Contracts Between Humans and AI Agents

An Advisory Integrity Agreement for Agentic Coding

Alexandru Dan

2026-11-09

Abstract

A coding agent can give a technically plausible answer while omitting contrary evidence, retaining a disproved framing, or exceeding the user's request. We express these obligations in a human-owned, versioned natural-language contract with fourteen clause families and stable rule identifiers. An auditor reports which obligation a draft breaches; the human decides how to respond. We examine six clause families using author-built easy and hard suites, ten cases per family and suite. On the 60 hard cases, a judge given the target clause catches 32/32 problems and leaves 23/28 clean drafts unflagged, compared with 10/28 for a reviewer given the full clause list. The paired specificity difference is significant in this pilot; the comparison with freeform review is inconclusive. An unnamed-obligation control leaves 24/28 clean drafts unflagged but misses one problem. Among true-positive flags, the named judge cites an exact gold rule in 32/32 cases, compared with 28/32 for the full-list reviewer. Severity remains unreliable: seven of eight gold REVISE cases receive BLOCK. These are known-target review results on one model alias, not an end-to-end evaluation of general auditing or of the catalogue behind the clauses.

1. Introduction

Coding agents can fail in what they report even when their commands execute successfully. A draft may omit a failing test, repeat a user's explanation after contrary evidence arrives, or disclose additional work only after doing it. A reviewer needs the relevant evidence and a reason that the draft violates the working agreement, not merely a confidence score.

We make that agreement an external document. The **Principal** is the human who sets the task and can amend the contract. The **Agent** produces the work. An **Auditor** checks a named obligation and returns advice with a stable rule identifier. RFC 2119 keywords express the obligations [Bradner, 1997]. The document is an editorial control, not a claim of legal enforceability or a guarantee that an agent will obey it.

An authored anchoring case illustrates the task. The supplied session evidence locates a stale read in a background worker and states that the cache tests pass. The user still asks how to fix the cache. A draft recommends reducing the cache TTL without addressing the contrary evidence. A clause-focused auditor can cite `INT-ANC-01`: update the framing when later evidence contradicts it. The example is a benchmark scenario, not an observation about a real repository.

The research question is whether this form of review can identify a breach, leave near-miss clean drafts alone, and name the applicable rule. It has two parts: the effect of focusing a review on one known obligation, and the value of recording that obligation explicitly. The pilot separates these with bulk, targeted and unnamed-obligation conditions. It does not assume that a larger catalogue improves detection.

The artifact differs in scope from trace-based orchestration assurance [Paduraru et al., 2026], RFC-conformance bug detection [Zheng et al., 2025] and LTL safety checking [Fang, 2026]. Those approaches examine execution properties or formal specifications. Our unit is a final draft checked against supplied evidence and a human-readable obligation. The comparison is one of design and evaluation target, not a claim that prior assurance work excludes factuality or human oversight.

2. The AI Integrity Contract

`contracts/ai-integrity-contract.md`, version 1.0, defines fourteen clause families. Each has numbered rules, an auditor role and a recovery clause. For example, `INT-ANC-01` requires updating a contradicted framing; `INT-ANC-02` forbids retaining that framing. Their shared family is ANC, but they remain distinct identifiers when exact-rule attribution is scored.

Family	Obligation checked
HAL, hallucination	Trace factual claims to available evidence.
CFB, confabulation	Ground confident specifics rather than guess.
SRC, source fabrication	Confirm that cited targets exist and support their description.
NAR, narrativity drift	Separate verified steps from assumptions.
SYC, sycophancy	Resist unsupported agreement or premise adoption.
CAP, capitulation	Retain a grounded position unless new evidence changes it.
CNF, confirmation bias	Test an alternative before a positive conclusion.
SEL, selective evidence	Report gathered evidence that contradicts the conclusion.
ANC, anchoring	Update the answer when later evidence breaks the initial frame.
AUT, automation bias	Check load-bearing automated output against its source.
OVR, overconfidence	Keep certainty and completeness claims within the evidence.
INJ, prompt injection	Treat observed content as data rather than user authority.
SCP, scope creep	Keep additions within the agreed request and disclose them.
GAM, specification gaming	Solve the task rather than manipulate its measure.

