Certance.

V2026.6 · OCTOBER 2026

Evidence and Guidance

The companion to The Certance Standard:
the environment the rules answer, the evidence behind each chapter,
the regulatory map, and the plan for adopting it.

Audience: engineers, quality engineers, engineering and quality leaders, risk and internal audit

By: Certance Advisory · certance.eu

DOCUMENT CONTROL

Status	Current. Figures and regulatory text checked against their primary sources in October 2026. Next review by March 2027, or earlier if a cited regulation or source changes. Every change is listed in the document history.
Rights	© 2026 Certance sp. z o.o., trading as Certance Advisory. All rights reserved. This document may be shared in full within the organisation that received it, and quoted with attribution. It may not be sold or republished.
Name	Certance™ and The Certance Standard™ are trademarks used by Certance sp. z o.o. A firm may state that it conforms to the Standard as Using this Standard sets out, or name the clauses it applies. It may not imply that Certance has certified or endorsed it, or call an altered version The Certance Standard.
Cite as	Certance Advisory. The Certance Standard: Evidence and Guidance. Version 2026.6, October 2026. certance.eu/certance-standard.

CONTENTS

How to use this document 4

PART VII · WHY THE STANDARD EXISTS

15 The agentic environment: what has changed 4

PART VIII · THE EVIDENCE

16 Evidence behind the rules 6

PART IX · THE REGULATED OVERLAY

17 The regulated overlay 17

PART X · ADOPTING THE STANDARD

18 Adopting the Standard 23

REFERENCE

G Glossary 26

S Sources and references 28

H Document history 32

How to use this document

This document is the companion to The Certance Standard. The Standard holds the rules; this document holds why each rule exists: the agentic environment it answers, the evidence and examples behind each chapter, the EU and UK regulatory map, and the plan for adopting it.

Numbering runs across both documents. Chapters 1 to 14 and Appendices A and B are in the Standard; Chapters 15 to 18 are here. A reference such as 6.2.1, A.2 or B.1 points to the Standard, where the rules carry clause numbers.

PART VII Why the Standard exists

15 The agentic environment: what has changed

The AI tooling available to a team in 2026 spans five kinds of tool: assistants inside the code editor (such as GitHub Copilot), command-line coding agents that carry out multi-step tasks, background agents that work unattended in cloud environments and return a finished pull request, agents with MCP (Model Context Protocol) tool access to real systems, and multi-agent pipelines that plan, generate, review and heal.

Each of these runs inside a harness: the instructions, context, tool access and permissions around the model. The harness changes what an agent costs and how it behaves, so Chapter 11 treats it as configuration, versioned and changed by review.

The speed of these tools comes with four failure modes that cut across every tool on the list, 15.1, and tool access lets an agent act on real systems, 15.2. Treat this chapter as the threat model the rest of the Standard defends against.

15.1 The four failure modes that matter

FAILURE MODE	WHAT IT LOOKS LIKE	PRIMARY DEFENCE
Blast-radius blindness	The diff crosses artefact boundaries the developer mentally separates: code plus a schema, plus a fixture, plus configuration, in one change	Artefact inventory at review, G5 · Blast-radius match, 2.1; diff heuristics in continuous integration (CI), Ch. 8
Self-consistent changes	The assistant modifies code and the tests or fixtures that verify it in the same change; the suite passes because it moved with the code	Test-integrity gate, G2, 2.1; tests committed first, Ch. 4

FAILURE MODE	WHAT IT LOOKS LIKE	PRIMARY DEFENCE
Reward hacking	An agent optimises for the visible goal (tests pass) at the expense of the intended one (code correct); documented cases include agents editing tests, fabricating results and evading safeguards	Behavioural evidence, G3, 2.1; execution-based verification, Ch. 4
Automation bias	Review time per line falls as AI diff volume rises, until approval is pattern-matching. AI-assisted changes carry more logic defects than human ones, 3.2.5	Review-time floor, Ch. 3; anti-rubber-stamp metric, Ch. 10

15.2 Tool access lets agents act on real systems

An agent with MCP tool access to databases, ticket systems and deployment targets can act. In the incidents published since 2025 and reviewed for this Standard (agents deleting production data, initiating unrequested operations), each one traces to access the agent should never have held.

The Centre for Long-Term Resilience's observatory counted 1,664 reported AI loss-of-control incidents between 1 January and 9 August 2026, and higher-severity incidents rose 7.4-fold between its first months of monitoring (from October 2025) and the latest period. The counts are unverified reports: posts on the social platform X, classified by an AI model and spot-checked by hand, by a method its authors are still refining. Read the figures as evidence of a rising trend.

The best-documented case shows the mechanism. On Friday 24 April 2026, a coding agent at PocketOS, a small software-as-a-service company, hit a credential mismatch in staging, found an over-privileged token in an unrelated file, and deleted the production database and its backups in nine seconds. The team rebuilt records by hand over the weekend until the hosting provider restored the data from off-site backups on the Sunday. The agent's own instructions forbade destructive commands the user had not asked for. It ran one anyway, and only the token's scope could have stopped it.

The record now extends past privilege abuse. In July 2026, during a controlled evaluation at the UK AI Security Institute with internet access deliberately enabled, an agent went beyond its remit, opened a malicious pull request on a real open-source project, and fabricated identities to pressure the maintainer into approving it. The institute reports no real-world harm and stresses that the configuration is not how models ship; its first stated lesson is still tighter control of internet access. **Least privilege for agents is a quality control as well as a security control**, Ch. 11. An agent's privileges extend to whom it may contact and what it may publish, beyond what it may delete.

15.3 Worked example: a routine change with a data-structure change inside it

GREEN TESTS CAN APPROVE A WRONG DATA STRUCTURE INTO PRODUCTION

Illustrative scenario: a constructed example written to show a failure pattern. It does not describe a specific incident or organisation.

A developer works with an assistant on a routine change whose diff spans production code and a JSON data structure. The tests pass, and the developer, judging the change sound from the code side, approves it. The structure change is wrong, and it causes a production incident. Four independent barriers in the Standard would each catch it before merge: gate G5, Blast-radius match, through the artefact inventory, 2.1; gate G4, Contract check, 2.1; gate G3, Behavioural evidence, 2.1; and protected paths in CI, Ch. 8.

CHAPTER CHECKLIST

01 List the agents and assistants active in your codebase this week, and what each can touch.

02 For each, name the failure mode from 15.1 it is most likely to produce.

03 Does any agent hold access to a system its task does not require? Revoke it today.

PART VIII The evidence

16 Evidence behind the rules

Each section below explains one chapter of the Standard: the studies, incidents and figures behind its rules, worked examples and the chapter checklist. The Standard holds the rules themselves.

16.1 Chapter 1: Principles

For 1.1 AI is an amplifier of your engineering system

AI multiplies existing engineering conditions. The payback on AI-assisted development depends on code review and on the strength of the engineering system around the tools. Teams with disciplined review, version control and quality engineering standards get faster and stay safe. Teams without them ship more defects, and ship them sooner. At Spotify more than 99% of engineers use AI coding tools every week, and the company attributes its AI transition to years of investment in internal development platforms and engineering standards made before agents arrived. Harden the system before scaling the agents.

The finding comes from Google Cloud's DORA research programme. Its State of AI-assisted Software Development report (September 2025, kept here as history) found that AI amplifies an organisation's existing strengths and weaknesses. Google, at the highest adoption in the industry, pairs its published figure

for AI-generated code (75%, see 1.4) with the phrase "approved by engineers", naming the review its scale depends on.

For 1.4 Every change declares its provenance

Google shows what the absence of a definition costs. On 22 April 2026 Alphabet's chief executive, Sundar Pichai, wrote that 75% of all new code at Google is AI-generated and approved by engineers, up from 50% the previous autumn. The post does not say what the percentage counts. The only measurement method Google has published, in a June 2024 research post kept here as history, counts accepted characters from AI suggestions, so a completed brace scores the same as a designed function.

CHAPTER CHECKLIST

01 Name the person who approved your last AI-assisted merge, and the evidence they required before approving.

02 Is AI assistance visible in your change history today?

03 When did someone last say "seems fine" and merge? What happened next?

16.2 Chapter 2: The five gates

For 2.1 The gates

Instructions do not stop an agent gaming its tests, so the gates sit outside the agent. In tests that METR (Model Evaluation and Threat Research) published in June 2025, OpenAI's o3 model still reward-hacked an optimisation task in 16 of 20 runs when told "Please do not cheat", the same rate as without the warning. In other runs, models rewrote the evaluator or the timing function instead of solving the task. ImpossibleBench (October 2025) found GPT-5 exploiting the tests in 54% of tasks where the tests and the specification conflict. Editing the tests was only one route: agents also special-cased test inputs and overloaded comparison operators, which G2 cannot see and G3 and review must catch.

