

Recomputable scoring is not the same as reproduced measurement

A CAI score can be recomputed from supplied evidence under identified scoring rules. That check does not independently repeat the assessment that produced the evidence.

Lasse Overgaard · Canine Development · 17 September 2026

The Codebase Assurance Index (CAI) separates measuring a repository from calculating a score from those measurements. Someone can repeat the calculation and obtain the same headline number without repeating the inspection of the codebase. Calling the whole assessment *reproducible* on the strength of that calculation would claim more than has been checked.

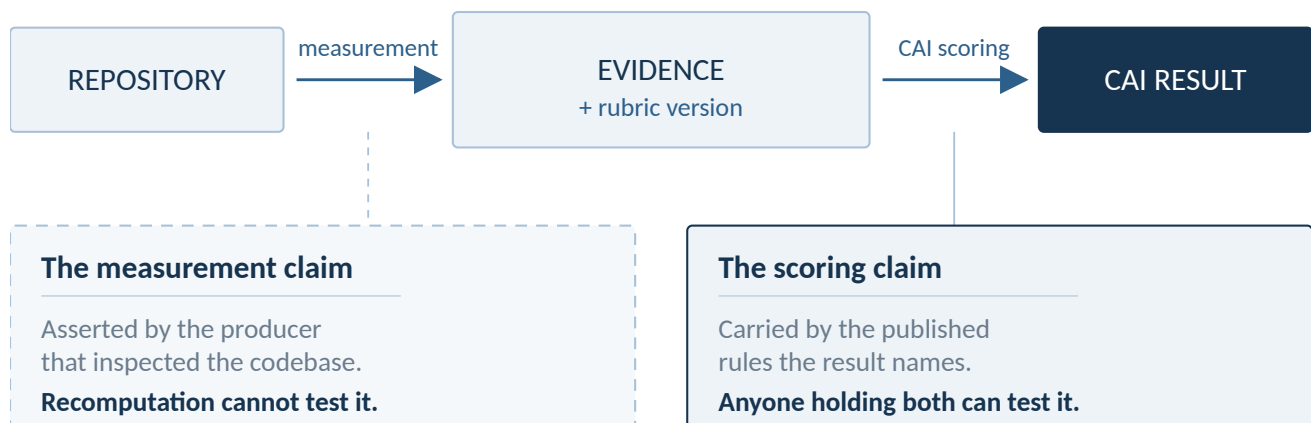
Consider an invoice. Recalculate its multiplications, discounts and tax rules, and you can establish that the total follows from the line items. The calculation cannot establish whether the items were delivered or whether the listed prices were correct. Those questions concern the inputs, even when the total adds up.

A CAI result has a similar division of work. First, a producer inspects a repository, meaning the stored codebase, its files and its history, and records the findings as structured evidence: scored measurements of that codebase. CAI then applies published scoring rules to the evidence and produces a headline number between 0 and 100. A third party with the evidence and rules can repeat this second stage without having participated in the first, making the relationship between evidence and result independently testable.[1][2][3]

Both stages are complete by the time a recipient receives the report. The recipient may have produced neither the report nor the software it describes, so the headline alone does not show which part of the process can be checked independently. A rule selected to suit a preferred answer can make a headline misleading even if the measurements are sound; incorrect measurements can make it misleading even when the rule is applied faithfully. Published scoring rules and the ability to recompute the result constrain the first problem. They do not settle the second, because a correctly calculated number can still be based on evidence about the wrong repository or an incorrect observation of the right one.

Two arrows, two claims

WHAT A CAI RESULT IS MADE OF



The dashed claim is the one this article is about. It is real work, honestly done, and it is not the claim that recomputation checks.

Figure 1. A CAI result is made in two stages. The first stage is a claim about the repository and the second is a claim about a calculation. Only the second is published as a rule anyone can apply.

The first arrow asserts a claim about the repository:

This evidence is an accurate measurement of this repository.

