

How the Codebase Assurance Index is constructed

Ten lenses, ordered weighting, and the advisory firewall

This paper explains the construction of CAI. It does not create normative requirements. The current CAI specification, versioned rubric and reference scorer control in case of conflict.

Version	1.0
Date	23 August 2026
Signed	14 September 2026 by Jimmy Borch
Author	Jimmy Borch
Originating organisation	Canine Development
Series	CAI Founding Papers
Status	Informative, non-normative technical paper
Technical reference state	CAI repository through commit 1d557c36faa6f7650cfe6746db01665ad2ac9bfc

Contents

Abstract

1. Executive summary

2. Claim and scope

3. Terms and data model

4. The ten-lens model

5. Aggregation method

6. The advisory firewall

7. Reporting: score, bands, context and overrides

8. Worked examples and sensitivity

9. Reproducibility, versioning and verification

10. Limitations and research requirements

11. Conclusion

References

Abstract

A codebase assurance score compresses a large body of technical evidence into a form that can be compared, discussed and tracked. The value of that compression depends on making the construction visible: a strong area should not be able to erase a serious weakness, a missing measurement should not silently become a failure, and interpretive judgement should not be allowed to move a deterministic score.

This paper describes the aggregation method used by the Codebase Assurance Index (CAI). Measuring a repository and aggregating that measurement into a score are separate processes in CAI: a producer emits structured evidence, and a reference scorer folds that evidence into a result. The fold runs through ten lenses, each a standing perspective on the codebase such as code health, architecture or security, and advisory judgement stays outside its arithmetic. Wherever several scores are combined, they are ordered from weakest to strongest and combined by ordered weighted averaging, an operator that assigns weight by rank rather than by source, and CAI's parameters place the greatest weight on the weakest measured area so that it retains material influence rather than being averaged away.

Those parameters do not live in the scorer. The versioned rubric carries the score-moving configuration: the OWA decay constants, the critical gate, which is a contributor threshold below which a lens's reported band is capped, the architecture-surface rules, the band cutlines that turn a number into a label, and the quality-bar shifts that move those cutlines for a repository's criticality tier. A rubric version therefore identifies the scoring semantics under which a result was produced.[5][6][14]

Missing evidence is not converted into a synthetic zero. An omitted dimension does not participate in the fold, and neither does an absent lens or a null meta-dimension, the latter being a criterion that enters a lens directly rather than through a category. Partial coverage behaves differently: coverage records how much of a dimension's target surface its evidence represents, and it reduces that dimension's effective score rather than removing the dimension. Survey completeness travels beside the score instead of inside it, in a signed but non-scored `surveyFit` object reporting how much of the applicable survey surface resolved, which exposes a thin reading without penalising the codebase for one.[7][16]

Given the same eligible evidence and the same rubric version, the fold is deterministic, which makes the evidence-to-score relationship inspectable and testable. That is a narrower property than measurement validity: recomputing a result establishes that it follows from the stated evidence and rules, not that the upstream repository measurement was correct. Software-measurement literature draws the same distinction between a repeatable measurement procedure and evidence that the resulting measures are valid for their intended use.[22][23]

CAI is therefore best understood as an inspectable aggregation standard whose assumptions are exposed rather than hidden. This paper sets out those assumptions and the scoring mechanics, and it marks the research questions that would be required only for stronger empirical claims about robustness, calibration or external validity.

1. Executive summary

The CAI is a 0 to 100 aggregation method for technical evidence about a codebase.[1][3] It answers a practical reporting question: given a structured body of evidence, how should that evidence be reduced to lens scores and a headline result without allowing numerous strong areas to conceal a serious measured weakness?

That question is harder than it first appears, because every reasonable answer is a choice that could have gone another way. An average lets strength hide weakness. A minimum discards everything except the worst finding. Scoring a missing measurement as zero penalises a codebase for what nobody looked at, while treating it as absent changes what the remaining evidence is weighed against. No compression of many measurements into one number is free of such choices, so the useful question is not whether a scoring method makes them but whether a reader can see which ones it made.

CAI's answer has five parts.

First, measurement and aggregation are separate processes. A producer measures a repository and emits an evidence bundle. The reference scorer folds the score-bearing evidence into categories, lenses and a headline. Re-running the fold does not require the producer's measurement engine, although evaluating the correctness of the upstream evidence does.

Second, CAI uses ten lenses. Five are core lenses and five are model-aware lenses whose relevance depends on characteristics of the codebase.[8] The numeric headline is calculated over the lenses that actually produce scores.

Third, advisory judgement is separated from the number. Dimensions or meta-dimensions marked advisory can remain visible for interpretation, but the reference scorer excludes them from category, lens and headline arithmetic.[4][6][7]

Fourth, the fold is deliberately asymmetric. Inputs are ordered from weakest to strongest and combined through ordered weighted averaging. Under the current rubric defaults, the weaker measured areas receive greater rank weight than the stronger ones. The exact OWA parameters are normative rubric data, not natural constants.[6][14][19][20]

Fifth, the scoring semantics travel with the rubric version. The versioned rubric now carries the score-moving aggregation parameters, including OWA decays, the critical gate, architecture-surface parameters, band cutlines and quality-bar shift rules.[5][14] The narrow deterministic contract is therefore:

> same eligible evidence + same rubric version = same CAI result

Canine Development created and currently maintains the founding implementation and specification. Open source code and public rules demonstrate implementation openness; they do not by themselves establish independent governance or third-party conformance. Those are separate maturity properties.

CAI should be read as structured measurement, not certification. It does not turn a technical score into proof of security, compliance, business value or organisational capability. Its strength is more specific: the aggregation rule is inspectable, the score can be recomputed from the supplied evidence under the named rubric, and disagreements can be located in the measurement, the fold or the methodology.

2. Claim and scope

2.1 The claim made by this paper

This paper makes a bounded reproducibility claim:

> Given the same eligible evidence and the same rubric version, the CAI aggregation process should produce the same result.

The design makes `rubricVersion` the selector for score semantics. The rubric catalog carries the score-moving `scoring` block, including:

- within-lens OWA decay;
- across-lens OWA decay;
- the critical-contributor gate;
- architecture-surface thresholds and cap;
- baseline band cutlines;
- quality-bar offsets, lens-group factors and clamps.[5][14]

The rubric also freezes score-moving ontology, meaning the structural decisions that determine which measurements are combined with which: the dimension-to-category mapping is frozen where the published catalog carries that assignment, because moving a dimension between categories changes what it is averaged against.[5][6]

A scorer build or package version can be useful provenance. It is not a second authority for the numeric result, and historical scoring semantics should be recoverable from the named rubric rather than reconstructed from an undocumented combination of code and prose.

