction to the Kaname team in their work, decisions, and behavior. The choices made, the gates exercised, and the way Kaname is used should reinforce these values, not diminish or undermine them.

Kaname Team

The fundamental unit of Kaname is a small team of people operating with five defined roles. Four roles each own one primary artifact and one layer of the delivery governance chain. The fifth role - the Delivery Coach - owns the delivery system itself rather than a specific artifact. No accountability may go unassigned. If any artifact or the delivery system lacks a named owner, the team is not running Kaname.

Kaname roles are not job titles. One person may hold multiple roles. What may not be compressed is accountability - each artifact and the delivery system must have a named owner.

Constitution Guardian

The Constitution Guardian is accountable for the integrity of the system's architectural, security, and quality boundaries. They define what the system may and may not do, independent of any specific feature or delivery cycle.

The Constitution Guardian is accountable for:

- Authoring and maintaining `constitution.md`
- Reviewing AI-generated output for constitutional compliance
- Approving all changes to constitutional constraints
- Vetoing implementations that violate architectural or security boundaries

The Constitution Guardian is one person. Others may propose constitutional changes. Only the Constitution Guardian ratifies them.

Spec Owner

The Spec Owner is accountable for the completeness and precision of behavioral specifications. They translate stakeholder intent into a form that both humans and AI agents can act on without ambiguity.

The Spec Owner is accountable for:

- Authoring and maintaining the specification
- Ensuring each Use Case defines complete inputs, outputs, variants, and acceptance criteria
- Reviewing AI-generated output for specification alignment
- Approving specification changes before implementation proceeds

The Spec Owner owns the specification. Stakeholders may express intent. The Spec Owner determines how that intent is encoded. A specification that requires clarification during

implementation is an incomplete specification.

Plan Reviewer

The Plan Reviewer is accountable for the architectural soundness of the technical plan derived from the specification. They bridge behavioral intent and technical implementation, ensuring that what the AI generates as architecture is both feasible and aligned with constitutional constraints.

The Plan Reviewer is accountable for:

- Reviewing AI-generated `plan.md`, data model, and API contracts
- Identifying architectural risks before implementation begins
- Verifying that the technical plan conforms to constitutional constraints
- Approving the plan before tasks are generated and pulled

The Plan Reviewer does not author the plan. The AI generates the plan from the specification. The Plan Reviewer determines whether that plan is acceptable.

Task Implementer

The Task Implementer is accountable for executing work pulled from `tasks.md` using AI coding assistants, within the constraints set by the specification and constitution. They are the primary interface with AI execution tooling.

Task Implementers are accountable for:

- Pulling tasks according to WIP limits
- Operating AI agents with the appropriate artifact context for each task
- Verifying AI-generated code against the relevant Use Case before marking the task complete
- Flagging specification ambiguities discovered during implementation

Task Implementers do not unilaterally resolve specification ambiguity. Ambiguity discovered during implementation is returned to the Spec Owner.

Delivery Coach

The Delivery Coach is accountable for the health of the delivery system - the policies, WIP limits, gate criteria, and board state that govern how work moves through Kaname. In AI-augmented delivery, the Delivery Coach is the primary guardian of human gates: ensuring that the speed of AI generation does not pressure the team into bypassing human judgment.

The Delivery Coach is accountable for:

- Facilitating the Flow Review
- Ensuring all Kaname events are held as defined and all gates are exercised without exception

- Monitoring board health, WIP limits, and blocked items between events
- Surfacing impediments and escalating those that cannot be resolved within the team
- Coaching team members on Kaname practices and values

The Delivery Coach does not own a primary artifact. Their accountability is the delivery system - not what the system produces, but how it operates. The Delivery Coach does not make content decisions: specification, architecture, and implementation remain with the roles that own them.

Kaname Artifacts

Kaname's artifacts represent the team's current understanding of intent, architecture, and work. Each is a structured markdown artifact with a designated owner. Ownership means accountability for quality and currency - not exclusive authorship. Each artifact carries a commitment: a binding quality standard against which the artifact is evaluated at its corresponding human gate or review event.

Each artifact's commitment is:

- For `constitution.md` it is the Technical Contract.
- For `spec.md` it is the Behavioral Intent.
- For `plan.md` it is the Architecture Alignment.
- For `tasks.md` it is the Implementation Completeness.