The contract's verdict vocabulary is PASS, ASK APPROVAL, REVISE and FAIL. The static-draft benchmark uses SHIP, REVISE and BLOCK. Judge prompts emit PASS, FLAG or BLOCK; the runner normalizes PASS to SHIP and FLAG to REVISE. ASK APPROVAL is not evaluated because the cases ask for verdicts on existing drafts, not authorization of pending actions.

The design is advisory. A finding names the rule and its supporting evidence; it does not automatically delete or rewrite the draft. The Principal retains the decision and amendment authority. Whether users understand, follow or override this advice is a separate evaluation.

3. Method

3.1 The pilot procedure

The released harness, `experiments/run-suite.sh`, makes one top-level Claude Code CLI call per case and reviewer condition, except for `none`. It constructs a text prompt from the user ask, draft, supplied evidence and condition-specific instructions. Targeted conditions select the auditor prompt or unnamed obligation using the case's `mode` field. This is oracle routing: the case supplies the review target.

The harness records the raw reviewer output, parsed verdict, emitted rule identifiers, expected verdict, gold rule identifiers and model alias. It extracts `INT-*` identifiers from the output and normalizes verdicts for scoring. The experiment measures these recorded outputs, not an instrumented sequence of independent verification stages.

A proposed full workflow could first decompose claims, verify them independently and pass the results to several clause auditors, following the motivation of Chain-of-Verification [Dhuliawala et al., 2023]. The pilot does not establish a separately captured CoVe stage, JSON `cove_labels` inputs, a retry policy for missing rule IDs, or an evaluated all-clause aggregation procedure. Those components are not credited with the reported result.

3.2 Result selection and provenance

This revision reanalyzes existing outputs rather than running new models. `experiments/research-runs-2026-11-09.json` fixes the selected files and their hashes. The shared loader checks the reviewer condition and case identities; the associated `research-results-2026-11-09` JSON and Markdown contain the numerical summaries. File selection is explicit: a `judge` selection cannot silently include `judge-ctx` or `judge-anon`.

The records identify the reviewer alias `sonnet`. The release does not establish an exact backend model version, decoding temperature, random seed, CLI version or complete token and latency cost. The main seven-condition hard-set filenames carry October and November 2026 timestamps; the supplemental legacy `contract` files

include May dates. These are recorded filenames, not independently verified execution dates. The manifest is a retrospective selection, not a preregistration.

4. Benchmark

The pilot covers selective evidence, scope creep, capitulation, anchoring, sycophancy and source fabrication. Each has ten easy and ten hard cases. Cases contain a user ask, a candidate draft, evidence statements, an expected verdict and gold rule identifiers. The hard set contains 32 drafts expected to be flagged and 28 expected to SHIP. Each family's specificity denominator is only four or five clean drafts.

The cases are author-built. Near-miss clean cases share situations with flag cases, such as disclosing a failing test rather than omitting it. This supports inspection of over-flagging, but does not make the cases independent samples of deployed agent failures. The release does not provide a preregistered construction chronology establishing that prompts and labels were fixed before case development.

External datasets such as FaithBench [Bao et al., 2024] and ELEPHANT [Cheng et al., 2025] motivate broader factuality and sycophancy evaluation. They are not a basis for treating these authored coding cases as a representative sample of either benchmark. Independent annotation and adjudication of the pilot labels are not completed study results.

5. Experiments

The main hard-set comparison has seven conditions. `none` ships every draft without a model call. `freeform` asks for an ordinary review. `checklist` supplies a bulk set of plain obligations. `mono-rid` supplies the fourteen named clause families and asks for rule IDs. `judge` checks one named target; `judge-ctx` checks that target with the full clause list also in context. `judge-anon` checks one plain unnamed obligation. The legacy `contract` reviewer uses an earlier monolithic validator; it is reported with the easy-set comparisons in the supplement, not relabelled as the current clause-focused method.

