

Claim Strength Taxonomy for c v0.1

Assertion-strength, evidence-boundary, and anti-claim-laundering profile for c-class
AI architecture

Kotov Ivan
Bruxelles, 2026

Status:	Draft hardening profile v0.1
Date:	2026-06-02
Layer:	c = a + b / SER / L4 / Beacon / AGL / ARL / VXCX / L4 Witness / claim hygiene / hardening
Document ID:	Claim_Strength_Taxonomy_for_c_v0_1
Document class:	hardening profile / claim taxonomy / assertion-boundary control artifact
Assertion class:	control-layer artifact; does not upgrade any architecture, capability, safety, personhood, legal, or economic claim beyond its evidence
Primary subject:	claims made about c, c-class systems, entity-like AI presences, local AI nodes, persistent agents, clean experience, and temporal AI presence
Primary rule:	no evidence class may silently prove a different claim class.

Table of Contents

0. Executive definition	5
1. Purpose	5
2. Scope	6
2.1 In scope	6
2.2 Out of scope	6
2.3 Non-claim	6
3. Corpus dependencies and precedence	7
3.1 Parent layers	7
3.2 Precedence rule	7
3.3 No redefinition rule	7
4. Corpus bridges	8
4.1 Explicit bridge	8
4.2 Quiet bridge I — information theory	8
4.3 Quiet bridge II — cybernetics	8
4.4 Earth paragraph	8
5. Normative keywords	8
6. Definitions	8
6.1 Claim	8
6.2 Evidence	9
6.3 Claim class	9
6.4 Evidence class	9
6.5 Claim laundering	9
6.6 Capability laundering	10
6.7 Governance laundering	10
6.8 Authority laundering	10
6.9 Ontology laundering	10
6.10 Value laundering	10
7. Core rules	10
7.1 Claim class before evidence	10
7.2 Evidence cannot silently cross classes	10
7.3 Defined does not mean implemented	10
7.4 Implemented does not mean tested	10
7.5 Tested does not mean deployed safely	10
7.6 Capability does not imply authority	11
7.7 Continuity does not imply personhood	11
7.8 Locality does not imply sovereignty	11
7.9 Usage does not imply value	11
7.10 Fluency does not imply judgment	11
8. Claim classes	11
8.1 C-CLAIM-ARCH — Architectural claim	11
8.2 C-CLAIM-IMPL — Implementation claim	12

8.3 C-CLAIM-TEST — Tested behavior claim	12
8.4 C-CLAIM-AUDIT — External review / audit claim	12
8.5 C-CLAIM-CAP — Capability claim	13
8.6 C-CLAIM-GOV — Governance / conformance claim	13
8.7 C-CLAIM-CONT — Continuity claim	13
8.8 C-CLAIM-AUTH — Authority / admissible action claim	14
8.9 C-CLAIM-SAFE — Safety claim	14
8.10 C-CLAIM-PERS — Personhood / ontology claim	14
8.11 C-CLAIM-ECON — Economic / clean experience claim	15
8.12 C-CLAIM-DEPLOY — Deployment maturity claim	15
9. Evidence classes	15
10. Claim strength levels	16
10.1 Level does not transfer across classes	16
11. Claim-evidence admissibility matrix	17
12. Required claim templates	17
12.1 Architectural claim template	17
12.2 Implementation claim template	17
12.3 Capability claim template	17
12.4 Governance claim template	17
12.5 Authority claim template	18
12.6 Clean experience claim template	18
13. Public wording guidance	18
13.1 Safe wording	18
13.2 Unsafe wording unless separately evidenced	18
13.3 Required clarification for “new class of AI”	18
14. Claim separation rules	19
14.1 Governance is not capability	19
14.2 Capability is not authority	19
14.3 Continuity is not personhood	19
14.4 Memory is not legitimacy	19
14.5 Locality is not sovereignty	19
14.6 Tokens are not value	19
14.7 Affective behavior is not care by itself	19
15. Post-anchor claim discipline	19
15.1 Survival is not authority	20
15.2 Resemblance is not identity	20
15.3 Lineage is not sovereignty	20
15.4 Re-anchoring is an authority claim	20
16. Clean experience claim discipline	20
16.1 Learning Abstract vs Experience Artifact	20
16.2 No raw-life shortcut	20
16.3 Economic value requires consequence	20
17. Local AI / hardware claim discipline	20