2.2 What reproducibility does not establish

Repeatable aggregation and valid measurement are different properties, and that distinction bounds everything this paper claims.

A reproducible fold does not establish that:

- the producer represented the repository accurately;
- individual detectors have acceptable precision or recall;
- the selected dimensions measure every intended assurance concept;
- the category and lens ontology is complete;
- the weighting parameters are empirically optimal;
- applicability was inferred correctly;
- two scores are automatically comparable merely because their numeric values are close;
- the score predicts operational or commercial outcomes.

ISO/IEC/IEEE 15939 treats software measurement as a broader process involving information needs, measure definition, application, analysis and evaluation of whether results are valid for their intended use.[22] Kitchenham, Pfleeger and Fenton similarly treat software-measurement validation as broader than the repeatability of a calculation.[23]

The consequence for CAI is straightforward: deterministic aggregation can be established from the implementation and tests. Stronger empirical claims about robustness, construct validity or decision usefulness require the study that actually tests them.

2.3 CAI as a composite aggregation problem

CAI belongs to the broader class of systems that combine multiple indicators into a summary measure. The OECD/JRC handbook on composite indicators identifies missing-data treatment, weighting, aggregation and model assumptions as consequential design choices, and recommends making those choices explicit and subjecting them to robustness analysis where stronger claims depend on them.[17] The JRC's current ten-step guidance retains the same structure, including data treatment, weighting, aggregation and sensitivity analysis.[18]

Those sources do not validate CAI's ten lenses, its OWA choices or its band cutlines; they support a narrower proposition, that when many inputs are compressed into one index the construction choices matter and should be inspectable. That narrower proposition is the methodological problem this paper addresses: how CAI answers it, and where the answer stops.

2.4 Open method and producer boundaries

The current CAI architecture separates three layers.[2][3][4]

1. **Evidence production.** A producer analyses source code and repository artefacts and emits score-bearing evidence.
2. **Deterministic aggregation.** The CAI reference scorer folds eligible evidence under the named rubric.
3. **Advisory interpretation.** Human or language-model judgement may explain findings, prioritise work or add contextual observations without entering the deterministic numeric fold.

The separation is what makes the open parts meaningful and what bounds them. The reference scorer and rubric are open and inspectable, so layer two can be examined by anyone.[1][3] The scorer does not independently reproduce the producer's source-to-evidence measurement in layer one. An open fold does not make a proprietary measurement engine open, and a reproducible result does not imply that the upstream measurement has been independently reproduced.

2.5 Intended use

CAI is intended to support:

- structured technical review;
- longitudinal tracking under a named rubric;
- inspection of lens-level strengths and weaknesses;
- evidence-led conversations between technical and non-technical recipients;
- recipient recomputation of a published aggregation when the evidence is available;
- comparison where the rubric, subject, scope and evidence context make the comparison meaningful.

It is not intended to serve as:

- certification of compliance or security;
- a substitute for professional judgement;
- a penetration test or dynamic application security test;
- a measure of individual or team performance;
- a direct estimate of business value;
- a universal ranking across repositories without regard to scope, applicability or evidence context.

The second list is not a disclaimer attached to the first. Each item on it names a question that would require evidence CAI does not collect, or a process outside what it performs, so the boundary follows from what the method measures rather than being a caution added afterwards.

3. Terms and data model

Stable terminology matters because the CAI contains several distinct levels of aggregation and several distinct ways for evidence not to participate.

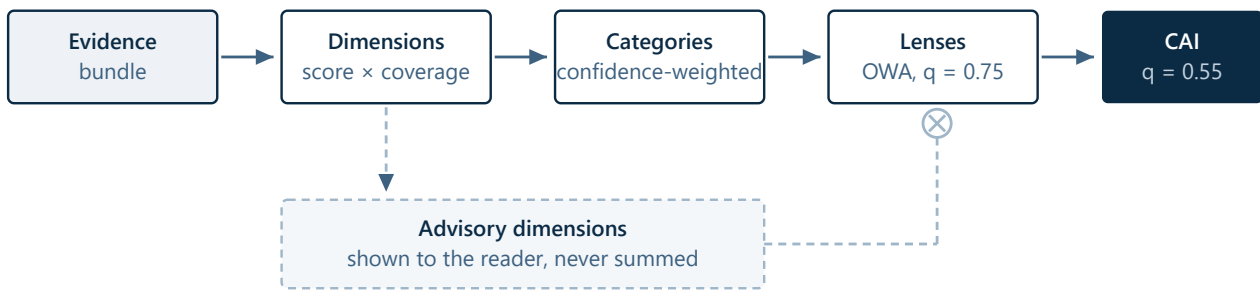
Term	Meaning
Evidence bundle	A machine-readable record containing the score-bearing evidence and the rubric version under which it is folded.
Rubric version	The versioned rule identity that selects the scoring semantics for the result.
Catalog entry	A criterion defined in the rubric catalog. A scored dimension may also carry a category assignment frozen by the catalog.
Scored dimension	A deterministic criterion whose result is eligible for numeric aggregation.
Advisory dimension	A criterion retained for interpretation but excluded from the numeric fold.
Meta-dimension	A criterion that enters a lens directly rather than first being combined inside a category. A null-scored meta-dimension does not participate.
Category	A group of related deterministic dimensions combined before lens aggregation.
Lens	A major assurance perspective represented by a score when enough score-bearing evidence resolves.
Coverage	The fraction of a measured dimension's target surface represented by its evidence. The scorer uses $\text{effective} = \text{score} \times \text{coverage}$.
Confidence	The weight used when measured deterministic dimensions are combined inside a category.
Omission	A dimension, meta-dimension or lens that is not present in the numeric fold. The evidence bundle records that an item is absent, not why.
Survey completeness (surveyFit)	Signed, descriptive, non-scored information reporting how many lenses applied to the codebase at all and how many of those resolved.
Baseline band	The band obtained by applying the rubric's baseline cutlines to a numeric score.
Critical gate	A rubric-carried threshold that can cap a lens's displayed band without changing the numeric lens score.
Coherence cap	A headline-band constraint that prevents the headline band from out-promising the weakest measured category by more than the permitted amount.

The important distinction is between **score-bearing evidence** and **descriptive completeness information**. A thin survey is not automatically a bad codebase, so CAI keeps `surveyFit` outside the score; at the same time a recipient should be able to see when a headline was produced from a thin applicable surface, so CAI keeps it beside the score. The score answers what the resolved evidence says. Completeness answers how much of the applicable evidence resolved. Neither question is served by folding one into the other.[7]
[16]

3.1 The aggregation path

The numeric path is:

SCORED PATH



A dimension that cannot be measured leaves the path entirely. It is recorded with its reason, and never enters as a zero.