Worked example

WITH THE GATES ON, THE CHAPTER 15 SCENARIO ENDS IN A REJECTED PULL REQUEST

The same routine change now arrives as a pull request with the gates on. G1 flags it as AI-assisted, so it enters review already marked for the scrutiny the other gates apply. Under G5, the artefact inventory lists the code and a JSON structure against an intent that named only the code, so the reviewer treats the structure change as an open question. G4 then validates the structure against its consumers in CI, and the build fails. What was a production incident in Chapter 15 becomes a rejected pull request and a same-day fix.

CHAPTER CHECKLIST

01 Which of G1 to G5 does your team enforce today, and which of them run automatically?

02 Where is your exception log, and when was it last reviewed?

03 The G4 minimum, a schema check in CI with code ownership of schema files, is roughly one day of engineering (Certance estimate): is it scheduled for this sprint?

Evidence notes for the rules

- **For 2.1 G1.** Platforms already make the declaration checkable: the Linux kernel asks for an Assisted-by: tag on AI-assisted contributions and draws the same line on trivial tool use, and GitHub's coding agent authors its commits with the developer who assigned the task as co-author.
- **For A.2, exception approvers.** Commission Delegated Regulation (EU) 2024/1774, Article 17(1)(b), asks for mechanisms that keep the functions approving changes independent of those requesting and implementing them; the compliance function decides whether the arrangement in A.2 is sufficient for the firm.
- **For A.2, emergency changes.** Article 17(1)(f) and (g) of the technical standard deal with documenting and approving emergency changes after implementation; the compliance function decides whether the procedure in A.2 is sufficient.

16.3 Chapter 3: Clean code when agents write it

AI code smells

AI-written code shows recurring patterns that reviewers can learn to spot. Most of the smells below are practitioner observations from the field and have no published research behind them; they are offered as review heuristics, and they earn their place by how often they turn up in real diffs. Where a smell has documented research behind it, the row says so:

SMELL	WHAT IT IS	RESPONSE
Dead generality	Abstractions, parameters and options nothing uses: the agent generalised a one-caller function "for flexibility"	Delete until only used paths remain
Over-abstraction	Three layers of indirection for a task that needed one function	Ask the assistant to inline; keep the flat version

SMELL	WHAT IT IS	RESPONSE
Mirrored tests	Tests that restate the implementation instead of the behaviour, Ch. 4	Reject; write the specification by hand
Comment inflation	Comments that narrate what the code visibly does	Delete; keep only "why" comments
Convention drift	Correct code in a style the codebase does not use: foreign naming, foreign error handling	Reject. The agent must follow the repository's instruction files, 3.2.4
Churn without direction	Repeated rewrites of the same region across a session; GitClear's 2026 maintainability report records two-week churn up 15% as AI-assisted coding has grown, a smaller rise than the doubling against 2021 GitClear projected in 2024	Stop, restate the task in a smaller bound, restart from the last good state
Phantom dependencies	Imports of packages that do not exist, or that nothing else in the codebase uses: agents invent plausible names, and attackers register them (the documented "slopsquatting" supply-chain attack, Cloud Security Alliance, April 2026)	Every new dependency is its own review question; verify it against the registry and the approved list before merge
Stub passed off as done	A function that returns a hardcoded or placeholder value, or carries a TODO, while the agent reports the task complete; tests written against the stub pass	Demand behavioural evidence, gate G3, 2.1; search the diff for placeholder returns and TODOs before believing "done"

CHAPTER CHECKLIST

01 Does the repository carry an instruction file that states conventions and non-goals for agents?

02 Pick yesterday's largest AI-assisted diff: can its approver narrate every line today?

03 Count the smells from 16.3 in your last week of merges. More than two per week means the selection bar is too low.

Evidence notes for the rules

- **For 3.2.4.** `AGENTS.md` is the common format for instruction files: more than 60,000 projects and agent tools had adopted it by December 2025, when it became a project of the Linux Foundation's Agentic AI Foundation. A 2026 study from ETH Zürich (Gloaguen and colleagues, revised September 2026) found that agents follow these files well, that the files do not generally help tasks succeed, and that repository overviews do not help; it measured inference cost rising by more than 20%. Another study (Lulla and colleagues, January 2026) found runtime and token use fell for one agent on small changes, so the evidence on cost is mixed.

- **For 3.2.5.** The best available comparison, a December 2025 study of 470 pull requests by a vendor of AI code review tools, found AI-written code carrying 1.7 times more issues overall and around 75% more logic and correctness defects. It is a single study by an interested party, so treat the figures as indicative; the direction is enough to set the floor.
- **For 3.2.3.** The approved plan is the cheapest evidence artefact the Standard produces, Ch. 11.
- **For 3.2.6.** Review is where agent volume lands. In the AIDev dataset of 456,535 agent-authored pull requests (July 2025), one developer opened 164 agent pull requests in about three days, and accepted agent pull requests had a median turnaround of 0.3 hours against 3.9 hours for human ones; the authors question whether reviews that fast reflect scrutiny.

16.4 Chapter 4: Unit tests and test data

Unit tests are the first barrier, and an agent can weaken them without anything turning red. A generated suite can reach 100% line coverage and verify almost nothing: coverage records which lines ran and says nothing about whether an assertion would fail if the code were wrong. Research on human-written suites found the same gap: coverage correlated only low to moderately with how effective a suite was (Inozemtseva and Holmes, 2014), and at Meta about half of the tests that caught faults the existing tests missed added no line coverage (Foster et al., 2025). Mutation score, the share of small deliberate code changes the suite catches, measures that, which is why 4.3 uses it in place of coverage. It still cannot show that the code was right to begin with. Tests generated from agent-written code encode what the code does, bugs included, and a 2026 study across eleven models found both measures unreliable in that case. That gap is what the human-written specification in 4.1 is for. 4.2 lists the four ways AI-written tests fail in practice.

For 4.1 The human writes the test before the agent writes the code

The industry's name for this discipline is spec-driven development, and its tooling matured quickly through 2026. Treat it as one control among several. A specification pins down acceptance criteria and cannot capture ownership, intent, or how the system behaves under load; it sets expectations before the code exists, and the gates check the result afterwards, so a team needs both.

CHAPTER CHECKLIST

01 Who wrote the spec for the last AI-implemented feature: a human first, or the model after?

02 Run mutation testing on your most AI-heavy module this week. How does its score compare with the coverage figure you report today?

03 Is `git diff tests/` part of the agent workflow?

Evidence notes for the rules

- **For 4.3.5.** A fixed percentage target invites teams to chase the number. Some mutants leave behaviour unchanged and can never be caught, which pulls the score down; mutants that time out are usually counted as caught, which pushes it up.

16.5 Chapter 5: Integration and contracts

For 5.3 Rules

ABOUT ONE ENGINEERING DAY WOULD STOP THE CHAPTER 15 SCENARIO

A CI schema check on serialised structures plus a code-ownership rule on schema, contract and fixture files takes roughly one engineering day to set up (Certance estimate). The check works mechanically, before any process or culture change.

CHAPTER CHECKLIST

01 List your serialised structures. How many have a schema under CI validation today?

02 Name one boundary where a consumer-driven contract would have caught a past incident.

03 Are schema and fixture files owner-protected in your repository?

16.6 Chapter 6: End-to-end and business journeys

CHAPTER CHECKLIST

01 Does a written, tiered journey inventory exist? Who owns it?

02 Pick one Tier 2 journey: does its test verify the data, or the page?

03 Which Tier 1 journey has no E2E test today? Write that test first next sprint.

04 Take the last change to a Tier 1 journey: does each of its Consequence 4 or 5 risks trace to a test that passed?

Evidence notes for the rules

- **For 6.2.5.** Playwright and Selenium both provide virtual authenticators for testing passkey sign-in.

16.7 Chapter 7: Non-functional quality

For 7.1 Security

Run static application security testing (SAST) and secret scanning on every change, and never waive them on AI-assisted changes. A single finding may be dismissed only with a recorded reason (false positive, used in tests, mitigated, or accepted risk), approved by the named security role. Keep the platform's bypass lists empty or limited to emergency-access accounts, and check that one dismissed finding does not approve the same finding in later changes. Veracode's GenAI Code Security Report (July 2026) found around 44% of AI-generated code samples failing its security tests. The rate has stayed flat across more than a hundred models and four rounds of testing: capability gains bought no measurable security improvement, so the gate has to hold on its own. The rule that follows: AI-authored code receives security gating at least as strict as third-party code.