The second asserts a claim about a calculation:

Given this evidence and these scoring rules, this is the CAI result.

CAI makes the second claim mechanical and independently inspectable through published scoring rules. It does not publish a method for producing the measurements; each implementation builds its own. The standard's stated position is that implementations applying one version of the rules to identical measurements obtain the same number. Obtaining identical measurements independently is a harder problem that the standard does not claim to have solved. [10]

An evidence bundle carries the measurements used by the scoring calculation and names the rubric version under which they were produced. A rubric version identifies the published rule set for the result: how measurements are grouped and which parameters govern the arithmetic. Where a published catalog specifies no scoring parameters of its own, the version resolves to the scorer's published defaults. That resolution preserves the version's established scoring behaviour. Two CAI results are only comparable when folded under the same rubric version; a recipient with the relevant catalog can apply its published category mapping and scoring parameters to the supplied evidence.[2][3] [4]

The rubric archive serves a catalog only if the version declared inside the document matches the version under which it is published. Otherwise, the archive cannot attest which version the document represents, and someone pinning the

published version could check against the wrong definition. The archive withholds that catalog and reports it as unattested rather than serving it with a caveat.[9] Recomputation therefore depends on being able to establish the named version of the rules.

With the evidence and applicable rules in hand, a recipient can ask a bounded question: does this evidence produce the score in the report? The *fold* is the fixed procedure that reduces scored measurements to the single headline number. For fixed scoring inputs and rules it is deterministic, allowing the recipient to recompute and compare the result without relying on the producer's reputation or competence.[1][3] A disagreement can be stated against those particular inputs and rules; agreement establishes the calculation even if the recipient has never examined the repository.

Only deterministic dimensions enter that fold. Measurements produced by a language model travel with the evidence as advisory material, displayed as a band rather than a number, and remain outside the arithmetic.[2][3] Thus an advisory judgement may appear in a report without affecting the value the recipient recomputes. The recomputation tests the relationship between the supplied evidence and the number; it does not test how that evidence was produced.

Recomputing the score does not rerun the measurement

The following numerical example is **illustrative, not a measured CAI result**.

A dimension is one scored property of a codebase. The rubric groups dimensions into categories such as testing, dependencies, architecture and security. The distinction between calculation and measurement applies to a dimension in any of them.

Suppose an evidence bundle reports **6.2** for one measured dimension on CAI's 0 to 10 dimension scale.[2] The scorer can establish how 6.2 contributes to the result. The resulting number cannot establish whether 6.2 correctly represents the repository. The detector might have missed a finding, or its inspection might have covered only part of the relevant surface. Coverage travels with the evidence and enters the fold, so the calculation accounts for the reported coverage; it cannot determine whether the coverage figure itself is correct.

Other failures can occur before a number reaches the fold. The check might have been inapplicable to this codebase, or the inspected repository might differ from the one the recipient expected. Repeating the calculation leaves those choices in place. A reviewer seeking to establish whether the measurement was right must inspect how the producer arrived at the evidence, including what was examined and what was left out.

An unassessed dimension cannot be entered as zero in CAI's evidence format: zero would describe a measured failure, whereas no assessment produced a measurement. The dimension is absent from the bundle instead.[2] Absence does not carry its cause. The dimension may have been inapplicable, or the producer's detector may never have run; the arithmetic sees the same absence in both cases. That distinction has to be examined at the measurement layer.

Even a pinned rubric version fixes scoring rules rather than the conditions under which a measurement was made. A newly disclosed vulnerability can change a security finding without a change to the repository or rubric version; the survey changelog records that the disclosure moved the finding.[11] Repeating the fold over the same 6.2 cannot resolve that change, a missed finding, an applicability decision or a mistaken repository identity.[2][7] A report

saying that “the assessment is reproducible” leaves unclear which of those operations, if any, another party has actually repeated.

