Helmark STD 3.0

Standard for Responsible AI Software Delivery

Human-Controlled AI Software Delivery

Official name: Helmark Standard

Version: Helmark STD 3.0

Contents

01 Overview and Scope

02 Core Principles

03 Helmark Delivery Loop

04 AI Involvement and Criticality

05 Proportional Verification and Agent Autonomy

06 Understanding and Sustainable Human Control

07 Flow

08 Definition of Done, Metrics and Exceptions

09 Minimum Artefacts, Adoption Levels and Audit

10 Implementation, Quick Reference and Governance

1. Overview and Scope

The standard in nine sentences

1. AI may analyse, design, implement, test and verify software, but responsibility for the accepted result remains with people.
2. Every significant piece of work performed with AI has a human responsible for consciously closing it — the Human Owner.
3. AI involvement is classified as NONE, ASSIST or CORE.
4. Technical risk follows primarily from component criticality: LOW, HIGH or CRITICAL.
5. Work with AI follows a simple cycle: FRAME → RUN → CHECK → OWN.
6. FRAME defines four things: Outcome · Owner · Boundary · Evidence.
7. CHECK always separates two questions: Right Thing? — did we produce the right thing? Works Right? — was it implemented correctly?
8. A result is not done merely because it works. A human must have sufficient Evidence and sufficient understanding to consciously accept it.
9. An organisation should not increase the speed of AI work beyond people's ability to verify, understand and responsibly take ownership of results.

Purpose

Helmark answers one question:

> **How do we preserve human control over software delivery when AI performs an ever larger share of the work?**

The standard assumes that AI may perform a significant part of software-development work. It does not try to restrict that by default.

Helmark is not intended to:

- reduce AI use;
- require manual approval of every agent step;
- create additional bureaucracy;
- control prompts;
- measure how much code was written by a human versus AI.

Its purpose is to preserve direction, boundaries, verifiability, responsibility and understanding.

Scope

Helmark is intended for teams that build, develop and maintain software using:

- generative AI;
- coding assistants;
- autonomous and semi-autonomous agents;
- agents performing review;
- AI performing tests and analysis;
- agents using tools, repositories, data and environments.

Helmark works together with the team's existing way of working. It can be used with Scrum, Kanban, Scrumban, Pull-Request-centric flow or an organisation's own process.

Helmark does not replace security, privacy, legal requirements, organisational governance, access management, supply-chain security, deployment rules or industry regulation. Those requirements operate alongside Helmark.

What Helmark does not define

Helmark does not define organisational structure, job titles, ceremonies, estimation, sprint length, ticketing system, AI vendor, AI model, IDE, agent platform or prompt-writing method.

The standard defines the conditions for controlling the result — not a tool or project-management method.

Normative language

Term	Meaning
MUST	A requirement necessary for claiming conformance.
SHOULD	A recommended practice. Deviation is allowed when the organisation consciously chooses another solution.
MAY	An optional practice.

Core concepts

Term	Definition
Work Unit	A unit of work the team can consciously close: a ticket, issue, task, change, Pull Request or another unit used by the team.
Human Owner	The human responsible for consciously closing a Work Unit. The Owner does not need to create the solution personally and is not automatically responsible for every future incident. The role is to ensure the result has been understood, verified and consciously accepted.
Outcome	The state that should be true after the work is complete. Outcome describes the result, not the implementation.
Boundary	The limits within which a human or AI may work. A Boundary may cover repositories, components, systems, environments, data, tools, permissions and elements that must not be changed.
Evidence	The basis for judging that the result is correct. Evidence may come from tests, scenarios, runtime behaviour, observability, review, data, requirements, user feedback or domain knowledge.
Agent	AI capable of independently performing a series of actions leading to a result, such as analysing a repository, modifying files, running tests, using tools, delegating tasks or creating a Pull Request.
Cognitive Debt	A situation in which software works, but the team does not understand it well enough to safely evolve, diagnose and evaluate proposed changes.
Knowledge Anchor	A human who genuinely understands an important component and can explain it to others.
Intervention Boundary	The point at which the team is no longer able to independently judge whether a solution proposed by AI is correct.