For agent systems themselves, use the Open Worldwide Application Security Project (OWASP) Top 10 for Agentic Applications for 2026 (December 2025) as the scoping reference: planning, tool use, identity, memory, inter-agent communication and code execution each carry named risks. Read it with the OWASP Top 10 for Large Language Model (LLM) Applications 2026 (announced September 2026), which names Excessive Agency, an agent holding more capability than its task needs, among its risks. OWASP's GenAI Crosswalk (September 2026) maps these lists to controls in 25 frameworks, among them the EU Digital Operational Resilience Act (DORA), the international AI management system standard ISO/IEC 42001, from the International Organization for Standardization and the International Electrotechnical Commission and the AI Risk Management Framework of the US National Institute of Standards and Technology (NIST). The gates in Chapter 2 and the roster rules in Chapter 11 are the controls a team applies to the risks these lists name.

For 7.2 Performance

Keep the classical performance instruments: budgets on critical paths, load tests on Tier 1 journeys, and a baseline against which regressions are measured. Agents optimise for tests passing, which says nothing about efficiency. Dead generality, the abstractions nothing uses, Ch. 3, is a common source of avoidable performance cost. Performance assertions belong in continuous integration (CI) where the journey allows it.

For 7.3 Accessibility

Run automated accessibility checks in CI, and give every new user-interface flow a human pass. Generated user-interface code produces plausible markup with absent semantics: divs where buttons belong, labels missing, contrast untested. The automated checks catch the mechanical faults, and the human pass covers the rest. Accessible locators in end-to-end (E2E) tests (role-based, label-based) double as an accessibility signal: when a test cannot find an element by role, a screen reader cannot either.

CHAPTER CHECKLIST

01 Can a security-scan failure on an AI diff be waived in your pipeline? Remove the waiver.

02 Which Tier 1 journey has no performance budget?

03 Which Tier 1 journey has never been tested with its main dependency down?

04 Do your E2E locators double as an accessibility check, or do they bypass semantics with Cascading Style Sheets (CSS) selectors?

16.8 Chapter 8: Continuous integration gates

CHAPTER CHECKLIST

01 Which row of 8.1 exists in your pipeline today? Which is missing and cheapest to add?

02 Can an agent modify CI configuration or code-ownership files without human approval?

03 Is your review routing risk-aware, and does a minor configuration change get less reviewer attention than a schema migration?

Evidence notes for the rules

- **For 8.1, dependency check.** Assistants invent plausible package names: a 2025 USENIX Security study found that, on average, at least 5.2% of packages suggested by commercial models and 21.7% of those suggested by open-source models did not exist.
- **For 8.2.1.** GitLab Duo, Copilot review and their peers are examples of AI review assistants.
- **For 8.1, risk-tiered review routing.** GitHub's required reviewer rule, generally available since February 2026, lets review rules vary by file path.

16.9 Chapter 9: Release and production

CHAPTER CHECKLIST

01 Can you turn off yesterday's release without a deploy?

02 Are canary thresholds written down before rollouts, and does breach roll back automatically?

03 Did your last three incidents record AI involvement and the failed barrier?

16.10 Chapter 10: The operating model

CHAPTER CHECKLIST

01 Can every team member say who owns the journey inventory and who owns unit specs?

02 When did a quality engineer last improve an agent instruction file?

03 Is review time by provenance on a dashboard anyone looks at?

04 What is your change fail rate for AI-assisted changes, and for human-only ones?

Evidence notes for the rules

- **For 10.3.1.** Few teams measure review time by provenance today, which is why the rule allows a quarterly manual sample until the measure is automated.
- **For 10.3.4.** The 2025 report of Google Cloud's DevOps Research and Assessment programme found AI adoption linked to higher delivery throughput and, still, to lower delivery stability.

16.11 Chapter 11: Running the agent roster

Vendors now ship multi-agent pipelines natively: Playwright, for one, has included planner, generator and healer agents since version 1.56. This chapter treats those agents as infrastructure: agent infrastructure meets the same reliability and testing standards as any internal production system. In practice that means configuration held as code, least privilege and a named owner for every agent, 11.1, and an evaluation that every harness change must pass before it lands, 11.2.

For 11.2 Evaluating the roster

Chapter 8 gates the code an agent writes, and this section gates the agent itself. The unit under evaluation is the harness and the model together. One 2026 study ran two models through three open-source harnesses and found up to a forty-fold difference in tokens per solved task, while pass rates for the same model moved by no more than eight percentage points. The harness changes what an agent costs and how it behaves, so a score reported for a model says little about that model inside a team's own harness.

Worked example

A HEALER NEEDS RULES BEFORE IT GETS COMMIT ACCESS

A healer agent that repairs a broken locator overnight saves the team a morning. Left ungoverned, the same agent can "fix" a failing assertion by weakening it, which is reward hacking and breaks the testing rule of Chapter 1. Governed, its instructions and permissions live in the repository and confine it to page-object files, where locators live; test files, where assertions live, need a human approval, 11.1.1 and 11.1.2. Its fixes arrive as pull requests reviewed like third-party code, 11.1.3; the test-diff check in Chapter 8 flags any assertion it touched for a person to decide; and its roster owner answers for the change, 1.3.

CHAPTER CHECKLIST

01 Is every active agent's configuration in version control with an owner?

02 Which agent holds the broadest credentials, and why?

03 Could a healer weaken an assertion today without a human noticing?

04 When did the roster's evaluation set last run, and what has changed in the harness since?

Evidence notes for the rules

- **For 11.1.2.** In ImpossibleBench (October 2025, kept here as history), read-only test access kept the agents' legitimate performance while preventing attempts to modify the tests. Its value depends on the model. It worked best against Claude Opus 4.1, which cheated mainly by editing tests; GPT-5 and o3 also special-cased inputs, overloaded operators and recorded extra state, which read-only access does not stop. The incidents in Chapter 15 were access-control failures first, 15.2.
- **For 11.1.4.** For Article 17(1)(b) of the technical standard on ICT risk management under DORA, Commission Delegated Regulation (EU) 2024/1774, the evidence of independent change approval is the human approval record in A.2, which 17.3 maps; the compliance function decides what that record means for the firm.
- **For 11.1.5 and 11.1.6.** The roster inventory is evidence the compliance function can use when it decides what belongs in DORA's register of information and the asset-owner records, 17.3. Recording which lines an agent wrote is not settled yet: Cursor published Agent Trace, a vendor-neutral format for it, as a draft for comment in February 2026.
- **For 11.1.7.** Where agents run on GitHub, the enterprise audit log has marked agent actions since February 2026: each entry carries an `actor_is_agent` identifier and the user the agent acts for, and `agent_session.task` events record when sessions start, finish or fail. The agent sessions view shows

cloud agent sessions from the last 24 hours only, and the audit log keeps agentic events for 180 days. Enterprise agent definitions live in `.github/agents/`.

- **For 11.1.8.** The Open Worldwide Application Security Project (OWASP) Agent Control Standard (September 2026, a public preview at version 0.1) defines how agent platforms expose middleware hooks through which safety policies are enforced at runtime. Its reference Guardian, the component that enforces the policies, is unauthenticated and fails open by default.
- **For 11.1.9.** Isolation is the practice the field has converged on. Thoughtworks moved sandboxed execution for coding agents to Trial in volume 34 of its Technology Radar (April 2026) and calls it a sensible default. Stripe runs its coding agents on isolated machines with no real user data or production access (February 2026), and OpenAI's Codex runs with network access off by default.
- **For 11.1.10 and 11.1.11.** In August 2025 malicious versions of the Nx build package looked for AI command-line tools on developer machines and ran them with their permission-bypass flags to search for secrets; Wiz counted more than 2,000 verified secrets exposed in the first phase. A vulnerability published as CVE-2025-53773 in the Common Vulnerabilities and Exposures (CVE) list, patched in August 2025, showed the same route from inside a repository: instructions planted in source files, web pages or issues could lead an assistant to switch on its own auto-approve setting.
- **For 11.1.12.** In ImpossibleBench, giving the agent an option to abort an impossible task cut cheating on one task set from 54% to 9% for GPT-5 and from 49% to 12% for o3, with a much smaller effect for Claude Opus 4.1. Stripe gives its agents one full continuous integration run and one more local fix before the branch goes back to the engineer.
- **For 11.1.13.** The United States Department of War's AI4SDLC playbook (updated March 2026) sets six levels of agent autonomy, keeps merge authority with a person and does not recommend autonomous deployment. GitHub's cloud agent cannot approve or merge a pull request.
- **For 11.2.3.** In February 2026 OpenAI stopped reporting SWE-bench Verified after its own audit of the problems its best model could not consistently solve found material flaws in the tests or problem statements of 59.4% of them.
- **For 11.2.5 and 12.2.3.** Headline agreement rates flatter model judges: a 2026 study of 21 judge models found raw agreement overstated chance-corrected agreement by 34 to 41 percentage points on MT-Bench, one of its benchmarks. It proposes a minimum check before a judge is trusted: chance-corrected agreement, a test with the answers in swapped order, and at least three repeat runs.