Figure 1. The fold. Evidence becomes dimension scores, dimensions become categories, categories become lenses, and ten lens scores become one headline. Advisory judgement travels beside the path and never joins it.

Two things sit outside that straight line. Meta-dimensions enter at the lens stage directly, skipping the category step. Advisory dimensions and advisory meta-dimensions remain available to reporting but are excluded from the numeric fold entirely, so they appear nowhere on the path above.[6][7]

3.2 Score scales and category calculation

The scorer represents deterministic dimension scores on a 0 to 10 scale and category, lens and headline scores on a 0 to 100 scale.[6][7]

For a deterministic dimension `i`:

```
effective_i = score_i * coverage_i
```

where:

- `score_i` is in `[0, 10]` ;
- `coverage_i` is in `[0, 1]` ;
- absent coverage defaults to `1.0` .[7]

Dimensions are combined inside a category using **confidence**, not coverage, as the averaging weight:

```
Category_j =  
  10 * (  
    sum of (effective_i * confidence_i)  
    /  
    sum of confidence_i  
  )
```

for the non-advisory dimensions assigned to category `j` .[6]

The two quantities do different work inside the same calculation: coverage changes the effective value of a measured dimension, and confidence changes how strongly that measured value contributes to its category. A dimension measured over half its target surface and a dimension given half the category weight will both pull a category score down, but they are not the same claim about the evidence, and the fold keeps them apart.

A worked category makes the two roles concrete. The figures are synthetic and illustrate the arithmetic only.

Dimension	Score	Coverage	Effective	Confidence
D1	8.0	1.00	8.0	0.9
D2	6.0	0.50	3.0	0.8
D3	9.0	1.00	9.0	0.4
D4	4.0	1.00	4.0	1.0

The category score is:

$$\begin{aligned}
 \text{Category} &= 10 * (8.0*0.9 + 3.0*0.8 + 9.0*0.4 + 4.0*1.0) / (0.9 + 0.8 + 0.4 + 1.0) \\
 &= 10 * 17.2 / 3.1 \\
 &= 55.48
 \end{aligned}$$

A plain mean of the four raw scores would give 67.5. The difference is the two quantities doing their separate work: D2's half coverage halves what it is worth before any weighting, and D3's high score arrives with the lowest confidence, so the strongest measurement in the category is also the one that counts least toward it.

If the category's confidence sum is zero or no score-bearing dimensions resolve, the category has no numeric result.[6] Numerically, a zero-confidence dimension contributes no category weight and does not trigger the critical gate, because that gate only considers measured contributors with confidence greater than zero.[6] The result is that a zero-confidence dimension behaves in the arithmetic exactly as an absent one does, and the evidence-model documentation accordingly says such a dimension should be absent rather than encoded as a measured zero-confidence result.[7] Across every one of these cases, no missing dimension is replaced with a synthetic failure score.

4. The ten-lens model

The current CAI reference implementation organises evidence through ten lenses.[8]

#	Lens	Applicability class	Primary concern
1	Code Health	Core	Maintainability, complexity, duplication and local code quality
2	Architecture	Core	Structural boundaries, coupling, dependencies and analyzable architecture
3	Maturity	Core	Documentation, maintainership signals, conventions and lifecycle discipline
4	Production Readiness	Core	Testing, dependency condition, operational safeguards and release readiness
5	Security and Compliance	Core	Secrets, insecure patterns, dependency exposure and code-visible control posture
6	Domain Modelling	Model-aware	Domain boundaries, aggregates, identifiers and separation of domain concerns
7	Event-Driven	Model-aware	Event contracts, publication and consumption patterns, and asynchronous design
8	Event Sourcing	Model-aware	Event-store discipline, replayability, projections and event-sourced consistency
9	Accessibility	Model-aware	Code-visible implementation of accessibility requirements
10	Performance	Model-aware	Performance-sensitive patterns, resource use and measurable efficiency controls

The five core lenses are defined as the standard survey surface. The five model-aware lenses become relevant when repository characteristics make the corresponding model visible.[8]

Core · scored on every codebase



Model-aware · scored only where the model is visible in the code



A lens that does not apply is recorded as not applicable and takes no part in the fold. Its share of the weight is re-normalised across the lenses that did resolve, so absence neither penalises nor quietly flatters.

Figure 2. The ten lenses. The five core lenses define the standard survey surface. The five model-aware lenses become relevant only when repository characteristics make the corresponding model visible.

A command-line utility should not receive an Accessibility failure merely because Accessibility exists in the catalog. Conversely, a repository that actually contains the relevant model should not gain a stronger headline merely because that lens failed to resolve and disappeared without explanation. Both are failures of the measured set, which is what the headline is computed over.

4.1 Applicability and survey completeness

Applicability is part of the measurement problem because the scored set affects the headline. Which lenses resolve determines what the fold operates on, so a decision about applicability is a decision about the number.

The evidence bundle records that an item is missing and nothing more: a missing dimension tells the recipient only that it is missing, not whether it was unsupported, not attempted, unresolved or deliberately omitted. Nothing in the evidence bundle supports a richer account of absence than that.

The signed delivery surface exposes a higher-level descriptive completeness object, `surveyFit`, whose two central fields are:[7][16]

- `depthApplicable` : the number of lenses that applied to this codebase at all;
- `depthFired` : the number of those applicable lenses that actually resolved.[7][16]

The difference between the two exposes blind spots in the survey. `surveyFit` is deliberately **non-scored**: the reference scorer does not read it and it cannot move the CAI headline.[7][16]

This creates a useful separation:

> **The score describes the evidence that resolved.** `surveyFit` **describes how complete the applicable reading was.**

That distinction prevents survey incompleteness from being converted automatically into a code-quality penalty while still making the evidential basis visible to the recipient.

4.2 Catalog size and versioning

The number of catalog entries is version-specific. It should not be treated as a permanent characteristic of CAI or used as a proxy for method quality, since a catalog can grow by subdividing existing criteria without measuring anything new.

Each released rubric should make its own catalog inspectable, including the dimensions, categories, lens assignments and score-moving configuration carried by that version.[1][5] Dynamic counts belong to release metadata, not to the invariant definition of the method.

5. Aggregation method

5.1 Category aggregation

Deterministic dimensions are grouped into categories.[6][9] The category layer reduces the influence of arbitrary catalog granularity because several related dimensions are combined before they enter the lens fold.