2. Core Principles

R1 — Human Responsibility

> **Responsibility for an accepted change cannot be delegated to AI.**

AI may analyse, propose, implement, test, verify and document. AI cannot be the party responsible for final acceptance of a change. “The AI did it that way” is not a reason to skip verification.

R2 — Human Decision

AI may recommend a decision. The final decision concerning the Outcome, Boundary, acceptance of Evidence and acceptance of the result remains with a human.

R3 — Bounded Autonomy

AI may act autonomously within an accepted Boundary. An agent should not independently expand the goal, scope, permissions, data, systems or environments beyond the accepted Boundary. If such expansion is needed, return to FRAME.

R4 — Evidence Before Done

An agent saying “Done” is not Evidence. Working code is not automatically evidence of correctness.

> **Evidence must answer: Why do we have justified grounds to believe this result is correct?**

R5 — Understanding Matters

A team should not consciously increase dependency on a system it can no longer independently evaluate. AI may increase productivity. It cannot replace the team's ability to understand the most important parts of its own system.

3. Helmark Delivery Loop

> **FRAME → RUN → CHECK → OWN**

These are not new board statuses. They are not ceremonies. A new workflow is not required. They are four kinds of decisions that occur inside the team's existing process.

FRAME

FRAME answers: What exactly are we trying to achieve, and within what boundaries?

Element	Question
Outcome	What should change?
Owner	Who will consciously close this work?
Boundary	Where may AI act, and where may it not?
Evidence	How will we know the result is correct?

FRAME does not have to be a separate document. It may live in a ticket, issue, Pull Request, specification or another durable location.

Example

> **Outcome: the user can safely reset a password.**
> **Owner: Anna.**
> **Boundary: authentication and e-mail modules; no database-schema changes.**
> **Evidence: the token expires, cannot be reused and the complete reset scenario works correctly.**

When FRAME is mandatory

Full FRAME MUST exist for:

- every Work Unit classified as CORE;
- ASSIST involving a HIGH component;
- ASSIST involving a CRITICAL component;
- autonomous agent work.

For ASSIST / LOW, a shorter record of where AI was used and how the result was checked may be sufficient. For NONE, Helmark adds no FRAME requirement.

RUN

RUN means executing the work. It may be performed by a human, a human supported by AI, one agent or multiple collaborating agents.

Helmark does not require approval of every AI step. An agent may analyse, plan, generate code, modify code, test, review, create documentation and delegate tasks to other agents — provided the Boundary is respected.

Re-FRAME

If RUN creates a need to change the Outcome, expand the Boundary, materially change the Evidence or enter a new higher-risk component, the work returns to FRAME.

Re-FRAME does not mean restarting the task. It means consciously updating the decisions.

CHECK

CHECK answers two independent questions.

Question	What it verifies	Possible evidence
RIGHT THING?	Did we produce the outcome that was actually needed?	Outcome, expected behaviour, requirements, user scenario, business knowledge, data, feedback.
WORKS RIGHT?	Was the solution implemented correctly?	Tests, build, code review, security checks, runtime, observability, dependency analysis, failure scenarios.

A technically correct implementation can solve the wrong problem. A correct Outcome can be implemented badly. RIGHT THING? + WORKS RIGHT? therefore form CHECK together.

If the answer to either question is insufficient, the work returns to RUN or FRAME.

OWN

OWN means conscious human acceptance of the result.

Before OWN, the Human Owner should be able to answer:

1. What changed?
2. Why do we believe the result is correct and works?
3. What is the most important thing that could go wrong?

The Human Owner does not need to know every line of code. They must understand the solution well enough for maintenance, further change, diagnosis and conscious acceptance.

4. AI Involvement and Criticality

NONE / ASSIST / CORE

Level	Meaning

NONE	AI did not participate in creating the result.
ASSIST	AI supported the human, but the human created the main result.
CORE	AI created the substantive part of the result, while the human primarily requested, integrated or verified it.

Classification is not a quality score. CORE does not mean worse work. It describes how the result was created.

Classification test

> **If the AI contribution disappeared, would a recognisable human-created implementation remain?**

If yes — usually ASSIST. If without AI there would mainly be a problem description, prompt, choice between variants or fixes applied to generated implementation — usually CORE.