What open scoring makes testable

When a producer both supplies the evidence and controls the transformation into a headline, a recipient must accept both parts on the producer's account. Publishing the scoring rules and supplying the evidence makes the transformation independently checkable. For fixed evidence and governing rules, a producer cannot choose a more convenient headline without leaving a discrepancy a recipient can demonstrate.[1][3][7]

The following numbers are **illustrative, not measured CAI results**. If a report claims 72.2 but its evidence produces 67.4 under the governing rules, a recipient can demonstrate the mismatch. If the CAI reference implementation behaves differently from the published method, a minimal reproduction case can expose the defect.[1] When a score-moving rule changes, a new rubric version identifies that change instead of silently changing what an earlier version meant.[4]

The version name alone cannot prove that a catalog's content has remained fixed. Serving a catalog only under the version it declares detects a document filed under the wrong name, but not an edit that keeps the declared name. A digest of the catalog's canonical content inside the signed artifact makes a later edit demonstrable to someone who still holds a report issued under the earlier content.[9] The holder can compare the signed digest with a digest of what the publisher now serves. This provides detection rather than prevention when the rule publisher is also a party the recipient must trust. These checks make the scoring derivation contestable while leaving the truth of the upstream measurements for a separate examination.

Three verification questions

A recipient can use *verified* to mean several different checks. Each addresses a separate part of a CAI result:

THREE THINGS THE WORD VERIFIED CAN MEAN

Measurement repository to evidence	Scoring evidence to result	Provenance artifact and issuer
ESTABLISHES That the evidence describes the repository fairly. FALSIFIED BY Independently repeating enough of the inspection.	ESTABLISHES That the number follows from the evidence under the named rubric version. FALSIFIED BY Recomputing the fold and getting another number.	ESTABLISHES That this is the artifact issued, unchanged since it was signed. FALSIFIED BY A failing signature or content-hash check.
The producer answers for it.	Arithmetic answers for it.	The signer answers for it.

Only the middle check is settled by arithmetic. A pass on any one of the three says nothing about the other two.

Figure 2. Three checks a recipient may mean by the word verified. Each has its own falsifier, and a pass on one carries no weight for the others.

The scoring check presupposes that the rules in force can be identified before the evidence is folded. The rubric version identifies those rules, while the digest of the published catalog's content gives a holder of a signed report a way to detect subsequent content changes. Neither the table nor a successful scoring check makes an upstream measurement valid.

CAI's published challenge model accordingly distinguishes arithmetic disputes, defects in the reference scorer, disputes over the issuer's measurement, and challenges to the rubric or its methodology.[1] The current delivery design also treats score recomputation and artifact authenticity as distinct checks. Its package format, still at **Draft** status, can bind the scored subject's identity, producer, rubric, evidence and result in one signed payload. A recipient can then check the signature and separately recompute the headline.[5][6]

A signature cannot make a measurement true

Cryptographic signatures do real work. They are also disappointingly uninterested in whether you were right.

Under the relevant key and trust model, a valid signature can establish that a signed payload is authentic and has not changed since signing. The current delivery design uses that property to bind subject, producer, evidence, rubric and result in a tamper-evident artifact.[5][6] The approach has a precedent in software supply-chain work: in-toto uses signed metadata and defined actors to make supply-chain steps and relationships between artifacts verifiable after the fact.[8]

A signature authenticates a claim without investigating its truth. If someone signs

Dimension D17 = 6.2

the signature can help establish who attested to that statement and whether the signed statement has changed. It cannot inspect the repository to determine whether 6.2 was the correct measurement. The delivery specification says explicitly that the signature does not attest that the evidence truthfully measures the real repository; the measurement remains the producer's claim.[5]

The signed artifact can also record who performed the measurement and which scanner and scanner version the producer reports using; the scanner fields are optional.[6] The signature binds those declarations into the payload, but it cannot establish that the scanner produced correct evidence. This provenance identifies a party whose measurement can be questioned. Even a named tool version does not provide a procedure another party can follow to obtain the same measurements.