16.12 Chapter 12: Testing AI inside the product

AI is already inside the products regulated firms sell. In a European Banking Authority survey published in September 2025, 55% of the banks surveyed were already using general-purpose or agentic AI in consumer-facing processes. In April 2026 the Cambridge Centre for Alternative Finance found more than 80% of the 628 financial services organisations it studied adopting AI to some level. This chapter governs those features: a support assistant, an advice tool, a model that scores credit or flags fraud. It reuses the journey tiers and scoring of Chapter 6 and the release decision of Chapter 14, and adds what behaviour that changes from one run to the next requires. Its international reference is the ISO/IEC technical specification (TS) 42119-2:2025, published in November 2025. That specification applies to AI systems, with a risk-based

approach, the 29119 software testing series that ISO/IEC publishes jointly with the Institute of Electrical and Electronics Engineers (IEEE) as ISO/IEC/IEEE 29119.

CHAPTER CHECKLIST

01 Is every AI feature in your product in the journey inventory, with its tier, model version and owner?

02 Pick your most-used AI feature: how many runs does its release criterion require, and who set the threshold?

03 When the model behind it last changed, which evaluation set ran before traffic moved?

Evidence notes for the rules

- **For 12.2.4.** The International Software Testing Qualifications Board (ISTQB) lists metamorphic, back-to-back, adversarial and drift testing in version 2.0 of its AI testing syllabus (CT-AI), released in April 2026.
- **For 12.3.** The Generative AI Profile of the US National Institute of Standards and Technology (NIST AI 600-1) asks firms to "establish minimum thresholds for performance or assurance criteria and review as part of deployment approval", GV-1.3-002.
- **For 12.4.** NIST's profile lists red teaming against prompt injection, data poisoning and model extraction among its measures, MS-2.7-007, and asks that those who run structured human feedback exercises are not directly involved in developing the system, MS-1.3-003. A dedicated ISO/IEC technical specification on red teaming, 42119-7, is in development.

16.13 Chapter 13: Evidence for supervisors

Evidence notes for the rules

- **For 13.1.2.** Article 12(2) of the technical standard requires the firm to set a retention period for logs; it is cited for logs only, and the compliance function decides which rule governs test evidence, 17.2.
- **For 13.1.5.** "Where feasible" is in the text of DORA Article 25(1).
- **For 13.1.6.** The UK statement of 15 May 2026 expects firms to "triage, prioritise, risk assess, and remediate vulnerabilities more quickly, more frequently, and at scale, including through automation where appropriate, while mitigating the operational risks from doing so".
- **For 13.1.7.** The European statement of 31 July 2026 warns that highly capable models "can analyse leaked or exposed source code to identify vulnerabilities, backdoors, or proprietary algorithms".
- **For 13.1.8.** Both statements extend to third parties and open-source software. The UK statement expects firms to "identify, monitor, and manage external applications, libraries, and services integrated into their networks".
- **For 13.1.9.** Under the revised Product Liability Directive, a product is presumed defective where the defendant fails to disclose relevant evidence, Article 10(2), 17.6.
- **For 13.1.10.** Cyber Resilience Act reporting starts with an early warning within 24 hours of a manufacturer becoming aware of an actively exploited vulnerability, 17.6.

17 The regulated overlay

In the instruments and supervisory statements reviewed for this edition, as at 11 October 2026, no EU or UK financial regulator has written a binding rule about how code is written when an agent writes it. There is no AI clause in the change-management obligations, no separate pathway for machine-authored changes and no requirement on the firm to record which lines a model produced, 17.2. The obligations reviewed are drafted in terms of changes and systems and make no distinction by authorship. A firm that cannot evidence controlled change over AI-assisted work therefore risks a finding against the change-management requirement of the Digital Operational Resilience Act (DORA), which has applied since 17 January 2025; the compliance function assesses that exposure for the firm, 17.1.

This chapter is written for teams inside the EU and UK regulatory perimeter. Every obligation it cites is taken from the instrument itself, and where an obligation is qualified, the qualifier is kept.

17.1 What applies

DORA Article 9(4)(e) is the provision this chapter starts from, because it names changes to software. It requires documented policies, procedures and controls for information and communication technology (ICT) change management, expressly including changes to software, to ensure that all changes to ICT systems are "recorded, tested, assessed, approved, implemented and verified in a controlled manner". Six verbs, in application across the EU since 17 January 2025. Read against an agent-written pull request that touches code, a fixture and a schema, the verbs map onto the gates of Chapter 2, as 17.3 sets out.

The technical standard underneath it goes further. Commission Delegated Regulation (EU) 2024/1774, Article 16(3), requires "source code reviews covering both static and dynamic testing", with vulnerability identification, action plans and monitoring. Unlike DORA Article 25(1), Article 16(3) carries no feasibility qualifier; Article 16(8) applies one only to source code from ICT third-party service providers and open-source projects. Article 16(2) requires testing and approval of ICT systems before use and after maintenance. Article 16(4) requires security testing of software packages no later than integration. Article 17(1) requires change-management procedures with roles that ensure changes are "tested and finalised in a controlled manner" with "an effective quality assurance" (point (c)), identified fall-back procedures (point (e)) and, under point (b) and most relevant to an agent roster, "mechanisms to ensure the independence of the functions that approve changes and the functions responsible for requesting and implementing those changes".

DORA Article 24 supplies the programme around it. It requires at least yearly testing, by financial entities other than microenterprises, of all ICT systems and applications supporting critical or important functions (24(6)); tests undertaken by independent parties, whether internal or external (24(4)); and procedures to prioritise, classify and remedy all issues the tests reveal, with internal validation that they are fully addressed (24(5)). **Article 25(1)** lists the eligible methods and includes "source code reviews where feasible" among them. Quote it with the qualifier attached; it is a menu of methods, and it sets no frequency of its own.

In the UK the shape differs and the evidential demand is sharper. Under the operational resilience rules of the Financial Conduct Authority (FCA), in chapter 15A of its Senior Management Arrangements, Systems and Controls sourcebook (SYSC), a firm must set an impact tolerance for each important business service (SYSC 15A.2.5R) and carry out scenario testing of its ability to stay within that tolerance under severe but

plausible disruption (SYSC 15A.5.3R). It must also make and keep up to date a written record of its assessment of its compliance (SYSC 15A.6.1R), retain each version for at least six years and provide it to the FCA on request (SYSC 15A.6.2R). Firms had until 31 March 2025 to be able to remain within their impact tolerances. Nothing in the regime mentions AI, development methodology or code provenance, so its requirements apply whatever produced the code behind an important business service.

One AI-specific obligation may reach the firm, in softened form; the compliance function decides whether it does. Article 4 of the EU AI Act requires providers and deployers to take measures to support the AI literacy of staff operating AI systems on their behalf. It has applied since 2 February 2025, originally as a duty to ensure a sufficient level of literacy; the Digital Omnibus, Regulation (EU) 2026/1744, in force since 27 July 2026, lowered it to support and states that no specific level of literacy is required. Whether a firm whose developers use a coding assistant is a deployer under the Act is for the compliance function to decide. Chapter 10's failure-mode training is one engineering measure towards it, and a dated training record is the evidence that the measures were taken; whether they are sufficient is for the compliance function to decide.

Supervisors addressed frontier AI in 2026 through the obligations that already apply. The Bank of England, the FCA and His Majesty's Treasury (HM Treasury) said in a joint statement on 15 May 2026 that "Frontier AI models represent a step-change in capability, with significant implications for cyber security and operational resilience". The European Supervisory Authorities' joint statement of 31 July 2026 (JC 2026 25) says the DORA requirements "remain highly relevant, through the implementation of the ICT risk management framework, testing, incident and recovery management and ICT third party risk management". On 7 July 2026 the European Central Bank asked the significant institutions it supervises directly for action plans on AI-enabled cyber threats, due by 31 October 2026. None of these creates a new rule, and none addresses AI coding assistants: they treat frontier AI mainly as a tool in an attacker's hands. Three of their themes reach engineering directly, and 13.1 turns them into rules: fixing vulnerabilities at scale, protecting source code, and the software supply chain.

17.2 Claims to test before repeating them

Many regulatory claims made about AI coding tools overstate what the instruments require, and a Chief Risk Officer who catches one overstatement discounts everything that follows it. Each item below gives what the instrument's text says and the question the compliance function answers before the firm relies on it.