Agent work and ownership

The result of autonomous or semi-autonomous agent work is CORE. This does not depend on the number of changed lines, solution quality, number of tests run by the agent or whether the agent corrected its own mistakes.

A Human Owner MUST be identified before autonomous agent work begins. If an agent runs cyclically or initiates work itself, the organisation may assign a standing Owner to a repository, component, work queue or change type.

Pre-assigning an Owner does not equal automatic acceptance. OWN must still happen consciously.

Trivial changes

A team MAY define a small written category of changes for which classifying AI involvement adds no value, for example typo corrections, formatting or automated import cleanup. The exception should not cover HIGH or CRITICAL components.

LOW / HIGH / CRITICAL

Helmark separates how much AI did from how serious the consequences of an error could be.

Level	Character
LOW	An error has limited impact and is usually easy to detect or reverse.
HIGH	An error may materially affect users, data, integrations, availability or a business process.
CRITICAL	An error may cause serious consequences involving security, money, data, privacy, permissions or critical processes.

Criticality belongs primarily to the component

The team should define default criticality for important components. A Work Unit inherits the component level. If it touches several components, it takes the highest level.

The Owner MAY raise the level for a specific change. Lowering it below the component level should be exceptional and justified.

5. Proportional Verification and Agent Autonomy

Verification principle

The greater the AI involvement and the more serious the possible consequences of error, the stronger the Evidence required before OWN.

> **Helmark does not assume: every AI use = maximum control. Control should be proportional.**

Minimum requirements

Case	Helmark minimum
NONE	The team's normal process.
ASSIST / LOW	Classification + short record of where AI was used and how the Owner checked the result.
ASSIST / HIGH	FRAME + test/scenario + real human review.
ASSIST / CRITICAL	HIGH requirements + an independent human verification criterion.
CORE / LOW	FRAME + test/scenario + real human review + OWN.
CORE / HIGH	CORE / LOW + conscious risk analysis of the specific change.
CORE / CRITICAL	CORE / HIGH + an independent human verification criterion.

Human Review

When the table requires human review, it is performed by a human other than the Human Owner. Review must be real. Automatic “LGTM” or an AI-tool approval does not satisfy the requirement. The reviewer should be able to explain the basis of the decision.

AI may support review

AI may analyse a diff, search for problems, generate questions, identify risks, analyse tests and look for vulnerabilities. AI review may be Evidence. It does not replace required human review.

Independent criterion for CRITICAL

For CRITICAL changes, at least one significant criterion answering “What should the correct result be?” MUST come from an independent human source.

That source may be:

- a requirement;
- domain knowledge;
- a business decision;
- a security decision;
- a threat model;
- a legal or regulatory rule.

AI may help create a test, but should not be the only source of knowledge about what the test is supposed to prove.

Boundary for agents

For agent work, the Boundary should specify what matters for the specific Work Unit. It may include repositories, modules, databases, APIs, environments, deployment ability, executable commands, accessible data, tools and the ability to start additional agents.

Not every Work Unit needs every element. Describe the boundaries whose violation could change the nature or risk of the work.

Agent access and automatic operation

An agent should have only the scope of action needed to perform the accepted Work Unit. Helmark does not define the technical permission-management system, but requires that autonomy not mean the absence of consciously defined limits.

An agent may create Pull Requests, run tests, prepare changes and perform periodic analysis. An organisation may permit further actions according to its security rules. An automatic tool action cannot by itself constitute OWN.

6. Understanding and Sustainable Human Control

Cognitive Debt

Cognitive Debt arises when software changes faster than the team's ability to understand it. Code may work, have tests and pass CI while still carrying high Cognitive Debt.

Typical signal

> **“I don't know exactly how it works. The agent did it.”**

This does not automatically mean there is a problem. Risk appears when the team cannot explain behaviour, diagnose failure or evaluate the correctness of a proposed change.

Knowledge Anchors

Every important component SHOULD have at least one Knowledge Anchor. An Anchor should be able to explain the component, identify key dependencies, identify key risks and transfer the knowledge to another person.

For CRITICAL components, the organisation should aim to avoid concentrating knowledge in only one person.