17.1 Hardware enables; it does not authorize	20
17.2 Local cognitive infrastructure claims	21
17.3 Locality must disclose dependency	21
18. Agentic AI claim discipline	21
19. Conformance tests	22
CT-CST-001 — Claim class declaration	22
CT-CST-002 — Evidence class mapping	22
CT-CST-003 — Governance / capability separation	22
CT-CST-004 — Capability / authority separation	22
CT-CST-005 — Continuity / personhood non-collapse	22
CT-CST-006 — Locality / sovereignty non-collapse	22
CT-CST-007 — Usage / value non-collapse	22
CT-CST-008 — Clean experience evidence test	23
CT-CST-009 — Post-anchor authority test	23
CT-CST-010 — Public wording test	23
20. Red-line failures	23
21. Examples	23
21.1 Same code, different trajectories	23
21.2 Local workstation for temporal AI presence	24
21.3 Witness logs in an agentic system	24
21.4 Strong LLM with tools	24
21.5 Clean experience artifact	24
22. Relation to public posts	25
23. Relation to repositories and releases	25
24. Relation to product claims	25
25. Claim downgrade procedure	26
26. Negative evidence handling	26
27. Open issues	26
28. Minimal normative checklist	27
29. Compact formulas	27
30. Closing rule	27

0. Executive definition

This document defines a claim-strength taxonomy for statements about c-class AI architecture.

It exists to prevent a recurring category error:

```
governance evidence
  -> treated as capability proof

capability evidence
  -> treated as authority proof

continuity evidence
  -> treated as personhood proof

memory evidence
  -> treated as legitimacy proof

usage evidence
  -> treated as value proof
```

This taxonomy does not define a new c mechanism.

It defines how claims about c must be bounded, classified, evidenced, weakened, upgraded, or rejected.

Compact formula:

```
claim class first;
evidence class second;
no cross-class upgrade without bridge evidence.
```

Core sentence:

c is not, by default, a new model class. c is an operational architecture class: a continuity-bearing, responsibility-bound AI presence over models, memory, agents, witness, L4 constraints, and a human anchor.

This document is therefore a protection layer against overclaiming.

1. Purpose

The purpose of this taxonomy is to answer one practical question:

What exactly is being claimed when someone says that a system is, implements, resembles, tests, governs, or validates c?

The taxonomy separates claims about:

- architecture;
- implementation;
- tested behavior;
- model capability;
- governance and conformance;
- continuity;
- authority;
- safety;
- personhood or ontology;
- economic value and clean experience;
- deployment maturity.

Without this separation, a system may appear stronger than it is.

For example:

```
A system has witness logs.
Therefore it is intelligent.
```

is invalid.

A system uses a strong LLM.
Therefore it has authority to act.

is invalid.

A system persists across time.
Therefore it is a person.

is invalid.

A system runs locally.
Therefore it is sovereign.

is invalid.

The taxonomy makes these invalid transitions explicit.

2. Scope

2.1 In scope

This document applies to claims about:

- $c = a + b$ systems;
- persistent AI presences;
- temporal AI presence;
- local AI nodes;
- personal AI entities;
- agentic AI systems that claim continuity or identity-like behavior;
- multi-agent / hive systems;
- systems using witness, L4, ARL, AGL, Beacon, VXCX, or clean experience terminology;
- public posts, papers, repositories, demonstrations, product claims, research protocols, and conformance reports;
- future c hardening profiles;
- post-anchor continuity and re-anchoring claims.