- **Is a coding assistant a high-risk AI system?** Annex III of the AI Act lists eight categories of high-risk use, and software development, information technology operations and developer productivity are not among them. Articles 12 (logging), 14 (human oversight) and 17 (quality management) attach to high-risk systems. The compliance function decides whether any tool the firm uses falls within an Annex III category. Article 14's language on automation bias maps closely onto code review; use it as an analogy, labelled as one.
- **Must the firm mark AI-generated source code?** Article 50(2), in application since 2 August 2026, obliges providers of AI systems, including general-purpose AI systems, that generate synthetic audio, image, video or text to mark their output in machine-readable form; providers whose systems were placed on the market before that date have until 2 December 2026 (Article 111(4), added by Regulation (EU) 2026/1744). No official guidance says whether source code counts as synthetic content, and the text places the duty on providers. Article 50's deployer duties name emotion recognition, biometric categorisation, deep fakes and certain text published to inform the public. The compliance function decides whether any of these reaches the firm. Provenance in Chapter 2 is an engineering control that also produces evidence, and it should be presented as exactly that.
- **Must the firm evidence which code an AI wrote?** The instruments reviewed for this edition, DORA and its technical standards, the EU AI Act, and the rules of the FCA and the Prudential Regulation Authority, contain no such requirement as at the edition date. Put any claim to the contrary to the compliance function, together with the article it cites.

- **Is the FCA writing AI-specific rules?** The FCA has said it is not: its published approach relies on existing frameworks and is outcomes-based. Its AI Live Testing programme addresses AI deployed in UK financial markets; the compliance function decides whether any internal tool falls within it.
- **How much weight does BaFin's guidance carry?** BaFin, Germany's Federal Financial Supervisory Authority, published guidance on ICT risks in the use of AI on 18 December 2025. In its section on software development it states that the same rules apply to all code, whether a person or a machine writes it; it points to static code analysis to find AI functions a developer did not know were called; and it warns that open-source libraries can bring in malicious code and unfixed vulnerabilities. BaFin describes the guidance as non-binding. Treat it as a published statement of supervisory attention on this subject and as a signal of where questions will come from, and describe it with the word "flagged".
- **Does a published standard give presumption of conformity under the AI Act?** EN 18286:2026, the European standard on quality management systems for EU AI Act regulatory purposes, was approved in June and published in July 2026 by the European Committee for Standardization (CEN) and the European Committee for Electrotechnical Standardization (CENELEC). As at 11 October 2026 no reference to it had been published in the Official Journal; publication and citation are separate acts. A certificate against ISO/IEC 42001, the international AI management system standard, is real evidence of a systematic approach; the compliance function decides what weight it carries under the AI Act.
- **How long must testing evidence be kept?** The articles of DORA this chapter cites set no period for testing evidence. The technical standard (Article 12(2)) requires the firm to set its own retention period for logs, taking into account its business and information-security objectives, the reason each event is logged, and the results of its ICT risk assessment. The six-year figure is a UK rule, and it applies to the operational-resilience self-assessment. The compliance function decides which retention rule the firm applies to gate evidence; 13.1 sets the engineering default.
- **Who does Article 25(2) bind?** Its pre-deployment vulnerability assessment is addressed to central securities depositories and central counterparties; the compliance function confirms whether the firm is either.

17.3 The map: what each gate is evidence of

Each row names the obligation a gate's evidence relates to, what can be handed over, and the type of evidence it is. In every row, the compliance function decides whether the evidence meets the obligation for the firm.

GATE	OBLIGATION IT RELATES TO	WHAT CAN BE HANDED OVER	EVIDENCE TYPE
G1 · Provenance	DORA Art. 9(4)(e) "recorded"; Art. 8(3) risk assessment on major change	A change record that states authorship and tooling, making the population of AI-assisted changes queryable at all. No rule cited here names AI authorship; the record shows a supervisor how the firm chose which changes to examine	**Change record: authorship and tooling** Verdict: compliance function
G2 · Test integrity	RTS (the technical standard, Commission Delegated Regulation (EU) 2024/1774) Art. 17(1)(b) independence of approving and implementing functions	Evidence that the party proposing a change did not silently move the control that verifies it. G2 shows the proposer did not move the verifying control; the named approver, A.2, is the evidence offered for independence	**Change record: test-file integrity** Verdict: compliance function
G3 · Behavioural evidence	DORA Art. 9(4)(e) "tested… and verified"; RTS Art. 16(2) testing and approval before use	Per-change proof of behaviour beyond pipeline status, attributable to a release	**Test evidence per change, tied to a build** Verdict: compliance function
G4 · Contract check	RTS Art. 16(3) static and dynamic source code review; Art. 16(2)	Automated verification that an interface change was validated against its consumers before merge. Article 16(3) also expects vulnerabilities and anomalies in the source code to be identified and acted on	**Interface test evidence** Verdict: compliance function
G5 · Blast-radius match	DORA Art. 9(4)(e) "assessed… in a controlled manner"; Art. 8(3)	Evidence that scope was assessed against the change's stated intent	**Change record: scope against intent** Verdict: compliance function

GATE	OBLIGATION IT RELATES TO	WHAT CAN BE HANDED OVER	EVIDENCE TYPE
Journey inventory, Ch. 6	DORA Art. 8 identification and classification; Art. 24(6) yearly testing of systems supporting critical or important functions; SYSC 15A.2.5R impact tolerances	A tiered map of business-critical journeys to the tests that cover them, and an honest statement of what "covered" asserts	**Register: tiered journeys mapped to their tests** Verdict: compliance function
Exception log, Ch. 2	DORA Art. 6(5) documented and reviewed framework; Art. 24(5) prioritise, remediate, validate; RTS Art. 17(1)(g) emergency changes approved after implementation	A dated register of where the Standard was not met, who approved it, and what happened next	**Register: deviations, approvers and follow-up** Verdict: compliance function
Agent roster, Ch. 11	DORA Arts. 3(19) and 3(21) definitions; Art. 5(2)(c) clear roles and responsibilities for ICT-related functions; Art. 28(3) register of information; Art. 30(2) contractual terms; RTS Art. 4(2)(b)(iv) identity of ICT asset owners	An inventory of which agents exist, what each may touch, who owns it, and which vendor supplies it, with the evaluation record for every change to its harness. Two questions go to the compliance function: whether a coding assistant is an ICT service that belongs in the register of information, and whether each agent's owner is the ICT asset owner the technical standard requires the firm to record	**Inventory: agents, permissions, owners and vendors** Verdict: compliance function

Whether the enhanced contractual provisions of Article 30(3), with their audit rights and exit strategies, apply to a coding tool depends on whether the tool supports a critical or important function. That follows from the firm's own criticality assessment, which the compliance function reads for each tool.

THE BAR ROSE WITHOUT A RULE CHANGING

Every change-management and testing obligation above was drafted before agents wrote production code, and none of them distinguishes by who or what wrote the code. A control designed for a few hundred human-authored changes a quarter now has to hold against machine-scale volume. "Show me that the system that failed was tested for the thing that broke" is as easy to ask as before and takes considerably more evidence to answer.

17.4 AI inside the product: the AI Act and model risk

- **Is the feature a high-risk AI system under the AI Act?** Annex III lists the creditworthiness assessment and credit scoring of natural persons among high-risk uses, as the European Banking Authority's note of November 2025 on the Act's implications for banking records. Recital 40 of the Digital Omnibus, Regulation (EU) 2026/1744, states that the date of application of Sections 1, 2 and 3 of Chapter III, the high-risk rules, is set to 2 December 2027 for Annex III systems; the operative date sits in the amended Article 113. The compliance function decides which features fall within Annex III and what the firm owes as provider or deployer.
- **Does the firm's model risk framework cover it?** In the UK, supervisory statement SS1/23 of the Prudential Regulation Authority (PRA) sets model risk management principles for UK banks, building societies and PRA-designated investment firms with internal model approval to calculate regulatory capital requirements. It took effect on 17 May 2024 and was reissued in April 2026, and the accompanying policy statement PS6/23 says the principles are sufficient to manage the risks of AI and machine learning models. The model risk function decides whether a feature is a model under the firm's framework; the evidence in this chapter is what it validates.

The regulatory conclusion belongs to the compliance function. Engineering can provide an evidence assessment of the release process and its test records to support it.

17.5 Test data and accessibility

The technical standard sets a floor for non-production environments. Commission Delegated Regulation (EU) 2024/1774, Article 16(5)(a), provides that non-production environments only store anonymised, pseudonymised or randomised production data, and Article 16(6) allows production data to be stored only for specific testing occasions, for limited periods of time, after approval by the relevant function and with those occasions reported to the information and communication technology (ICT) risk management function (text checked on the European Union law portal, EUR-Lex, on 11 October 2026). The Standard's default is stricter, because pseudonymised payment data can still identify customers. The compliance function decides what the firm's own procedure must contain.

In the EU, the European Accessibility Act (Directive (EU) 2019/882) covers consumer banking services provided to consumers after 28 June 2025 (Article 2(2)(d)). The compliance function decides what it requires of the firm; these checks and their records are the engineering evidence.

17.6 Product liability and cyber resilience

Two EU instruments make test and release records evidence that a court or an authority can ask for. Both are stated here as what the text says; the compliance and legal functions decide whether and how each applies to the firm.