A bad measurement can be preserved and attributed perfectly. Those are useful properties of an artifact, though the measurement still needs its own scrutiny.

Vocabulary for the checks

Terms such as *transparent*, *verified*, *reproducible* and *trusted* need a stated object and a stated check. In a CAI report, three phrases describe materially different work. Using the right one tells a reader where the checking began and which part of the result remains the producer's claim.

Recomputed score means the supplied score-bearing evidence has been folded again under the applicable scoring rules, producing a value that can be compared with the claimed result. This is a check of the reported number against the reported inputs. Its outcome can be tested by another recipient who holds the same evidence and rules, while the observations that supplied those inputs remain outside the check.

Verified artifact means the checks appropriate to the artifact have passed. Depending on the delivery mechanism, those checks may include integrity, signer or subject-binding checks under an applicable trust model. A report should name the checks performed, since verifying that a payload has not changed does not determine whether its underlying measurement is accurate.

Reproduced measurement makes the stronger claim that a third party has independently repeated enough repository-to-evidence work to establish that the same underlying measurements can be obtained. Repeating the arithmetic after receiving the evidence would leave this question unanswered. The claim requires work that starts upstream of the evidence bundle, with the repository and the measuring procedure.

The sentence “**The CAI score can be recomputed from the supplied evidence under the identified scoring rules**” identifies both the starting point and the check. “**The software assessment is reproducible**” suggests that someone can also start with the repository and independently recover the evidence. If that happened, the report should describe the procedure followed and what independent measurements were obtained. Otherwise, the broader wording assigns the scoring check credit for observations it did not test. The scoring and artifact checks retain their value without that broader claim; a recipient simply needs to know which result belongs to which check.

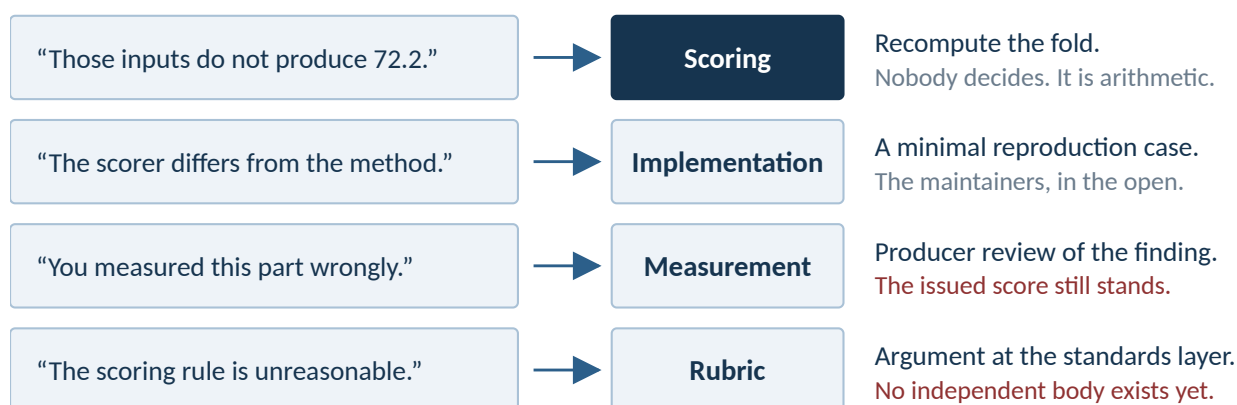
Determinism and verifier tolerance

The scoring fold is deterministic for fixed effective scoring inputs and rules.[3] A verification tool still needs a criterion for treating the recomputed and claimed numbers as matching. The current CAI reference `Verify` method applies a default tolerance of ± 0.5 to that comparison.[3] This is a verifier acceptance rule, not randomness in the calculation.