For each category, the scorer:

1. excludes advisory dimensions;
2. calculates each measured dimension's `effective = score × coverage` ;
3. combines effective scores using confidence as the category weight;
4. scales the result from 0 to 10 onto 0 to 100.[6]

Steps two and three use their two quantities for different jobs, and the difference is easy to lose. Coverage scales what a dimension is worth, because a measurement of half the target surface is half a measurement. Confidence scales how much that worth counts against its neighbours inside the category. A category is unscored when its non-advisory confidence sum is zero, since there is then nothing left to weight the average with.[6]

Not measured	Partly covered	Scored as zero
Nobody looked, or nothing could be read.	Half the target surface was reachable.	Absence treated as failure.
Leaves the fold entirely, carrying its reason.	Stays in, worth $\text{score} \times \text{coverage}$.	Punishes a codebase for what nobody examined.
Effect on the score: none	Effect: reduced, not removed	CAI does not do this

Survey completeness travels beside the score in a signed, non-scored object, so a thin reading is visible without the codebase being blamed for

Figure 3. Missing evidence and partial coverage are different problems with different answers, and neither is a zero. Coverage scales what a dimension is worth; absence removes it from the fold and says so.

The dimension-to-category mapping is score-moving, because changing a dimension's category changes which other dimensions it is averaged with, and so what its own value is diluted or sharpened by. Current rubric catalogs can therefore freeze that mapping, and the scorer rejects a bundle that contradicts the published assignment for a rubric that carries one, which stops the grouping from drifting quietly between one result and the next.[5][6]

5.2 Meta-dimensions

Meta-dimensions bypass categories and enter the lens fold directly.[6][9] A measured non-advisory meta-dimension occupies one lens-level position beside the category scores, and positions are what the lens fold assigns weight to.

That is a meaningful structural choice. Splitting one concept into several direct meta-dimensions can give it more rank positions. Combining several dimensions inside one category can reduce the number of positions by which they influence the lens. The shape of the ontology is therefore itself a form of weighting: two catalogs could contain the same criteria, measure the same repository identically, and still produce different lens scores purely by grouping those criteria differently.

This is not a defect hidden by the implementation. It is part of the ontology and should remain inspectable, which is why changes to score-moving ontology belong in a new rubric version.[5]

5.3 Worst-first ordered weighting

CAI combines category scores and eligible meta-dimensions inside a lens using an ordered weighted average. Measured lens scores are then combined into the headline using the same operator family with a sharper decay.[3][6]

The OWA family was introduced by Ronald Yager in 1988 as a method for aggregating multiple criteria according to their ordered positions rather than fixed source identities.[19] Subsequent literature describes a large family of OWA operators and parameterisations, of which CAI uses one particular worst-first parameterisation.[20]

Let the inputs be sorted from weakest to strongest:

$$x_{(0)} \leq x_{(1)} \leq \dots \leq x_{(n-1)}$$

For decay constant q :

$$w_r = q^r / (q^0 + q^1 + \dots + q^{(n-1)})$$

and:

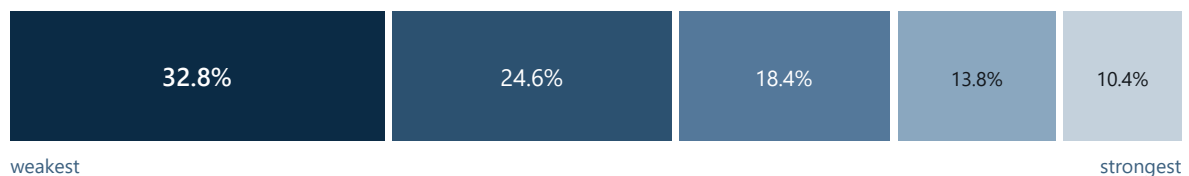
$$OWA_q(x) = \sum_{r=0}^{n-1} (w_r * x_{(r)})$$

Because $0 < q < 1$, the weakest input receives the largest rank weight, and each successive input receives q times the weight of the one below it, so influence decays geometrically as the inputs improve.

The current default scoring parameters are:

- $q = 0.75$ within a lens;
- $q = 0.55$ across measured lenses.[6][14]

Within a lens · $q = 0.75$ · five contributors



Across the measured lenses · $q = 0.55$ · ten lenses



Figure 4. Rank weight, not source weight. Weight is assigned by where a score sits in the order, weakest first. The sharper decay across lenses is why the headline answers to its weakest measured lens more strongly than any lens answers to its own weakest contributor.

The sharper decay across lenses means the headline is governed by its weakest measured lens more strongly than a lens is governed by its own weakest contributor. Those values are not properties of OWA itself. They are **normative CAI design choices**, now carried as rubric data. The source base establishes what the parameters are and how they affect the fold. It does not establish that those exact values are empirically optimal.

5.4 Ties and ordering

The reference scorer uses stable ordering for equal values, so equal scores retain input order while OWA rank weights are assigned, which means two bundles carrying the same values in the same order receive the same assignment of weights rather than one that depends on how a particular sort was implemented.[6]

Stable ordering removes one avoidable source of implementation variation. Other sources remain: cross-implementation conformance still depends on a canonical numeric profile, including operation order, rounding and serialised output rules.

5.5 Architecture surface rule

Architecture requires enough analyzable surface for structural metrics to carry meaningful signal. Coupling, boundaries and dependency structure are relations between units, so a repository with almost nothing to relate does not produce a weak architecture result. It produces no measurable architecture at all.

Under the current default scoring parameters:

- zero analyzable projects drops the Architecture lens;
- a thin but non-zero architecture surface can cap the Architecture lens at 69;
- the default full-surface thresholds are two analyzable projects or 1,500 production lines of code.[10][14]

Dropping the lens rather than scoring it low follows the same principle as the rest of the fold: a measurement that cannot be made is not a failing grade. A dropped Architecture lens leaves the headline to be computed over the lenses that did resolve.

These values are no longer merely constants in one scorer build. They are rubric-carried scoring parameters, which matters for historical replay: if the thresholds or cap change in a way that can move a result, the scoring semantics change with the rubric version.[5][14]

5.6 Headline aggregation and renormalisation

The headline is calculated over the lens scores that actually resolve.[6]

For the measured lens set M :

CAI = OWA with acrossLensQ over every measured lens score L_k in M

Under the current defaults, `acrossLensQ = 0.55`.[14] If a lens is absent, it is not assigned a synthetic zero, and the OWA weights are derived over the remaining measured set and sum to one.

That does **not** mean absence is numerically neutral, and the reason is positional: removing one input moves every stronger input one position toward the weakest end, where the weight is larger. Changing the measured set therefore changes the ordered fold, so omission can move the headline either upward or downward depending on which item disappears and where it would have ranked. External composite-indicator guidance makes the general point that missing-data treatment can affect reliability and the resulting composite.[21] CAI's actual omission semantics come from the CAI implementation, not from that external literature.[6][7]

Where `surveyFit` is available, a recipient can separately see how much of the applicable survey surface resolved, which is what lets a reader tell a headline computed over everything applicable from a headline computed over part of it.[7][16]

5.7 Absence, confidence and partial coverage

The implementation contains several different non-full-measurement states. They should not be collapsed into a single word such as *abstention*.