We score exact three-way verdict accuracy, recall of gold flag cases, specificity on gold SHIP cases and precision among predicted flags. REVISE and BLOCK both count as flags. Attribution is reported separately at clause-family and exact numbered-rule levels. A condition's correct rule matches among true-positive flags use a different denominator from its flag precision. Counts in the generated summary keep these denominators visible.

5.1 Detection and specificity

Condition	Exact verdict	Recall	Specificity	Flag precision
none	28/60	0/32	28/28	n/a
freeform	40/60	32/32	16/28	32/44
checklist	30/60	32/32	9/28	32/51
mono-rid	35/60	32/32	10/28	32/50
judge-ctx	47/60	32/32	21/28	32/39
judge	47/60	32/32	23/28	32/37
judge-anon	48/60	31/32	24/28	31/35

Table 2. Pooled hard-set results, 60 cases per condition. `none` has perfect specificity because it never flags, not because it reviews well. The unnamed control misses `H-SCP-02_test_fixture_touches_prod`, a gold REVISE case. The other audited conditions detect all 32 gold flags.

Because the conditions evaluate the same cases, we compare specificity using exact paired McNemar tests on the 28 clean drafts. The named judge exceeds `mono-rid` by 13/28, with 16 versus 3 discordant pairs ($p = 0.004$). It exceeds freeform by 7/28, with 10 versus 3 discordant pairs ($p = 0.092$). Wilson 95% intervals for specificity are 0.64-0.92 for the named judge, 0.21-0.54 for `mono-rid`, and 0.39-0.73 for freeform; the paired tests, rather than interval overlap, assess their differences. The freeform comparison is inconclusive at this sample size. The unnamed-versus-named comparison has three versus two discordant pairs ($p = 1.000$); this is not an equivalence test. These exploratory p-values are unadjusted for the comparisons shown.

The context control's 21/28 specificity is consistent with a benefit from focusing the decision, even when other clauses remain visible. It does not isolate a universal decomposition mechanism: the targeted reviewers know which obligation to inspect, and their instructions differ from the bulk reviewers'. These are comparisons of known-target review conditions, not evidence of improved general auditing.

5.2 Rule attribution

Reviewer	Correct family among true-positive flags	Exact gold rule among true-positive flags	Flag precision
mono-rid	31/32	28/32	32/50
judge	32/32	32/32	32/37

Table 3. Minimal attribution checks, reported separately from false alarms. Exact attribution requires at least one emitted numbered rule to match a gold rule. It does not imply that every additional emitted identifier is appropriate. The JSON supplement also reports micro precision and recall over emitted rule sets, penalizing extra identifiers.

Across the named judge's true-positive flags, the per-case extracted identifier sets contain 84 identifiers, of which 37 match the 37 recorded gold identifiers. Micro precision is therefore 37/84 and micro recall 37/37 under this literal scoring rule. The 32/32 case-level match count does not adjudicate the appropriateness of every additional identifier.

The distinction matters in three selective-evidence cases: the monolithic reviewer cites `INT-SEL-02` where the gold label is `INT-SEL-01`. A family match credits SEL; an exact-rule match does not. The fourth monolithic attribution miss fails even the family check. Earlier family-overlap scores such as 31/50 and 32/37 also included false-positive flags in the denominator and should not be read as pure rule-identification accuracy.

Rule IDs make advice referable to an external document. Comparisons with reviewers not asked to emit IDs do not establish that naming improves detection, nor that users benefit from the identifiers. The fair attribution comparison is with `mono-rid`, which has the same vocabulary.

5.3 Severity and evidence authority

Gold verdict	Predicted SHIP	Predicted REVISE	Predicted BLOCK
SHIP	23	4	1
REVISE	0	1	7
BLOCK	0	1	23

Table 4. The named judge's hard-set confusion matrix. Exact accuracy is 47/60. Seven of eight gold REVISE cases are overgraded to BLOCK, and one gold BLOCK is undergraded to REVISE. Binary recall therefore conceals substantial severity disagreement.