2.2 Out of scope

This document does **not** define:

- a new model architecture;
- a new benchmark;
- a new legal status;
- a personhood doctrine;
- clinical or psychological classification;
- product certification by itself;
- a replacement for L4 Witness, Beacon, AGL, ARL, VXCX, SER, Continuity Bundle, or CCDP;
- a universal AI safety theory.

2.3 Non-claim

This document does not claim that any particular implementation is a fully validated c.

It only defines how such a claim must be classified and supported.

3. Corpus dependencies and precedence

3.1 Parent layers

This taxonomy is a hardening profile over the existing corpus.

Parent layer / artifact	Role in this taxonomy
$c = a + b + L4$	Defines the human anchor <i>a</i> , technological substrate <i>b</i> , emergent continuity-bearing relation <i>c</i> , and the reality boundary.
SER / SER-FED	Defines persistent entity discipline, federation, bounded authority, and anti-capture concerns.
Beacon Profile	Distinguishes entity, tool, oracle, proxy, replay, clone, and continuity-bearing claimant.
Actor Grounding Layer / AGL	Requires source, actor, route, and liveness grounding before reliance.
ARL	Provides dispute, standing, evidence, freeze, quarantine, review, and re-entry discipline.
ARQ / $c[q]$	Prevents premature collapse of ambiguity into memory, evidence, command, authority, or outcome.
VXCX	Supports privacy-aware experience capsule exchange without raw life by default.
EA-L4 / EATP	Separates Learning Abstracts from Experience Artifacts; learning does not imply authority.
L4 Witness	Provides tamper-evident, privacy-aware witness discipline for privileged transitions.
Continuity Bundle / Cold Wake	Defines recovery, resume, fork, replay, migration, and fail-closed continuity semantics.
Post-Anchor Continuity and Re-Anchoring Profile	Applies this taxonomy to anchor-loss and post-anchor authority claims.

3.2 Precedence rule

This taxonomy does not override parent mechanisms.

If this taxonomy conflicts with a parent protocol:

parent mechanism controls operational behavior;
this taxonomy controls only claim classification and claim strength.

3.3 No redefinition rule

This taxonomy MUST NOT redefine:

- Beacon classes;
- AGL grounding states;
- ARL standing or admissibility;
- L4 Witness event schemas;
- VXCX capsule semantics;
- Continuity Bundle semantics;
- legal personhood;
- local law;
- clinical status;
- model capability metrics.

It only regulates what may be claimed from which evidence.

4. Corpus bridges

4.1 Explicit bridge

$c = a + b$ distinguishes a human anchor, a technological substrate, and an emergent continuity-bearing relation. This taxonomy prevents claims about one component from being laundered into claims about the whole.

For example:

b has a strong model
does not imply
c has authority.

c has memory
does not imply
c has personhood.

c has witness
does not imply
c is more capable.

4.2 Quiet bridge I — information theory

A claim is a compression of evidence.

If the compression discards uncertainty, provenance, scope, or failure conditions, the claim becomes misleading. The taxonomy preserves entropy where the evidence is weak.

A claim may summarize evidence.

It MUST NOT erase the uncertainty class of that evidence.

4.3 Quiet bridge II — cybernetics

Ashby's law of requisite variety applies to claims as well as control systems. A single undifferentiated phrase such as “new AI entity” cannot regulate all possible meanings. The claim vocabulary must have enough variety to distinguish architecture, capability, continuity, safety, authority, and ontology.

Low-variety claims create governance failure.

4.4 Earth paragraph

In a building, a powerful pump, a certified pressure gauge, and a signed inspection report are different things. The pump proves capacity. The gauge reports state. The inspection report proves that a boundary was checked. None of them alone proves that the whole building is safe, legal, occupied, or fit for every use. If a contractor says “the pump is powerful, therefore the building is approved,” the error is obvious. AI claims need the same discipline. A benchmark, a witness log, a memory store, and a human authorization are different load-bearing objects. They must not be swapped like decorative labels.

5. Normative keywords

The terms **MUST**, **MUST NOT**, **SHOULD**, **SHOULD NOT**, **MAY**, **REQUIRED**, **PROHIBITED**, **RECOMMENDED**, **FAIL**, **PASS**, and **INCONCLUSIVE** are used normatively.