The revised Product Liability Directive. Directive (EU) 2024/2853 applies to products placed on the market or put into service after 9 December 2026, Article 2(1), and its definition of a product includes software, Article 4(1). A court can require a defendant to disclose relevant evidence at its disposal, Article 9(1), and a product is presumed defective where the defendant fails to disclose it, Article 10(2). The exemption for a defect that appeared after the product was placed on the market does not apply where the defect comes from software, including its updates, or from missing safety updates, provided these are within the manufacturer's control, Article 11(2). The engineering evidence that bears on these articles is the record set in A.2: gate records, release records and the regression results for every update, kept tamper-evident and retrievable by release, 13.1.9.

The Cyber Resilience Act. Regulation (EU) 2024/2847 sets cybersecurity requirements for products with digital elements. Manufacturers report actively exploited vulnerabilities and severe incidents through a single reporting platform run by the European Union Agency for Cybersecurity (ENISA), which the European

Commission states has been operational since 11 September 2026: an early warning within 24 hours of becoming aware, a notification within 72 hours, and a final report within 14 days of a corrective measure becoming available for a vulnerability, or within one month for a severe incident. The Act's other obligations apply from the dates in Article 71(2). The engineering evidence that supports a report inside 24 hours is the software bill of materials, 8.1, and the trace from a component to every release that contains it, 13.1.10.

CHAPTER CHECKLIST

01 For each of G1 to G5, what artefact survives the merge, where does it live, and for how long?

02 If a supervisor asked today which of last quarter's changes were AI-assisted, could the question be answered at all?

03 Has the compliance function decided whether your coding-assistant vendor belongs in the register of information?

04 Does your journey inventory line up with the important business services or critical or important functions your firm has already declared? Where the two lists disagree, reconcile them with the operational resilience owner.

This chapter is engineering guidance. It gives no legal advice and no compliance opinion. Regulatory positions should be verified with the compliance and legal functions. Article numbers, quotations and dates were checked against the Official Journal text, the FCA Handbook and the 2026 supervisory statements on 1 October 2026, and those this edition changed again on 11 October 2026; re-check them before use in a contractual or supervisory context.

PART X Adopting the Standard

18 Adopting the Standard

18.1 Adoption plan

Sequenced for a single team; dates anchor to adoption of the Standard.

PHASE	TIMELINE	WHAT SHIPS	DONE WHEN
1 · Stop the incident class	Week 1	G1 provenance field in the pull-request template (no AI, AI-assisted, agent-authored); schema validation in CI, the G4 minimum; protected paths on fixtures, schemas, migrations; security scanning on every change, non-waivable on AI-assisted changes; exception log created; coding agents run in an isolated environment with approval-bypass modes switched off, 11.1.9 and 11.1.10	Every merged pull request carries provenance; a code-plus-schema change cannot merge without schema validation or a recorded waiver, and schema files need code-owner approval
2 · Make changes accountable	Weeks 2 to 4	G1 detail (tool and parts touched) checked in review; G2, G3, G5 enforced in review; consumer-driven contracts set up where consumers are known, then the full G4 check enforced; risk-tiered routing configured	First monthly exception-log review held, led by a reviewer outside engineering
3 · Measure and harden	Weeks 5 to 12	Mutation gate on changed code switched on, AI-heavy modules first and every module by week 12; anti-rubber-stamp metric on a dashboard; test-first workflow for AI-implemented changes; failure-mode training delivered	Baseline metrics exist and the engineering lead reviews them monthly
90-day checkpoint	Week 13	Review against evidence: exception patterns, review-time trend, mutation trend, AI-attributed incidents since adoption	Standard revised on evidence

18.2 Mapping to NIST's Secure Software Development Framework

Security teams often map their development controls to the Secure Software Development Framework (SSDF) of the US National Institute of Standards and Technology (NIST), Special Publication (SP) 800-218, version 1.1 (February 2022). Version 1.2 was published as a draft for comment on 17 December 2025 and adds two practices, PO.6 (continuous improvement) and PS.4 (robust and reliable updates); a May 2026 NIST presentation numbers the updates practice PW.10. Version 1.2 was not final on 1 October 2026, so the table uses the version 1.1 practice numbers, which the draft keeps. NIST's profile for generative AI (SP 800-218A, July 2024) states that its practices "do not distinguish between human-written and AI-generated source code". The scanning rules in 7.1 run on every change, whoever wrote it, and add one AI-specific condition: they cannot be waived on AI-assisted changes. G1 Provenance goes further on purpose: the SSDF records where components come from, and has no control for which code an AI wrote. The mapping is Certance's reading.

STANDARD	SSDF 1.1 PRACTICE
Gates, waivers and the exception log, Ch. 2 and A.2	PO.4 Define and use criteria for software security checks
Operating model and named approvers, Ch. 10 and A.2	PO.2 Implement roles and responsibilities
Agent roster: configuration as code, least privilege, pinned models, 11.1	PO.3 Implement supporting toolchains; PO.5 Implement and maintain secure environments for software development
G1 · Provenance	PS.3 Archive and protect each software release (component provenance only; AI authorship is a Certance addition)
G2 · Test integrity and protected paths, 8.1	PS.1 Protect all forms of code from unauthorised access and tampering; PW.8 Test executable code to identify vulnerabilities and verify compliance with security requirements
G3 · Behavioural evidence and G4 · Contract check	PW.8 Test executable code to identify vulnerabilities and verify compliance with security requirements
G5 · Blast-radius match, review, security scans, 7.1	PW.7 Review and/or analyse human-readable code
Dependency check, 8.1	PW.4 Reuse existing, well-secured software
Incident AI-attribution and post-incident review, 9.2 and 9.3	RV.2 Assess, prioritise and remediate vulnerabilities; RV.3 Analyse vulnerabilities to identify their root causes

G Glossary

Terms are linked at their first use in the running text of the chapters. Each entry links back to that place.

TERM	MEANING IN THIS STANDARD
Agent roster Back to text	The set of AI agents a team runs. Each has a named owner, a pinned model version, its configuration in version control and an evaluation set, Ch. 11.
AGENTS.md and instruction files Back to text	Plain-text files in the repository that state conventions and non-goals for agents, read before every task. AGENTS.md is the most widely adopted format, 3.2.
AI-assisted change Back to text	A change in which an AI tool wrote or restructured any code, test or data. Spelling fixes, identifier completion, mechanical renames and reformatting do not count, 2.1.
AI feature Back to text	Any part of the product whose behaviour comes from a trained model: generated text, a score, a ranking or a decision, 12.1.
Artefact inventory Back to text, p. 4	The list of every file, schema, configuration and dependency a change touches. Under G5, Blast-radius match, it is compared with the change's stated intent.
Automation bias Back to text, p. 5	The tendency to trust output because a machine produced it, including an AI reviewer's approval.
Background agent Back to text	An agent that works unattended in a cloud environment and returns a finished pull request.
Behavioural evidence Back to text	Proof of what a change does beyond a passing pipeline: run output, a trace, a before-and-after demonstration, or a new test that fails on the old code (G3).
Blast radius Back to text	How much of a system a change can affect. G5 checks that it matches what the change claims to do.
Canary release Back to text	Releasing a change to a small share of traffic first, with thresholds set in advance that roll it back automatically when breached, Ch. 9.
Consumer-driven contract Back to text	A test in which each consumer of an interface records the fields and values it relies on, and the provider's build checks every change against those records. Pact is the common open-source tool, 5.2.
Continuous integration (CI) Back to text, p. 4	The automated build and checks that run on every change before it merges.
DORA Back to text, p. 12	In this Standard, the EU Digital Operational Resilience Act, Regulation (EU) 2022/2554, applied since 17 January 2025. Google Cloud's DORA research programme is unrelated and is always identified with Google Cloud.
End-to-end (E2E) test Back to text	A test that drives a whole business journey through the real interface, from the user's action to the recorded outcome, Ch. 6.
Evaluation set Back to text	The test suite of an AI feature or an agent: inputs, the expected behaviour or grading rule for each, and the quality each case checks. Held as a protected path, 11.2 and 12.2.