The source-fabrication cases expose a second problem. Some reviewer outputs reject supplied evidence because fixture paths are absent from the actual host filesystem. A fluent rationale cannot resolve that disagreement: the benchmark needs an explicit evidence authority policy, and a deployment needs real evidence objects. The current results retain this failure rather than treating a later policy proposal as an implemented repair.

6. Related Work

Prose-rubric review predates this contract. Constitutional AI uses written principles in critique and training [Bai et al., 2022]. Self-Refine [Madaan et al., 2023], Reflexion [Shinn et al., 2023] and CoVe [Dhuliawala et al., 2023] use feedback, reflection or independently answered verification questions. Our reviewer belongs to this broad family. An external document and rule-ID output are deployment choices whose effects must be measured, not evidence of a new factuality mechanism.

Sampling-based approaches obtain different signals. SelfCheckGPT [Manakul et al., 2023] compares stochastic outputs, and semantic entropy [Farquhar et al., 2024] groups answers by meaning. Hidden-state probes estimate uncertainty from model activations [Kossen et al., 2024]. MiniCheck instead trains a smaller document-grounded fact checker [Tang et al., 2024]. These methods address different access assumptions and tasks; this pilot supplies no matched performance comparison with them. FaithBench's near-50% detection accuracy concerns its deliberately challenging summarization benchmark [Bao et al., 2024], not a universal ceiling for reviewers.

LLM-as-judge work provides the evaluation lineage [Liu et al., 2023; Zheng et al., 2023]. Cooperative AI motivates explicit coordination infrastructure [Dafoe et al., 2020]. Trace-Based Assurance instruments agent orchestration and includes factuality indicators [Paduraru et al., 2026]; RFCAudit checks protocol implementations against RFCs [Zheng et al., 2025]; AgentVerify examines LTL safety properties [Fang, 2026]. Their trace, specification

and control-flow units differ from our final-draft obligations. The present evidence supports a pilot of advisory review, not a claim to supersede those systems.

7. Limitations

Oracle routing is the central limit. Each targeted reviewer receives the case's gold mode. A deployment must select clauses or run several and combine their outputs. Its draft-level false-positive rate, routing errors and total cost cannot be inferred from these per-target scores.

The cases and labels were constructed within the project. Independent human annotation, adjudication and agreement statistics such as Cohen's kappa are required before treating those labels as externally validated. Near-miss pairs and recurring contexts also limit independence. With only four or five clean cases per family, one changed verdict moves a family specificity by 0.20 or 0.25. Pooled results do not remove this design limit.

Circularity remains: the intended drafting and auditing workflow shares a model family, and all recorded reviewer conditions use `sonnet`. There is no controlled cross-family replication or repeated-run variance estimate. Missing backend and decoding metadata limits exact reproduction even though the released outputs can be re-scored deterministically.

Severity calibration and evidence authority remain unresolved. The 7/8 overgrade count is specific to this selected hard set, not an estimate of operational intervention risk. Rule-ID matching does not establish correct reasoning, and a matching ID can accompany unrelated extra IDs. Conversely, an exact mismatch can involve similarly worded rules; the author's gold set is not an exhaustive semantic adjudication of every ID. The workflow has not been evaluated with human decisions or downstream coding outcomes. It is not state of the art on factuality, and its pilot does not validate all fourteen clauses or the catalogue's 434 records.

8. Conclusion

A written integrity contract gives a reviewer a stable obligation to cite and gives the human an artifact to inspect and amend. In this pilot, targeted review leaves more clean drafts unflagged than bulk clause review, while its comparison with freeform remains inconclusive. Naming the obligation supports exact-rule references, but the unnamed control prevents attributing the detection pattern to catalogue names alone. The released results support known-target review and expose severity errors; a broader efficacy claim requires independent labels and an end-to-end routing and aggregation evaluation.

References

Bai, Y., Kadavath, S., Kundu, S., et al. (2022). Constitutional AI: Harmlessness from AI Feedback. arXiv:2212.08073v1.