The specification, plan, and tasks form a directed chain. When the specification changes, AI agents regenerate the plan from it; when the plan changes, AI agents regenerate the task breakdown from it. Code is the final output of this chain. Specification changes may occur at any point in a delivery cycle - each one reruns the cascade. This is how specification primacy operates in practice.

constitution.md

The constitution defines the non-negotiable boundaries of the system: architectural decisions, approved technology constraints, security requirements, and quality standards. It is the highest authority in Kaname. All other artifacts and all AI-generated output must conform to it.

Commitment: Technical Contract. The constitution is complete when every constraint it contains is verifiable and when a reviewer can assess any AI-generated output against it without ambiguity.

spec.md

The specification defines system behavior from the user's perspective. It is organized as System Use Cases - each describing one meaningful interaction between an actor and the system, including all successful paths, variants, and failure conditions. As systems grow, teams may organize the specification across multiple documents by domain or Epic. The artifact is the totality of Use Case specifications, regardless of physical structure.

Commitment: Behavioral Intent. The specification is complete when a Task Implementer can execute any Use Case using an AI agent without requesting clarification, and when every acceptance criterion is independently verifiable by a human reviewer. A Use Case that meets this standard is Spec-Ready.

plan.md

The plan describes the technical architecture for implementing the specification. It includes the component structure, data model, API contracts, and implementation sequence. The plan is generated by an AI agent from the specification and reviewed by the Plan Reviewer.

Commitment: Architecture Alignment. The plan is complete when every Use Case in scope for the current delivery cycle maps to a defined technical component, no implementation decision contradicts the constitution, and the sequence is feasible given known dependencies.

tasks.md

The task breakdown translates the plan into discrete, executable units of work ordered for implementation. Each task maps to one Use Case and one plan component. Each task is self-contained: it carries the Use Case reference, relevant acceptance criteria, and plan component context sufficient for an AI agent to implement it without external clarification. Tasks are pulled by Task Implementers from the board as capacity allows.

Commitment: Implementation Completeness. A task is complete when the AI-generated output satisfies its linked Use Case acceptance criteria and passes the Implementation Gate review.

Kaname Human Gates

Kaname defines four mandatory human gates - decision points at which human judgment must approve before work advances. Gates cannot be bypassed by any role, including the roles that own the upstream artifacts. An AI agent cannot carry work across a gate boundary.

Gate 1 - Specification Gate

The purpose of the Specification Gate is to verify that the specification is complete and precise before technical planning begins.

The Spec Owner and Constitution Guardian jointly inspect the Use Cases selected for the current delivery cycle. Each Use Case is reviewed for completeness, constitutional alignment, and implementability. Work does not proceed to the Plan stage until both have approved.

Gate 2 - Plan Gate

The purpose of the Plan Gate is to verify that the AI-generated technical plan is architecturally sound and specification-aligned before task generation begins.

The Plan Reviewer inspects `plan.md`, the data model, and API contracts. The Spec Owner confirms that the plan covers all Use Cases in scope for the current delivery cycle. Work does not proceed to task generation until the Plan Reviewer has approved.

If plan generation reveals specification gaps, the Spec Owner resolves them before plan approval proceeds. Use Cases whose gaps cannot be resolved without significant rework are withdrawn from the current delivery cycle and returned to the specification stage.

Gate 3 - Implementation Gate

The purpose of the Implementation Gate is to verify that AI-generated code satisfies the linked Use Case before it is merged into the codebase.

The Task Implementer reviews AI output against the relevant Use Case specification and acceptance criteria. Code that does not satisfy the Use Case returns to implementation. The Task Implementer does not resolve genuine ambiguity unilaterally - those cases are escalated to the Spec Owner.

When escalation reveals a specification change rather than an ambiguity, the Spec Owner assesses urgency. Non-critical changes are recorded and queued for the next Spec Review Session. Changes that affect the integrity of the current delivery cycle are escalated to the Delivery Coach, who determines whether the affected Use Cases are withdrawn from the cycle and returned to the specification stage.

Gate 4 - Delivery Gate

The purpose of the Delivery Gate is to verify that the delivered system satisfies all Use Cases in the current delivery cycle before release.

The Spec Owner performs end-to-end verification of all implemented Use Cases in the current delivery cycle. A release is not authorized until the Delivery Gate is passed.