Current state	Numeric behaviour
Dimension omitted	Does not participate. The evidence bundle records the absence and not its reason.
Dimension with confidence 0	Carries zero category weight and cannot trigger the critical gate. Documentation says such a dimension should be absent.
Dimension with coverage below 1	Participates with <code>effective = score × coverage</code> ; partial coverage can reduce the category contribution and can contribute to a critical gate when the effective score is below the rubric threshold.
Null meta-dimension	Does not participate.
Absent lens	Does not participate in the headline; the remaining OWA weights are derived over the measured set.
surveyFit gap	Descriptive only. Shows applicable lenses that did not resolve and never changes the score.

This asymmetry is intentional in one respect: **missing measurement is not automatically scored as failure**. A survey that could not reach something does not manufacture a verdict about it.

It also creates a real methodological question. A weak partial measurement can lower a result while a completely absent item does not participate, which means thoroughness and score are not independent of one another: two repositories can carry the same headline under the same rubric while one of them was read further and absorbed the reduction that partial coverage produces. That makes reporting of the measured set and survey completeness consequential rather than decorative, and it makes systematic missingness and coverage analysis a useful later robustness study.

Documenting how the fold works requires no such study. The standard would need one before claiming that the scoring method is empirically robust to plausible missingness and coverage patterns.

6. The advisory firewall

CAI separates deterministic score-bearing inputs from advisory judgement. The two kinds can sit beside each other in the same report, and only one of them reaches the arithmetic.[4][6][7]

A deterministic scored dimension can enter the numeric fold when it is present and eligible under the scorer. An advisory dimension may remain in the evidence and report but is excluded from category, lens and headline arithmetic, and the same exclusion applies to advisory meta-dimensions.[6][7]

Three useful consequences follow from that exclusion, and they compound.

1. Advisory language-model variability cannot move the deterministic CAI.
2. Advisory interpretation can be removed or replaced without changing the numeric result.
3. A numeric result remains challengeable on the deterministic evidence and scoring rule alone.

The firewall controls one source of variation, and only one. It does not establish that deterministic detectors are accurate: a detector that runs the same way every time can be consistently wrong every time, and the fold will carry that error forward with the same determinism it carries everything else. Keeping language-model judgement out of the number removes a source of noise from the result without adding any evidence that the result is right. That is a measurement-validation question rather than an aggregation question, and nothing in the aggregation method can settle it.

6.1 Why the firewall is a design choice rather than a validation result

The firewall answers:

> Which kinds of evidence are allowed to move the number?

It does not answer:

> How accurate is every deterministic or advisory measurement?

Those are different questions and they require different evidence. The first is settled by inspection: the current CAI implementation makes advisory status explicit and excludes advisory dimensions from the arithmetic, which is an implementation fact that anyone holding the published scorer and its tests can check for themselves.[4][6][7]

The second is not settled by the design at all. Whether every dimension has been placed on the correct side of the firewall is a separate methodological and empirical question, and it carries a governance consequence: changing a score-moving classification should be governed as a scoring-method change rather than quietly adjusted after performance results are known. A boundary that can be moved once the results are in is not a boundary.

The public method should therefore be judged on two levels:

- whether the implementation obeys the declared firewall;
- whether later validation evidence supports the classification and measurement quality of the inputs.

Only the first can be tested today. The second waits on the relevant validation work, and until that work exists what CAI has achieved here is narrow: it has made the boundary explicit and testable, not demonstrated that every input sits on the correct side of it.

7. Reporting: score, bands, context and overrides

A CAI report should keep the numeric result, banding rules, completeness information and non-scored interpretation distinguishable. One principle runs through all three of CAI's interpretation mechanisms: the number does not move. A band is a label placed on a number for a reader, and CAI adjusts that label for context and for the worst thing inside it without ever rewriting the measurement underneath.

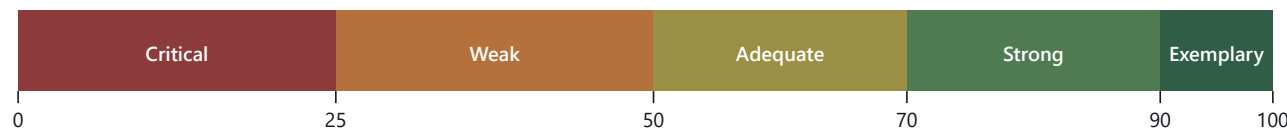
7.1 Baseline CAI band

The rubric carries the baseline band cutlines as scoring data.[5][11][14]

Under the current default values:

Baseline band	Score range
Exemplary	<code>90 <= score <= 100</code>
Strong	<code>70 <= score < 90</code>
Adequate	<code>50 <= score < 70</code>
Weak	<code>25 <= score < 50</code>
Critical	<code>0 <= score < 25</code>

Baseline cutlines, carried by the rubric version



The pin is the reading. The band is a label placed on it, and moving between bands never changes the number underneath.

Figure 5. The baseline band scale. These cutlines are rubric data rather than constants: a band only means something read together with the rubric version that defined it.

These cutlines should not be treated as timeless constants outside the rubric version. The important invariant is that the named rubric identifies the cutlines under which the result is interpreted, which means a band is only meaningful when read together with the rubric that defined it. If score-moving cutlines change, the rubric version changes with them.[5]

7.2 Quality-bar interpretation

The scoring parameters also carry quality-bar shifts.[14] A quality bar does **not** change the numeric score. It changes the cutlines used for lens-level band interpretation according to the repository's criticality tier and the lens group.[12][14]

Current canonical tiers include:

- prototype;
- preview;
- production;
- mission-critical.[14]

The reasoning is that identical code carries different consequences in different settings: a lens result that is acceptable in a prototype may be inadequate in a mission-critical system. The quality bar puts that difference where it belongs, in the interpretation, rather than adjusting the measurement to suit the context. This allows the same numeric lens score to be interpreted under a stricter or more lenient operational context without rewriting the measurement itself.

A report should therefore distinguish:

- **CAI score:** the numeric aggregate;
- **headline baseline band:** the rubric's baseline interpretation of the headline;
- **lens bands:** which may reflect the quality bar;
- **quality bar:** the named context applied to those lens bands.