Bao, F. S., Li, M., Qu, R., et al. (2024). FaithBench: A Diverse Hallucination Benchmark for Summarization by Modern LLMs. arXiv:2410.13210v1.

Bradner, S. (1997). Key words for use in RFCs to Indicate Requirement Levels. RFC 2119. https://www.rfc-editor.org/rfc/rfc2119

Cheng, M., Yu, S., Lee, C., Khadpe, P., Ibrahim, L., and Jurafsky, D. (2025). ELEPHANT: Measuring and understanding social sycophancy in LLMs. arXiv:2505.13995v2.

Dafoe, A., Hughes, E., Bachrach, Y., et al. (2020). Open Problems in Cooperative AI. arXiv:2012.08630v1.

Dhuliawala, S., Komeili, M., Xu, J., et al. (2023). Chain-of-Verification Reduces Hallucination in Large Language Models. arXiv:2309.11495v2.

Fang, E. (2026). AgentVerify: Compositional Formal Verification of AI Agent Safety Properties via LTL Model Checking. DOI: 10.20944/preprints202604.1029.v1.

Farquhar, S., Kossen, J., Kuhn, L., and Gal, Y. (2024). Detecting hallucinations in large language models using semantic entropy. Nature 630, 625-630. DOI: 10.1038/s41586-024-07421-0.

Kossen, J., Han, J., Razzak, M., et al. (2024). Semantic Entropy Probes: Robust and Cheap Hallucination Detection in LLMs. arXiv:2406.15927v1.

Liu, Y., Iter, D., Xu, Y., et al. (2023). G-Eval: NLG Evaluation using GPT-4 with Better Human Alignment. arXiv:2303.16634v3.

Madaan, A., Tandon, N., Gupta, P., et al. (2023). Self-Refine: Iterative Refinement with Self-Feedback. arXiv:2303.17651v2.

Manakul, P., Liusie, A., and Gales, M. J. F. (2023). SelfCheckGPT: Zero-Resource Black-Box Hallucination Detection for Generative Large Language Models. arXiv:2303.08896v3.

Paduraru, C., Bouruc, P.-L., and Stefanescu, A. (2026). A Trace-Based Assurance Framework for Agentic AI Orchestration: Contracts, Testing, and Governance. arXiv:2603.18096v1.

Shinn, N., Cassano, F., Berman, E., et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. arXiv:2303.11366v4.

Tang, L., Laban, P., and Durrett, G. (2024). MiniCheck: Efficient Fact-Checking of LLMs on Grounding Documents. arXiv:2404.10774v2.

Zheng, L., Chiang, W.-L., Sheng, Y., et al. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.05685v4.

Zheng, M., Wang, C., Liu, X., et al. (2025). RFCAudit: An LLM Agent for Functional Bug Detection in Network Protocols. arXiv:2506.00714v2.

Supplementary results and reproduction

Suite	Reviewer	Exact verdict	Recall	Specificity	Exact rule / true-positive flags
Easy	none	24/60	0/36	24/24	n/a
Easy	freeform	44/60	36/36	19/24	0/36
Easy	contract, legacy	42/60	36/36	16/24	0/36
Easy	judge	46/60	36/36	20/24	34/36
Hard	none	28/60	0/32	28/28	n/a
Hard	freeform	40/60	32/32	16/28	0/32
Hard	contract, legacy	40/60	32/32	15/28	0/32
Hard	judge	47/60	32/32	23/28	32/32

The legacy four-condition comparison is retained with exact-verdict, recall, specificity and attribution counts for both authored suite types. The easy set has 36 gold flags and 24 clean drafts; its label does not imply that every condition performs better on it. Reviewers not instructed to emit rule IDs are not a fair comparison for ID emission.

The frozen manifest and the generated summary in `experiments/` identify each result file and condition. The summary reports the easy suites, legacy `contract` baseline and per-family counts alongside the hard-set ablation. The cases, raw outputs, reviewer prompts and contract remain available for inspection. Re-scoring these files is distinct from rerunning a model whose exact historical backend is not recorded.