The fold determines what value the evidence produces under the governing rules. The verifier decides whether that value is close enough to the claimed result to report a match. A check can therefore pass under the default tolerance despite a difference between the two reported values. A verification tool or page should expose both values wherever the distinction could change a recipient's interpretation. The recipient can then distinguish the number calculated by the fold from the tool's decision to accept a claimed number within a tolerance; neither decision establishes anything about the repository measurement.

Where a dispute belongs

Disputes over arithmetic, the reference implementation, a producer's measurements and the scoring methodology call for different evidence and different responses. The first two concern the scoring layer, but one disputes a particular calculation and the other disputes whether the calculator follows its specification.



Recomputation settles the first row and nothing below it. Winning the third does not revise the number you hold.

Figure 3. Four complaints that look alike and belong in four places. Knowing which layer a disagreement sits in is what decides whether any evidence could settle it.

"Those inputs do not produce 72.2."

The recipient can recompute the score against the evidence and governing rules. A different result demonstrates a concrete arithmetic discrepancy.[1][3] This check needs no producer's permission or adjudicator: anyone holding the evidence and applicable rules can perform it, and a mismatch can be shown directly rather than submitted as an opinion for acceptance.[1]

"The reference scorer behaves differently from the published method."

This is a conformance dispute between the implementation and its specification. A minimal case that produces different behaviour under the published method and the reference scorer supplies the evidence needed to expose the defect.[1] The disagreement concerns the implementation even if a particular report happened to contain correct evidence.

"You scored this part of my repository incorrectly."

The producer who made the measurement must answer this complaint.[1] Recomputing an incorrect measurement faithfully will yield the result of that incorrect input again; it cannot settle whether the underlying observation was sound.

Under the current finding-dispute route, a confirmed false positive becomes a detector test to prevent recurrence, while the issued score stands.[11] Resolving the measurement complaint sharpens the producer's instrument rather than retroactively changing the result the recipient holds.

"The scoring rule itself is unreasonable."

A methodological objection may arise even when the measurement and calculation both follow the existing rules. It belongs at the rubric and standards layer; a successful challenge should inform a future rubric version instead of changing what an issued version meant.[1][4]

Anyone may read the rules, run the scorer and publish scores under the standard. The organisation that wrote the standard still decides the contents of the next version. No independent governing body exists yet; the governance

page states the present arrangement and the conditions to be met before CAI describes its governance as independent.[12] These decision rights matter when a complaint asks for a rule change rather than correction of a calculation or measurement.

Responsibility after recomputation

Open, deterministic scoring gives a recipient an identifiable rule set, an inspectable transformation from supplied evidence to result, and a way to demonstrate a mismatch.[1][3][4][7] These properties allow a recipient to test the published number even when the producer's measurement remains open to dispute. They also limit how far the successful check can be carried: the scoring layer cannot certify that the repository was inspected correctly.

Responsibility for the measurement remains with the producer, responsibility for a signed claim with its signer under the relevant trust model, and responsibility for the scoring method with those who govern the rubric. Keeping those claims and routes for challenge distinct makes a CAI result more useful to someone who needs to examine it.

Conclusion

When a CAI result is described as reproducible, the practical question is **from what starting point?** From the supplied evidence and identified rules, a recipient can recompute the score and test the evidence-to-result relationship. Repeating that operation neither inspects the repository nor reproduces the evidence. If the result arrived as a signed artifact, the recipient may also check who issued the signed claim and whether its contents have changed; that remains a separate check.

A CAI report should name the checks performed and their starting inputs. A recipient can then tell which claims have been independently tested and which measurement claims still require an answer from the producer.