7.3 Critical-contributor gate

A measured non-advisory contributor below the rubric's `criticalGate` can cap its lens's displayed band at Adequate without changing the numeric lens score, and under the current defaults `criticalGate = 4.0` on the 0 to 10 contributor scale.[6][14]

The gate exists because a band is a summary offered to a reader, and a summary that says Strong while containing a contributor scored below four is making a promise the evidence underneath does not support. Ordered weighting already gives the weakest input the largest share of the lens, but it leaves a diminishing share to the rest, so a lens can still average into a comfortable band while one measured contributor is genuinely poor. Capping the label rather than the number keeps both facts available: the arithmetic continues to report what the evidence produced, and the band stops short of a claim the worst contributor contradicts. The report can expose the specific critical contributors that triggered the cap, so a capped band can be traced to the contributor that caused it.[6]

7.4 Coherence cap

The headline band may be capped so it does not out-promise the weakest measured category beyond the permitted rule.[6][13] This is the same principle one level up: where the critical gate protects a lens band from its worst contributor, the coherence cap protects the headline band from the weakest measured category, and again the numeric headline does not change.

A complete report should make the layers visible:

Output	Purpose
Numeric CAI	Aggregated result
Headline baseline band	Rubric-defined interpretation of the headline
Lens bands	Lens interpretation, including quality-bar effect where applicable
Critical contributors	Explain a lens-level cap
Coherence note	Explain a headline-band cap
Measured-lens set	Show what entered the headline
<code>surveyFit</code>	Show applicable versus resolved survey completeness, when supplied
Advisory evidence	Non-numeric interpretation and recommendations

That separation makes the report more informative without asking one field to carry every kind of meaning. It also gives a reader who disagrees somewhere to put the disagreement: with the measurement, with the cutlines that turned it into a label, with the context those cutlines were shifted for, or with a cap that held the label back.

8. Worked examples and sensitivity

The following examples use synthetic lens scores and the current default headline OWA parameter $q = 0.55$.^[14] They illustrate aggregation behaviour and are not measurements of real repositories.

Five core lens scores are held constant:

Lens	Score
Production Readiness	58
Maturity	64
Architecture	71
Security and Compliance	77
Code Health	82

8.1 Model-aware lens absent

A repository has no scored Domain Modelling lens. The five measured lenses are weighted as follows:

Rank	Lens score	Weight
1, weakest	58	0.4738
2	64	0.2606
3	71	0.1433
4	77	0.0788
5, strongest	82	0.0434

The headline is:

CAI = 63.97

No zero is inserted for Domain Modelling, and no attempt is made here to infer *why* the lens is absent. In a real signed report, `surveyFit` can separately indicate whether a model-aware lens was applicable and whether the applicable surface resolved.^{[7][16]}

8.2 Strong additional lens

A second evidence bundle includes Domain Modelling at 88.

Rank	Lens score	Weight
1, weakest	58	0.4628
2	64	0.2545
3	71	0.1400
4	77	0.0770
5	82	0.0424
6, strongest	88	0.0233

The headline becomes:

CAI = 64.53

The additional strong lens raises the result by approximately 0.56 points. Because it ranks strongest, it receives the smallest rank weight.

8.3 Weak additional lens

A third evidence bundle includes Domain Modelling at 35.

Rank	Lens score	Weight
1, weakest	35	0.4628
2	58	0.2545
3	64	0.1400
4	71	0.0770
5	77	0.0424
6, strongest	82	0.0233

The headline becomes:

CAI = 50.56

The weak additional lens reduces the headline by approximately 13.41 points relative to the five-lens case.

8.4 Interpretation

The examples show three properties of the fold.

- An absent lens is excluded rather than replaced with a synthetic zero.
- A strong additional lens has limited ability to compensate for the weakest measured areas.
- A weak additional lens can have substantial influence because worst-first OWA gives it the highest rank weight.

They also demonstrate why the measured set matters. If the weak lens in the third example disappears from the numeric fold, the headline changes materially, and the score alone does not explain why it disappeared. That is why completeness and applicability information belong beside the result rather than being silently inferred from it.

8.5 Sensitivity as a higher-rigor research path

The implementation is sufficient to state and defend **how CAI scores**. A different claim, that the result is robust across plausible alternative modelling choices, requires sensitivity evidence, which has yet to be produced.

The OECD/JRC composite-indicator literature recommends sensitivity analysis because choices about missing data, weighting and aggregation can materially affect composite outputs.[17][21] The current JRC guidance similarly treats sensitivity analysis as the method for identifying assumptions to which the result is particularly sensitive.[18]

A CAI-specific robustness study should consider, at minimum:

- alternative values of the within-lens and across-lens OWA parameters;
- equal-weight and minimum-score comparators;
- category splitting, merging and membership changes;
- meta-dimension structure;
- omission and partial-coverage patterns;
- architecture-surface parameters;
- critical-gate behaviour;
- band cutlines and quality-bar shifts;
- repository classes with different measured-lens sets.

The need for such a study depends on the claim being made. It is not a prerequisite for documenting the normative method, and it becomes necessary when CAI wants to claim empirical robustness, parameter stability or a known meaningful difference between nearby scores.

9. Reproducibility, versioning and verification

9.1 Score identity

For the current CAI scoring design, `rubricVersion` is the score-semantics selector. The rubric named in a result therefore decides how that result was computed, not merely which criteria were assessed.[5][14]

The named rubric can carry:

- dimension-to-category assignments;
- within-lens OWA parameters;
- across-lens OWA parameters;
- the critical gate;
- architecture-surface parameters;
- baseline band cutlines;
- quality-bar offsets, factors and clamps.[5][14]

Every item on that list can move a number, which is why they belong to the versioned object rather than to whichever build of the scorer happened to run. Catalogs published before the `scoring` block resolve to `ScoringParameters.Default`, preserving the historical values under which they were scored, so an older result remains interpretable under its own rules instead of being silently reinterpreted under newer ones.[5][14]

A scorer build, CLI build or package version may still be useful provenance, and knowing which code produced a result is worth recording. It should not be presented as a second authority for the score. One result answering to two rule authorities is a result nobody can check, because a disagreement between them has nowhere to be located.

The scoring identity is conceptually:

```
eligible evidence
      +
rubricVersion
      =
deterministic CAI fold
```

9.2 Public verification surface

The public CAI implementation exposes the reference scorer, CLI and web/API verification surfaces, which puts the check in the hands of the recipient rather than only the issuer.[1] A recipient with an evidence bundle can recompute the result rather than accepting the headline as an opaque assertion, and the public rubric archive also makes historical rule identity inspectable, so a result issued under an earlier rubric can still be checked against the rules that actually produced it.[1][5]

The delivery-package design goes further by carrying subject, producer, rubric, evidence and verdict inside a signed artifact, so that the claim and the material needed to test it travel together rather than being assembled by the recipient from separate sources.[15]

The full provenance and signature boundary belongs to the companion paper on recomputation, provenance and measurement boundaries. The relevant point here is narrower:

> the construction described in this paper can be exercised from the evidence and named rubric.