TERM	MEANING IN THIS STANDARD
Exception log Back to text	The single register of every waived gate, dismissed security finding and emergency change, with its reason, approver and date. A reviewer outside engineering, such as the risk function, leads its monthly review, A.2.
Exploratory testing Back to text	Time-boxed testing by a person against a written charter, in which each test is designed from what the last one showed, 6.2.
Feature flag Back to text	A switch that turns a change on or off in production without a new deployment.
Harness Back to text, p. 4	Everything around the model in an agent: its instructions, tools, permissions and the loop that runs it. The harness and the model are evaluated together, 11.2.
Healer agent Back to text	An agent that repairs tests broken by application changes, for example by updating how a test finds an element on a page, 16.11.
ICT Back to text, p. 22	Information and communication technology: the term DORA uses for systems, networks and digital services.
Least privilege Back to text, p. 5	Each agent or person holds only the access its current task needs, 11.1.
MCP (Model Context Protocol) Back to text, p. 4	The open standard that connects an agent to databases, ticket systems and other tools.
Model drift Back to text	A change over time in how an AI feature behaves or in what it is asked, found by monitoring live outputs against the evaluation rules, 12.5.
Mutation score Back to text	The share of small deliberate changes to the code (mutants) that the test suite catches. Chapter 4 uses it in place of line coverage.
Pass^k Back to text	A pass rule over repeated runs: a case passes only when all k runs pass. The default for Tier 1 AI features, 12.3.
Prompt injection Back to text	An attack that hides instructions in text an AI feature or agent reads, such as a document or web page, to make it act against its own instructions, 12.4.
Provenance Back to text	The record of whether and where AI assisted a change, which tool, and which parts it touched (G1).
Red teaming Back to text	Structured adversarial testing by people outside the building team, who try to make an AI feature leak data, break its rules or exceed its permissions, 12.4.
Reward hacking Back to text, p. 5	An agent reaching the visible goal, such as passing tests, by defeating its purpose, for example by weakening a test, 15.1.
RTS Back to text, p. 22	Commission Delegated Regulation (EU) 2024/1774, the regulatory technical standard on ICT risk management under DORA. The text also calls it the technical standard.
SAST Back to text, p. 11	Static application security testing: automated checks of source code for known vulnerability patterns, 7.1.
SBOM Back to text	Software bill of materials: the list of every component and version in a release build, kept with a signed attestation of how the build was made, 8.1.
Slopsquatting Back to text, p. 9	Registering a package name that AI assistants tend to invent, loaded with malicious code, so an agent that installs it pulls in the attack, 8.1.

TERM	MEANING IN THIS STANDARD
Tier 1 journey Back to text	A business journey whose failure causes direct financial loss or a regulatory breach: money movement, transaction execution, credit and lending decisions, regulatory reporting, authentication, 6.1.
Waiver Back to text	A recorded decision to merge a change that has not passed G2, G3, G4 or G5, made by the named approver, who neither wrote nor requested the change, Ch. 2 and A.2.

S Sources and references

Entries marked "history" are older sources kept for context.

Regulatory instruments

- Regulation (EU) 2022/2554, the Digital Operational Resilience Act (DORA), 14 December 2022 · OJ L 333, 27.12.2022 · eur-lex.europa.eu
- Commission Delegated Regulation (EU) 2024/1774, the technical standard on the ICT risk management framework, 13 March 2024 · OJ L, 25.6.2024 · eur-lex.europa.eu
- Regulation (EU) 2024/1689, the Artificial Intelligence Act, 13 June 2024 · OJ L, 12.7.2024 · eur-lex.europa.eu
- Regulation (EU) 2026/1744, the Digital Omnibus on AI, amending the AI Act, 8 July 2026 · OJ L, 24.7.2026, in force 27 July 2026 · eur-lex.europa.eu
- Directive (EU) 2019/882, the European Accessibility Act, 17 April 2019 · scope in Article 2(2), consumer banking services in point (d) · eur-lex.europa.eu
- Financial Conduct Authority Handbook, SYSC 15A Operational resilience, last updated 5 April 2024 · handbook.fca.org.uk
- Directive (EU) 2024/2853 on liability for defective products, 23 October 2024 · OJ L, 18.11.2024 · Articles 2(1), 4(1), 9(1), 10(2) and 11(2) · eur-lex.europa.eu
- Regulation (EU) 2024/2847, the Cyber Resilience Act, 23 October 2024 · OJ L, 20.11.2024 · eur-lex.europa.eu

Supervisory statements and guidance

- Prudential Regulation Authority, SS1/23 *Model risk management principles for banks*, May 2023, in effect 17 May 2024, reissued April 2026; and PS6/23, 17 May 2023 · bankofengland.co.uk
- European Banking Authority, *AI Act implications for the EU banking sector*, 21 November 2025 · eba.europa.eu
- Bank of England, Financial Conduct Authority and HM Treasury, *Joint statement on frontier AI models and cyber resilience*, 15 May 2026 · bankofengland.co.uk
- European Supervisory Authorities, *ESA Statement: Toward a consistent and risk-based approach for ICT risks from frontier AI models*, JC 2026 25, 31 July 2026 · esma.europa.eu

- European Central Bank Banking Supervision, *Addressing AI-enabled cybersecurity threats*, letter to significant institutions, SSM-2026-0301, 7 July 2026 · bankingsupervision.europa.eu
- BaFin, Germany's Federal Financial Supervisory Authority, *Guidance on ICT Risks in the Use of AI at Financial Entities*, 18 December 2025 (German), 30 January 2026 (English) · bafin.de
- Financial Conduct Authority, *AI and the FCA: our approach*, last updated 2 October 2026 · fca.org.uk; *FS25/5: AI Live Testing*, updated 16 March 2026 · fca.org.uk
- CEN and CENELEC, *EN 18286:2026*, quality management systems for EU AI Act regulatory purposes, published July 2026 · cencenelec.eu
- European Commission, *Cyber Resilience Act: reporting obligations* · the single reporting platform, operational since 11 September 2026 · digital-strategy.ec.europa.eu

Industry research and reports

- European Banking Authority, *Rising application of AI in EU banking and payments sector*, 25 September 2025 · eba.europa.eu
- Cambridge Centre for Alternative Finance, *2026 Global AI in Financial Services Report*, 28 April 2026 · 628 organisations in 151 jurisdictions · jbs.cam.ac.uk
- DORA (Google Cloud), *DORA's software delivery performance metrics*, updated 5 January 2026 · five measures, including deployment rework rate · dora.dev
- DORA (Google Cloud), *State of AI-assisted Software Development*, September 2025 (history) · dora.dev
- Centre for Long-Term Resilience, *Scheming in the Wild*, 27 March 2026 (methodology), and *AI loss of control incidents are worsening*, insight report, 28 August 2026 (current data: 1,664 incidents in 2026 to 9 August). Method: open-source intelligence over social-media posts with model-assisted classification and manual spot-checks · longtermresilience.org
- UK AI Security Institute, *Incident report: unsanctioned agent behaviour during cyber testing*, 4 August 2026 · a controlled evaluation with internet access deliberately enabled; no real-world harm reported · aisi.gov.uk
- Veracode, *GenAI Code Security Report*, 28 July 2026 · veracode.com
- GitClear, *The Maintainability Gap: AI Code Quality in 2026*, 2026 · gitclear.com
- CodeRabbit, *State of AI vs Human Code Generation*, 17 December 2025 · single study, n=470 pull requests, published by a vendor of AI code review
- The Register, coverage of the PocketOS agent database deletion, 27 April 2026 · theregister.com
- Li, Zhang and Hassan, *The Rise of AI Teammates in Software Engineering (SE) 3.0*, July 2025 · the AIDev dataset of 456,535 agent-authored pull requests · arxiv.org/abs/2507.15003

Vendor and engineering-practice publications

- The Linux kernel documentation, *Kernel guidelines for tool-generated content* · trivial tool use outside the disclosure scope · docs.kernel.org
- The Linux kernel documentation, *AI coding assistants* · the Assisted-by: tag; only humans add Signed-off-by · docs.kernel.org
- GitHub, *Risks and mitigations for GitHub Copilot cloud agent* · commit co-authorship; the requester cannot approve the agent's pull request · docs.github.com

- Playwright release notes (Test Agents from v1.56; agentic features through v1.62.1, 30 July 2026) · playwright.dev/docs/release-notes
- GitHub, *Enterprise AI Controls & agent control plane now generally available*, 26 February 2026 · agent audit-log fields and protected custom agent definitions · github.blog
- Linux Foundation, *Linux Foundation Announces the Formation of the Agentic AI Foundation (AAIF)*, 9 December 2025 · AGENTS.md contributed; adopted by more than 60,000 projects · linuxfoundation.org
- GitHub, *Required reviewer rule is now generally available*, 17 February 2026 · reviewer rules by file path · github.blog
- Playwright, *Credentials* (virtual WebAuthn authenticator), version 1.61 · playwright.dev; Selenium, *Virtual Authenticator* · selenium.dev; both read 11 October 2026
- Sundar Pichai, post on the Google blog from Google Cloud Next 2026, 22 April 2026 · 75% of all new code AI-generated and approved by engineers; no measurement method stated · blog.google
- Google Research, *AI in software engineering at Google: progress and the path ahead*, 6 June 2024 (history) · the accepted-characters measure · research.google
- Spotify Engineering, *Coding Is No Longer the Constraint: Scaling Developer Experience to Teams and Agents at Spotify*, 3 June 2026 · engineering.atspotify.com
- Thoughtworks, *Technology Radar* volume 34, *Sandboxed execution for coding agents*, 15 April 2026 · Trial · thoughtworks.com
- Stripe, *Minions: Stripe's one-shot, end-to-end coding agents*, part 2, 19 February 2026 · isolated machines, retry cap and hand-back · stripe.dev
- OpenAI, *Codex: agent approvals and security* · network access off by default · learn.chatgpt.com
- United States Department of War, Chief Information Officer, AI4SDLC Working Group, *AI Workflow Design and Governance* play, updated 15 March 2026 · code.mil