Kaname Events

Kaname defines six events. Each event is a formal opportunity to inspect the artifact chain and adapt based on findings. Events are triggered by defined conditions, by regular cadence, or both - as specified for each event. Failure to hold events as defined results in specification drift, unreviewed architectural decisions, or accumulated delivery debt.

Spec Review Session

The purpose of the Spec Review Session is to advance Use Cases in the specification toward Spec-Ready quality and to maintain a sufficient queue ahead of Queue Replenishment. It is the primary event at which the Spec Owner receives structured feedback on in-progress Use Cases, resolves open questions, and agrees with stakeholders on what to specify next. Running at a regular cadence, it prevents the Spec Owner from becoming a bottleneck upstream of implementation.

The Spec Owner facilitates. The Constitution Guardian and at least one Task Implementer attend. The Plan Reviewer and stakeholders attend when a delivery cycle is being prepared.

When a delivery cycle is about to begin, the Spec Review Session additionally serves as the formal review at which Use Cases selected for Queue Replenishment are confirmed complete and constitutionally aligned.

The Spec Review Session concludes when in-progress Use Cases are assessed, open items are recorded, and the next Use Cases to be specified are agreed. When serving as a pre-delivery review, it concludes when the selected Use Cases are confirmed Spec-Ready or blocking items are explicitly recorded.

Queue Replenishment

The purpose of Queue Replenishment is to select Spec-Ready Use Cases for the next delivery cycle and commit them toward the Specification Gate.

The Delivery Coach facilitates. The Spec Owner and stakeholders provide priority input. Only Use Cases confirmed Spec-Ready through the Spec Review Session are eligible for selection. Queue Replenishment does not include estimation, technical planning, or architectural discussion.

Queue Replenishment is triggered when the pool of Spec-Ready Use Cases is insufficient to sustain the next delivery cycle, or at a defined regular cadence. It concludes when a set of Use Cases is committed to the current delivery cycle.

Plan Review

The purpose of the Plan Review is to inspect the AI-generated technical plan for architectural soundness and alignment with the specification and constitution.

The Plan Reviewer facilitates. The Spec Owner and Constitution Guardian attend. The Plan Review concludes when the plan is approved or when required revisions are recorded.

Delivery Review

The purpose of the Delivery Review is to inspect the delivered system against its Use Cases and gather stakeholder feedback before the next delivery cycle begins.

The Spec Owner facilitates. Stakeholders, Task Implementers, and the Delivery Coach attend. The Delivery Review is triggered when the Delivery Gate passes - it is not calendar-driven.

Each implemented Use Case is presented to stakeholders. Stakeholders assess whether the delivered behavior matches their intent. Feedback that reveals gaps, new needs, or changed priorities is captured by the Spec Owner as input for the next Spec Review Session. The Delivery Review does not include technical planning or architectural discussion.

The Delivery Review concludes when all implemented Use Cases have been reviewed and stakeholder feedback is recorded. It formally closes the delivery cycle.

Flow Review

The purpose of the Flow Review is to inspect the system of work - not individual work items - to identify policy changes that would improve delivery flow. Teams may establish a regular cadence for the Flow Review, or trigger it when blocked items accumulate or lead time increases significantly.

The Delivery Coach facilitates. The full team inspects the board, lead time distribution, and blocked items. The Flow Review concludes when one or more policy changes are identified or when the team confirms that existing policies are functioning as intended. The output is a change to WIP limits, gate criteria, or work handling policies, or an explicit record that no change is warranted.

Constitution Review

The purpose of the Constitution Review is to inspect `constitution.md` for completeness and continued relevance, and to ratify proposed amendments.

The Constitution Guardian facilitates. Constitutional amendments require the explicit ratification of the Constitution Guardian. This event is triggered by a proposed amendment, by architectural findings surfaced during a Delivery Review, or by a scheduled post-delivery constitutional inspection. It concludes when the proposed change is ratified, revised, or rejected, and `constitution.md` is updated accordingly.

End Note

Kaname is free and offered in this guide. Kaname, as defined herein, is immutable.

Kaname does not work in fragments. Human gates, structured artifacts, and named role accountabilities are mutually dependent. A team that selects only parts of Kaname may build something useful - it is not Kaname, and should be named on its own terms.

Kaname operates alongside existing engineering practices, CI/CD pipelines, and AI tooling. It does not replace them. It governs how intent moves through them.