9.3 Canonical output and current tolerance

The reference scorer is deterministic, but the current verification implementation separately accepts a default headline tolerance of plus or minus 0.5.[6][15] Those are different properties, and conflating them would misdescribe both. The tolerance is an acceptance rule in the current verification profile, and it should not be read as a statement that CAI itself is only approximately deterministic: a verifier that tolerates a small discrepancy has chosen how strict to be about matching, which is not the same as the fold being imprecise.

The delivery format already rounds embedded verdict values to fixed output precision before signing.[15] A mature cross-implementation conformance profile should make the canonical numeric behaviour sufficiently explicit that independent implementations can be tested against the published value and band without hiding a scoring-semantic disagreement inside the delivery tolerance. That is not a hypothetical tidiness concern. Under the current profile, two implementations that genuinely disagree

about the scoring semantics by less than the tolerance would pass verification as though they agreed, which is why the conformance vectors matter.

9.4 Historical replay

The August 2026 scoring-parameter change materially improved historical replay. Score-moving constants and cutlines that previously lived only in scorer code now travel in the rubric catalog, and older catalogs without the block resolve to the historical defaults, which means an archived rubric can identify its own scoring semantics rather than depending on a version-dispatch switch in the scorer.[5][14]

Reading the semantics of an old result no longer requires reconstructing them from a scorer build alongside the rubric, because the rubric now states them itself. Demonstrating that those semantics still reproduce their original results on archived material is a further step, and the remaining higher-maturity step is operational conformance evidence: versioned input-to-output vectors, which publish an evidence bundle beside the headline, lenses and bands it should produce, and later genuinely independent implementations that can be tested against those vectors without importing the reference scorer.

9.5 Evidence access and integrity

Recomputation requires the score-bearing evidence, so whether a recipient can check a result depends on what they are sent, not on the scoring method. A signed delivery package can carry that evidence together with the subject, producer, rubric and claimed result, allowing a recipient to verify the artifact and recompute the fold from the embedded evidence.[15] Where such a package is supplied, that check is available to whoever holds it.

Where it is not, the check is not available at all. If evidence is withheld, a recipient cannot independently perform the same evidence-to-score check merely from a headline number, and no property of the scoring method compensates for that: determinism establishes that the same evidence under the same rubric yields the same result, not that anyone besides the issuer will ever hold the evidence. That is an access limitation rather than a failure of deterministic scoring, and the two are worth keeping apart, because a score published without its bundle can look like a weakness in the method when it is a decision about disclosure.

9.6 Survey completeness travels separately from the score

The evidence bundle can carry signed `surveyFit` information.[7][16]

For survey depth:

```
depthApplicable = lenses that applied to this codebase at all
depthFired      = applicable lenses that resolved
```

The difference between the two is a descriptive blind-spot count: the applicable surfaces the survey was entitled to read and did not resolve. This information does not enter the scorer, so recording a gap costs the subject nothing in score, which is what makes the gap safe to report.[7][16]

That is an important design choice because two identical numeric headlines can rest on different evidential breadth. One may summarise a survey that resolved everything it was entitled to read; the other may summarise a fraction of the same applicable surface and arrive at the same number. CAI can expose that difference without turning a thin survey into a lower code-quality score, which keeps completeness answerable as its own question instead of trading it against the result. A report should therefore avoid presenting the headline as though it describes a complete survey when signed completeness information shows otherwise.

10. Limitations and research requirements

The following table separates current implementation facts from stronger claims that would require additional evidence. Each row names something the current implementation establishes and, beside it, the claim that same evidence does not reach. The middle columns are not a defect list; they are the boundary of what this paper is entitled to say.

Question	What current CAI already establishes	What remains open	Evidence needed for the stronger claim
Measurement accuracy	Deterministic evidence can be folded consistently.	A consistent fold can preserve an inaccurate upstream measurement.	Precision, recall and error analysis appropriate to the producer and criterion.
Construct validity	The lens, category and rubric structure is inspectable.	Inspectability does not prove that the ontology captures every relevant assurance concept.	Expert review, construct-validation work and, where relevant, outcome studies.
Applicability	Model-aware lenses can be excluded from the measured set; <code>surveyFit</code> can report applicable versus resolved lens depth.	The evidence bundle does not explain every dimension-level omission or prove applicability decisions are correct.	Labelled applicability corpus and error analysis if CAI later claims applicability accuracy.
Missingness and partial coverage	Omission, confidence-zero, partial coverage, null meta-dimensions and absent lenses have defined numeric behaviours.	Different missingness patterns can move the score and may create interpretation asymmetry.	Corpus-based missingness and coverage robustness study for empirical robustness claims.
Meta-dimension influence	Direct-entry meta-dimensions and categories are visible in the scoring structure.	Ontology structure creates implicit influence.	Structural and sensitivity analysis of grouping choices.
OWA parameters	The current rubric identifies the exact within-lens and across-lens parameters.	The current values are normative choices, not demonstrated optima.	Sensitivity and calibration studies for claims of robustness or optimality.
Band cutlines and gates	Cutlines, critical gate and quality-bar rules are rubric-carried scoring semantics.	Their normative status does not establish that the thresholds correspond to externally validated outcome boundaries.	Outcome or decision-usefulness studies if such external meaning is claimed.
Cross-repository comparison	Scores can be compared under a known rubric with visible subject, scope and evidence context.	A numeric difference alone does not establish that repositories are equivalent comparison objects or that a small point difference is meaningful.	Comparison design and empirical studies tailored to the intended use.
Numeric conformance	The reference scorer is deterministic and the delivery profile has defined output precision.	The current verifier still accepts plus or minus 0.5.	Published conformance vectors and independent implementations.
Historical replay	Rubric-carried scoring parameters now pin score-moving semantics, with historical defaults for older catalogs.	Operational replay coverage can still be strengthened.	Archived evidence/rubric fixtures and continuous replay tests.
Governance independence	Specification, rubric and implementation are publicly inspectable.	Open implementation is not independent governance.	A governance structure with real independent authority and participation.

These are not all publication blockers. The current implementation facts can be stated directly because the rubric, scorer and evidence model establish them, while the open research questions determine the ceiling of stronger claims CAI may later want to make.

That distinction is important, and it cuts both ways. A technical standard should not weaken a known implementation fact merely because a more ambitious validation programme remains possible, and it should not present an open question as settled because the validation programme is planned.

11. Conclusion

The Codebase Assurance Index solves a difficult compression problem: how to turn a large, uneven body of technical evidence into one useful result without hiding the construction that produced it. Its present architecture is stronger because the important choices are visible.

Three of those choices govern what reaches the number at all. Evidence is grouped into categories and then into ten lenses, so that the influence of any one criterion is bounded by the structure it sits in rather than by how many criteria the catalog happens to contain. Advisory judgement is kept outside the deterministic number, which means the arithmetic can be re-run without the interpretation and the interpretation can be revised without disturbing the arithmetic. And missing items are not replaced with synthetic failures: an omitted dimension, a null meta-dimension or an absent lens does not participate, so a survey that did not reach something does not silently become a finding against the codebase.