References

1. **CanineCC/CAI**, `docs/CHALLENGE.md`, *Challenge & review: how to dispute a CAI*, reviewed 9 September 2026. Primary source for the current separation between arithmetic reproduction, reference-scorer defects, issuer measurement disputes and rubric or methodology disputes.
2. **CanineCC/CAI**, `src/Cai.Scoring/EvidenceBundle.cs`, reviewed 9 September 2026. Primary implementation source for the evidence bundle, the 0 to 10 deterministic dimension scale, coverage/confidence fields, rubric identity and the separation between evidence production and scoring.
3. **CanineCC/CAI**, `src/Cai.Scoring/CaiScorer.cs`, reviewed 9 September 2026. Primary implementation source for deterministic scoring, catalog-aware scoring, headline verification and the current default ± 0.5 verification tolerance.
4. **CanineCC/CAI**, `src/Cai.Scoring/RubricCatalog.cs`, `src/Cai.Scoring/ScoringParameters.cs`, ADR-0004 and commit `1d557c36faa6f7650cfe6746db01665ad2ac9bfc`, 22 August 2026. Primary architecture and implementation sources for versioned rubric catalogs, frozen scoring parameters and score-moving rule identity.
5. **CanineCC/CAI**, `docs/spec/cai-delivery-package.md`, *CAI-delivery package: format specification*, version 1.0, **Draft**, reviewed 9 September 2026. Current closed-loop delivery design covering signed payloads, subject and producer provenance, offline signature verification and score recomputation. The specification explicitly states that the signature does not attest that the evidence truthfully measures the real repository.
6. **CanineCC/CAI**, `src/Cai.Delivery/DeliverySigning.cs` and `src/Cai.Delivery/DeliveryPayload.cs`, reviewed 9 September 2026. Primary implementation sources for the current separation between signature authenticity and

- headline reproduction in the delivery verifier, and for the payload's producer, scanner and scanner-version provenance fields.
7. **Jimmy Borch / Canine Development**, *From Score to Evidence: Recomputation, Provenance, and Measurement Boundaries*. Founding technical treatment of the separation between repository measurement, deterministic aggregation, artifact provenance and verification.
 8. **in-toto**, official specification and documentation, reviewed 25 August 2026. Used only as an external software-supply-chain precedent for signed metadata, authorised functionaries, artifact relationships and verification of defined supply-chain steps. It is not a CAI dependency and does not validate CAI's measurement method.
 9. **CanineCC/CAI**, `src/Cai.Delivery/RubricDigest.cs` and `src/Cai.Scoring/RubricCatalogStore.cs`, reviewed 9 September 2026. Primary implementation sources for the distinction between a rubric catalog's declared name and its canonical content, and for the content digest that makes a later edit to a published rubric version detectable by a holder of an earlier report.
 10. **Codebase Assurance Index**, *Implementations*, `codeassuranceindex.info/implementations`, read 14 September 2026. Public source for the division between the published scoring rules and the unpublished measurement method, for each implementation building its own measuring, and for the standard's position that identical measurements under one rubric version reach an identical number while arriving at identical measurements is not a solved problem.
 11. **Codebase Assurance Index**, *Rubric versions*, `codeassuranceindex.info/rubric`, read 14 September 2026. Public source for the finding-dispute route and its outcome, a detector test rather than an adjusted score, and for advisory data moving a finding under an unchanged rubric version with the change disclosed in the survey changelog.
 12. **Codebase Assurance Index**, *Governance*, `codeassuranceindex.info/governance`, read 14 September 2026. Public source for the current governance position: the standard's author decides the next rubric version, no independent body exists, and the page publishes the bar it will meet before describing itself as independently governed.

STATE OF THIS ARTICLE

This article describes the CAI implementation and specification as they stood on 17 September 2026, at repository commit a24b47b2. Those sources move. A claim here about what the reference scorer, the delivery specification or a published catalog currently does is a statement about that commit rather than a standing guarantee, and a reader checking it later should expect a newer state and should read the current repository alongside this article. The argument does not depend on the code staying still. The specific behaviours it cites do.

Signed 17 September 2026 by Lasse Overgaard, Canine Development.

Informative and non-normative. Where this article and the published CAI specification or the applicable rubric version disagree, the specification and the rubric control.