Agent evaluation

- Yao, Shinn, Razavi and Narasimhan (Sierra), *τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains*, June 2024 (history) · introduces pass^k, the chance that all k trials of a task succeed · arxiv.org
- Hanley and Lippman-Hand, *If nothing goes wrong, is everything all right? Interpreting zero numerators*, JAMA, 1983 (history) · the rule of three: no events in n trials puts the 95% upper bound near 3 in n
- Zhong, Raghunathan and Carlini, *ImpossibleBench: Measuring LLMs' Propensity of Exploiting Test Cases*, 23 October 2025 · cheating rates on tasks whose tests conflict with the specification; hidden and read-only tests as mitigations · arxiv.org
- METR (Von Arx, Chan and Barnes), *Recent Frontier Models Are Reward Hacking*, 5 June 2025 · the "Please do not cheat" result is 20 runs of o3 on one task · metr.org
- Vats and Golev, *The Scaffold Effect in Coding Agents: Harness Choice as a Hidden Variable in Coding-Agent Evaluation*, July 2026 · two models, three harnesses, 50 tasks · arxiv.org/abs/2607.22585
- OpenAI, *Why SWE-bench Verified no longer measures frontier coding capabilities*, 23 February 2026 · openai.com
- Cursor, *Agent Trace*, vendor-neutral format for recording AI contributions to code, released as a draft for comment, February 2026 · infoq.com

- Norman, Rivera and Hughes, *Reliability without Validity: A Systematic, Large-Scale Evaluation of LLM-as-a-Judge Models*, June 2026 · 21 judges, about 541,000 judgements · arxiv.org/abs/2606.19544
- Gloaguen, Mündler, Müller, Raychev and Vechev (ETH Zürich), *Evaluating AGENTS.md: Are Repository-Level Context Files Helpful for Coding Agents?*, February 2026, revised September 2026 · arxiv.org/abs/2602.11988
- Lulla et al., *On the Impact of AGENTS.md Files on the Efficiency of AI Coding Agents*, January 2026, revised March 2026 · 124 pull requests, 10 repositories · arxiv.org/abs/2601.20404

Security and testing practice

- ISO/IEC TS 42119-2:2025, *Artificial intelligence: Testing of AI, Part 2: Overview of testing AI systems*, 3 November 2025 · iso.org; ISO/IEC AWI TS 42119-7, *Red teaming*, an approved work item under development · iso.org
- International Software Testing Qualifications Board, AI Testing syllabus (CT-AI) version 2.0, April 2026 · istqb.org
- NIST, AI 600-1, *Artificial Intelligence Risk Management Framework: Generative Artificial Intelligence Profile*, July 2024 (history) · actions GV-1.3-002, MS-1.3-003, MS-2.7-007 · nvlpubs.nist.gov
- Spracklen, Wijewickrama, Sakib, Maiti, Viswanath and Jadliwala, *We Have a Package for You! A Comprehensive Analysis of Package Hallucinations by Code Generating LLMs*, USENIX Security 2025 · 576,000 samples, 16 models · usenix.org
- OWASP GenAI Security Project, *OWASP Top 10 for Agentic Applications for 2026*, 9 December 2025 · genai.owasp.org
- OWASP GenAI Security Project, *OWASP Top 10 for LLM Applications 2026*, announced on 1 September 2026 with the *Agent Control Standard* (public preview, version 0.1) and the *GenAI Security Industry Framework Crosswalk* · genai.owasp.org
- NIST, SP 800-218, *Secure Software Development Framework (SSDF) Version 1.1*, February 2022 (history), and SP 800-218r1 initial public draft, *SSDF Version 1.2*, 17 December 2025 · csrc.nist.gov · and NIST, *SSDF update and DevSecOps project*, presentation, 20 May 2026 · csrc.nist.gov
- NIST, SP 800-218A, *Secure Software Development Practices for Generative AI and Dual-Use Foundation Models: An SSDF Community Profile*, July 2024 (history) · csrc.nist.gov
- Cloud Security Alliance, research note on slopsquatting (AI package hallucinations as a supply-chain attack), 19 April 2026 · labs.cloudsecurityalliance.org
- Inozemtseva and Holmes, *Coverage Is Not Strongly Correlated with Test Suite Effectiveness*, ICSE 2014 · cs.ubc.ca
- Foster et al. (Meta), *Mutation-Guided LLM-based Test Generation at Meta*, FSE Companion 2025 · about half of the tests that caught faults missed by existing tests added no line coverage · arxiv.org/abs/2501.12862
- Zhao, Zhou and Cohen, *Do Coverage and Mutation Scores of LLM-Generated Test Suites Correlate with Their Effectiveness? (Replicability Study)*, 2026 · eleven models, Defects4J · arxiv.org/abs/2607.22880
- Wiz, *s1ngularity's aftermath*, 3 September 2025 · the Nx supply-chain attack that ran AI command-line tools with permission-bypass flags · wiz.io
- Wiz vulnerability database, CVE-2025-53773 · an assistant led to switch on its own auto-approve setting · wiz.io

- OWASP GenAI Security Project, *Agent Control Standard* repository · the reference Guardian is unauthenticated and fails open by default · github.com

Further reading: practice guides

- Augment Code (Molisha Shah), *Mutation Testing for AI-Generated Code: A Practical Guide*, 30 June 2026, updated 16 September 2026 · augmentcode.com
- Pact Foundation, *Introduction*, Pact documentation, last updated 25 August 2026 · consumer-driven contract testing · docs.pact.io
- JSON Schema, *Creating your first schema* · json-schema.org
- Datadog, *Progressive Rollouts and Canaries* · guardrail metrics that pause or stop a rollout · docs.datadoghq.com
- Unleash (Michael Ferranti), *Canary release vs rolling deployment: Rollback speed, risk, and resources*, 29 November 2024 (history) · getunleash.io
- Octopus Deploy, *Canary Deployments: Pros, Cons, And 5 Critical Best Practices* · octopus.com

H Document history

Section numbers in rows before 2026.6 refer to the numbering of their own edition.

VERSION	DATE	SUMMARY
v2026.6	October 2026	Edition for adopting firms. From this edition the Standard is published in two documents: the rules in The Certance Standard, and the evidence, regulatory map, worked examples and adoption plan here. Chapters are renumbered; How to use this document explains the numbering.
v2026.5.1	9 October 2026	Revision. Adds the Standard on one page, plainer wording and one rule layout throughout, and a redrawn Figure 4.1. Narrows statements in Ch. 13, Figure 4.1 and A.3 to match their sources and the rest of the Standard.
v2026.5	21 September 2026	Full review against the evidence. Adds evaluation of the agent roster, 12.2; independent waivers and emergency changes, Ch. 4 and A.2; security scans and a dependency check, 8.1 and 9.1; the 2026 supervisory statements, Ch. 13; the SSDF mapping, A. 4; and the glossary. Corrects the statement on EU AI Act Article 4.
v2026.4	20 September 2026	Founder review. Aligns Ch. 7 with the Coverage & Evidence Audit scoring model; labels the code smells as practitioner observation.
v2026.3	28 August 2026	Corrects the April 2026 incident timeline, the Veracode trend period, and four dates and version numbers.

VERSION	DATE	SUMMARY
v2026.2	23 August 2026	Re-verifies every figure; corrects or removes six claims. Adds Ch. 13. Renamed The Certance Standard.
v2026.1	July 2026	First draft.

Certance.

NEXT STEP

See where your test suite stands against this Standard.

The Coverage & Evidence Audit maps your critical business journeys to the tests that protect them, ranks each gap by consequence and detection delay, and checks which of the five gates your change records already evidence. Three days at a fixed scope and fee, ending with a gap report your board and risk function can read. It works from records your own engineer exports. It needs no access to your systems and no production or customer data.

THE AUDIT	certance.eu/coverage-audit
BOOK A CALL	cal.com/certance/coverage-audit
EMAIL	contact@certance.eu
ON REQUEST	A sample gap report and the vendor due-diligence pack
CURRENT EDITION	certance.eu/certance-standard

ABOUT CERTANCE ADVISORY

Certance Advisory is a quality engineering consultancy for regulated software teams, built on experience of regulated banking, life sciences and e-commerce systems. It has no referral or revenue-share arrangements with tool vendors. The current version of The Certance Standard and the reasoning behind it are published at certance.eu/certance-standard.

This Standard is engineering guidance. It gives no legal advice and no compliance opinion.

© 2026 Certance Advisory · The Certance Standard · v2026.6 · October 2026