Two further choices govern how the number behaves once the evidence is in. Weak measured areas receive greater rank weight through a published worst-first OWA rule, which gives added strength limited ability to compensate for the weakest measured area no matter how much of it arrives. Coverage changes a measured dimension's effective value and is not confused with confidence, which keeps two different questions apart: how much of the target surface the evidence represents, and how much weight that measured value should carry among its neighbours. Where the survey itself was thin, completeness can travel beside the score through signed, non-scored `surveyFit` rather than inside it.

Most importantly, the score-moving aggregation parameters now travel with the rubric version. The ruler is part of the versioned object, which is what turns the preceding choices from a description of one scorer build into a deterministic contract:

> same eligible evidence + same rubric version = same result

The contract is inspectable, and it can be tested. A recipient holding the evidence bundle can run the fold and check the published headline instead of accepting it as an assertion, and a disagreement can then be located in the measurement, the fold or the methodology rather than argued in general terms.

It is also appropriately bounded, and that boundary is the point. A recipient who recomputes the result has verified the fold from evidence to score; that recipient has not, by that act alone, independently re-measured the repository. Keeping those two claims separate does not make CAI weaker. It makes the assurance property precise, and a precise property is one that can be tested where it is actually weak.

The next stages of maturity can then build on a stable foundation: conformance vectors, independent implementations, missingness and sensitivity studies, applicability validation, external outcome research and independent governance. Each of those can raise the ceiling of what CAI is entitled to claim. None of them is a substitute for the central achievement already present in the method, which is an open, versioned and inspectable rule for turning structured codebase evidence into a deterministic assurance result.

References

CAI implementation references 1 to 16 are pinned to the technical state reviewed for this draft, through commit `1d557c36faa6f7650cfe6746db01665ad2ac9bfc`, unless a more specific commit is named. External references are used only for the methodological proposition stated in the paper and do not validate CAI by analogy.

1. Canine Development, *CAI README*, public description of the standard, ten-lens model, rubric versioning, reference scorer and verification surfaces. <https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/README.md>
 2. Canine Development, *Component architecture*.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/docs/architecture.md>
 3. Canine Development, *ADR-0002, deterministic reproducible scoring*.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/docs/adr/0002-deterministic-reproducible-scoring.md>
 4. Canine Development, *ADR-0003, the free/paid firewall*.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/docs/adr/0003-free-paid-firewall.md>
 5. Canine Development, *ADR-0004, versioned frozen rubrics*.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/docs/adr/0004-versioned-frozen-rubrics.md>
 6. Canine Development, *CaiScorer*. Current category, lens and headline fold, OWA weighting, critical gate and verification behaviour.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/CaiScorer.cs>
 7. Canine Development, *EvidenceBundle*. Current evidence model, coverage, confidence, advisory status and `surveyFit`.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/EvidenceBundle.cs>
 8. Canine Development, *Lenses*. Current ten-lens catalog and core/model-aware classification.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/Lenses.cs>
 9. Canine Development, *DimensionCategory*. Category definitions and category-to-lens mapping.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/DimensionCategory.cs>
 10. Canine Development, *ArchitectureSurfaceFloor*. Architecture-surface omission and cap logic.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/ArchitectureSurfaceFloor.cs>
 11. Canine Development, *Band*. Band vocabulary and default accessors.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/Band.cs>
 12. Canine Development, *QualityBarBands*. Lens-level quality-bar interpretation.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/QualityBarBands.cs>
 13. Canine Development, *BandCoherence*. Headline coherence cap.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/BandCoherence.cs>
 14. Canine Development, *ScoringParameters*. Rubric-carried OWA parameters, critical gate, architecture-surface parameters, band cutlines and quality-bar rules.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/src/Cai.Scoring/ScoringParameters.cs>
 15. Canine Development, *CAI-delivery package: format specification*, version 1.0, Draft. Current delivery and verification profile, including fixed output precision and plus-or-minus 0.5 reproduction tolerance.
<https://github.com/CanineCC/CAI/blob/1d557c36faa6f7650cfe6746db01665ad2ac9bfc/docs/spec/cai-delivery-package.md>
 16. Canine Development, commit `fc616ce3bf981194e0e351547a234aa6b22b3645`, 22 August 2026, introducing signed descriptive non-scored `surveyFit`. <https://github.com/CanineCC/CAI/commit/fc616ce3bf981194e0e351547a234aa6b22b3645>
1. OECD and European Commission Joint Research Centre, *Handbook on Constructing Composite Indicators: Methodology and User Guide*, OECD Publishing, 2008, DOI [10.1787/9789264043466-en](https://doi.org/10.1787/9789264043466-en). Used for the general role of framework, missingness, weighting, aggregation and sensitivity analysis in composite indicators.
 2. European Commission Joint Research Centre, *10 Step Guide*, Competence Centre on Composite Indicators and Scoreboards. Used as current methodological guidance on theoretical framework, data treatment, weighting, aggregation and sensitivity analysis.
 3. Ronald R. Yager, "On ordered weighted averaging aggregation operators in multicriteria decisionmaking", *IEEE Transactions on Systems, Man, and Cybernetics* 18(1), 1988, pp. 183-190, DOI [10.1109/21.87068](https://doi.org/10.1109/21.87068). Primary historical reference for the OWA operator family.
 4. Ali Emrouznejad and Marianna Marra, "Ordered Weighted Averaging Operators 1988-2014: A Citation-Based Literature Survey", *International Journal of Intelligent Systems* 29(11), 2014, pp. 994-1014, DOI [10.1002/int.21673](https://doi.org/10.1002/int.21673). Used to establish that OWA is a broader operator family and literature rather than CAI's specific parameterisation.

5. European Commission Joint Research Centre, *Step 3: Imputation of missing data* and *Step 8: Sensitivity analysis*, Competence Centre on Composite Indicators and Scoreboards. Used only for the general proposition that missing-data treatment and modelling assumptions can affect a composite result.
6. ISO/IEC/IEEE 15939:2017, *Systems and software engineering: Measurement process*. Current published edition as of 23 August 2026. Used for the distinction between measurement process, analysis and intended-use validity. This citation does not imply CAI is ISO-conformant or ISO-validated.
7. Barbara A. Kitchenham, Shari Lawrence Pfleeger and Norman E. Fenton, "Towards a Framework for Software Measurement Validation", *IEEE Transactions on Software Engineering* 21(12), 1995, pp. 929-943, DOI [10.1109/32.489070](https://doi.org/10.1109/32.489070) . Used as software-measurement validation context, not as CAI validation.