An Anchor is not a component owner

Knowledge Anchor does not mean “the person responsible for every problem in the component.” It answers one question: who can genuinely explain it?

Handover

Handover is required when responsibility for an important component consciously moves to another person or team.

A minimum Handover answers:

1. What does this element do?
2. Where is the most important risk?

The purpose is transfer of understanding, not extensive documentation.

AI may rebuild understanding

Helmark does not assume AI only increases Cognitive Debt. AI may help reduce it by explaining code, mapping dependencies, building diagrams, identifying failure modes, analysing change history, creating scenarios and supporting onboarding.

AI may help recover knowledge. It does not replace human ability to independently judge.

Intervention Boundary

> **Can we understand a failure of this component and judge whether the proposed solution is correct?**

A component crosses the Intervention Boundary when the team can no longer independently answer that question.

If the answer is “no”, the team MUST consciously decide what to do next. It may rebuild knowledge, reduce risk through another mechanism, limit further change or accept the risk. The decision should be explicit.

7. Flow

The new bottleneck

AI may produce results faster than people can check them. Performance should therefore not be judged only by the number of generated changes. The ability to perform CHECK + OWN becomes critical.

CHECK Load

CHECK Load is work that has been generated or performed but is still waiting for verification, understanding or human acceptance.

Growing CHECK Load may mean the system is generating more work than it can responsibly finish.

CHECK FIRST

> **If CHECK Load grows systematically: CHECK FIRST.**

The team should consider:

- reducing parallel Work Units;
- limiting the number of active agents;
- finishing existing work;
- improving Evidence;
- simplifying the Boundary.

Do not optimise RUN speed independently of CHECK and OWN.

8. Definition of Done, Metrics and Exceptions

Done for AI-assisted work

> **RIGHT OUTCOME + EVIDENCE + UNDERSTANDING + HUMAN ACCEPTANCE = DONE**

For NONE, the team's Definition of Done applies. Helmark adds the above requirements where AI influences the result according to this standard.

System metrics

Metric	Meaning
Lead Time	Time from the start of FRAME to OWN.

CHECK Load	Amount of work waiting for checking or ownership.
Return Rate	How often CHECK sends work back to RUN.

Activity is not productivity

Do not automatically interpret the following as productivity:

- prompt count;
- token count;
- agent count;
- generated lines of code;
- AI-created commit count.

These are activity measures. They do not prove value or delivery improvement.

Emergency work

An incident may require very fast action. Helmark allows the recording of FRAME or Evidence to be simplified during an emergency.

It does not remove the Human Owner, conscious acceptance or human responsibility. Missing records should be completed when operational conditions allow.

Conscious exception

If a Helmark requirement cannot be met, the exception should be explicit. Record at least:

- what was not done;
- why;
- who consciously accepted the risk;
- what action replaces the missing control or when the situation will be reviewed again.

9. Minimum Artefacts, Adoption Levels and Audit

Helmark does not require its own application

The organisation MUST only be able to record and find the required information:

Artefact	Minimum
AI involvement	NONE / ASSIST / CORE.
Component criticality	LOW / HIGH / CRITICAL.
FRAME	Recorded where required.
Evidence / review	Information showing CHECK actually occurred.
Human acceptance	Ability to identify the Human Owner.
Search	Ability to find e.g. CORE, HIGH and CRITICAL items for review or audit.

Full AI transcript is not required

Helmark does not require archiving full prompts, complete conversations, every agent action or its internal work trace.

Keep enough information to determine later: what we wanted, within what boundaries AI worked, and on what basis the result was accepted.

Adoption levels

Level	Name	What is true

L1	Visible Control	R1 is known; NONE / ASSIST / CORE works; agent work is CORE; Human Owners are visible; important components have criticality; minimum verification is proportional.
L2	Controlled Delivery	L1 plus: FRAME works where required; Boundaries are respected; CORE has required human review; HIGH and CRITICAL receive stronger verification; controls can be found and sampled.
L3	Sustainable Human Control	L1 and L2 plus: Knowledge Anchors and gaps are visible; Handover supports knowledge transfer; Intervention Boundary and Cognitive Debt drive decisions; CHECK Load is observed and leads to action.

L1 asks: Do we know where AI performs significant work and who accepts it?

L2 asks: Can we demonstrate that control is actually happening?

L3 asks: Can we preserve human ability to control the system over the long term?

Audit principle

A declaration that “we use Helmark” is not evidence of conformance. Audit should rely on artefacts, observation and samples of real changes.

L1 audit

- Do Work Units carry credible NONE / ASSIST / CORE classifications?
- Is autonomous agent work marked CORE?
- Is the Human Owner identifiable?
- Do significant components have defined criticality?
- Does ASSIST / LOW show where AI helped and how the result was checked?
- Does a random CORE item have required Evidence and human review?

L2 audit

- Does CORE have Outcome, Owner, Boundary and Evidence?
- Can an agent avoid routinely expanding its Boundary without a human decision?
- Do HIGH and CRITICAL receive increased verification?
- Does CRITICAL have an independent human correctness criterion?
- Does review show real human work rather than only acceptance of AI output?
- Can the organisation find CORE / HIGH / CRITICAL items for control?

L3 audit

- Do important components have visible Knowledge Anchors?
- Are components with insufficient understanding visible?
- Can the team identify components beyond the Intervention Boundary?
- Does Cognitive Debt lead to action?
- Does rising CHECK Load trigger action rather than more AI output?

10. Implementation, Quick Reference and Governance

Simplest Helmark adoption

1. Introduce NONE / ASSIST / CORE and the rule: Responsibility cannot be delegated to AI.
2. Define LOW / HIGH / CRITICAL for the most important components.

3. For CORE, start using FRAME → RUN → CHECK → OWN.
4. During CHECK always ask: Right Thing? Works Right?
5. Observe CHECK Load.
6. Once the basics work, add Anchors, Cognitive Debt and Intervention Boundary.

Integration with existing flow

Existing process	Helmark
Backlog	Need / problem
Ready / Planning	FRAME
Development	RUN
Review / Test	CHECK
Acceptance	OWN
Done	OWNED

Do not create a separate Helmark board if the existing board is sufficient.

Quick Team Card

Stage	Team prompt
FRAME	Outcome — What should change? Owner — Who will consciously close the result? Boundary — Where may AI act? Evidence — How will we know it works?
RUN	NONE · ASSIST · CORE Let people and agents work.
CHECK	RIGHT THING? — Did we build the right thing? WORKS RIGHT? — Did we build it correctly?
OWN	Can a human say what changed, why we believe it works, and what could fail?

Five sentences the team should remember

Humans decide.

AI stays within boundaries.

Evidence before ownership.

Understanding before dependency.

CHECK before more output.

Formal governance

Helmark has a versioned specification. A MINOR change means clarification without changing the main model. A MAJOR change may alter requirements, structure or the operating model.

An organisation claiming conformance states the version and level, for example: Helmark STD 3.0 — L2.

The official version is published by Helmark. The standard may be publicly read and implemented independently of training or certification. Training, certification and audit are services separate from the right to use the standard.

Final definition

> **Helmark is a standard that helps teams preserve human control over software delivery as AI performs an ever larger share of the work.**

It does not restrict AI autonomy where autonomy creates value. It requires, however, that direction be conscious, boundaries be known, the result be verifiable, acceptance be human and the organisation not lose the ability to understand its own system.

HELMARK STD 3.0

Human-Controlled AI Software Delivery

> **FRAME → RUN → CHECK → OWN**

Official source and updates: https://helmark.org

The **Helmark Standard** is licensed under the **Creative Commons Attribution-ShareAlike 4.0 International License (CC BY-SA 4.0).**

You are free to use, share, reproduce and adapt the Helmark Standard, including for commercial purposes, provided that appropriate attribution is given and any adapted material is distributed under the same or a compatible licence.

When quoting, reproducing, translating or adapting content from this standard, please provide attribution to:

Helmark Standard — Piotr Sobiegała and Mateusz Pluta — helmark.org

By using this publication, you acknowledge that its use is subject to the terms of the Creative Commons Attribution-ShareAlike 4.0 International License.

Authors: Piotr Sobiegała and Mateusz Pluta
Official website: helmark.org
Contact: contact@helmark.org
License: Creative Commons Attribution-ShareAlike 4.0 International (CC BY-SA 4.0)