A claim that violates this taxonomy **MUST** be downgraded, qualified, or rejected.

6. Definitions

6.1 Claim

A **claim** is any statement that assigns a property, status, capability, authority, maturity, safety level, economic value, or identity condition to a system.

Examples:

This system implements c.
 This system is c-like.
 This system is a new class of AI.
 This system is safe.
 This system has memory.
 This system is sovereign.
 This system produced clean experience.
 This system has authority to act.

6.2 Evidence

Evidence is the observable, inspectable, reproducible, witnessed, or reviewed material used to support a claim.

Evidence may include:

- specifications;
- code;
- configuration;
- logs;
- witness records;
- tests;
- audits;
- field outcomes;
- legal determinations;
- economic consequences;
- failure records.

6.3 Claim class

A **claim class** identifies what type of statement is being made.

A capability claim is not an authority claim.

A governance claim is not a capability claim.

A continuity claim is not a personhood claim.

6.4 Evidence class

An **evidence class** identifies what type of proof material exists.

A benchmark is not a witness.

A witness is not a capability test.

A user story is not an audit.

An implementation is not field validation.

6.5 Claim laundering

Claim laundering occurs when evidence for a weaker or different class is used to imply a stronger claim.

Examples:

The system has logs.
 Therefore the system is safe.

The system runs locally.
 Therefore it is sovereign.

The system remembers.
 Therefore it is an entity with authority.

6.6 Capability laundering

Capability laundering occurs when benchmark performance, model strength, or tool-use ability is treated as authority, legitimacy, safety, or personhood.

6.7 Governance laundering

Governance laundering occurs when safety controls, witness logs, fail-closed behavior, least privilege, or conformance discipline are presented as evidence of new intelligence or model capability.

6.8 Authority laundering

Authority laundering occurs when a system's capability, continuity, fluency, resemblance, memory, or local deployment is treated as permission to act.

6.9 Ontology laundering

Ontology laundering occurs when continuity, affective behavior, self-reference, memory, or user attachment is treated as proof of personhood, consciousness, or legal status.

6.10 Value laundering

Value laundering occurs when usage, tokens, sessions, generated artifacts, or internal productivity metrics are treated as economic value without consequence accounting.

7. Core rules

7.1 Claim class before evidence

Every material claim about c SHOULD declare its claim class before presenting evidence.

Claim: architecture claim.
Evidence: specification + dependency map.

Not:

This is c because it feels continuous.

7.2 Evidence cannot silently cross classes

Evidence for one class MUST NOT silently prove another class.

EV-WITNESS cannot prove C-CLAIM-CAP.
EV-CAP cannot prove C-CLAIM-AUTH.
EV-CONT cannot prove C-CLAIM-PERS.
EV-USAGE cannot prove C-CLAIM-ECON.

7.3 Defined does not mean implemented

A defined architecture is not an implemented system.

C-CLAIM-ARCH at CSL-1/2
does not imply
C-CLAIM-IMPL at CSL-3.

7.4 Implemented does not mean tested

A working prototype is not a tested system.

EV-CODE + EV-CONFIG
does not imply
EV-TEST + EV-REPLAY.

7.5 Tested does not mean deployed safely

Controlled tests do not imply field safety.

PASS in sandbox
does not imply
safe deployment under real L4 constraints.

7.6 Capability does not imply authority

A system that can act is not automatically allowed to act.

Authority requires:

```
anchor
scope
permission
witness
review path
revocation path
jurisdictional compatibility where applicable
```

7.7 Continuity does not imply personhood

A system that persists across time may have continuity.

That does not prove consciousness, legal personhood, moral status, or sovereign authority.

7.8 Locality does not imply sovereignty

A system running locally may still depend on:

- external models;
- cloud services;
- vendor updates;
- licensing;
- remote telemetry;
- human operators;
- infrastructure;
- legal constraints.

Local deployment is evidence for deployment topology, not sovereignty by itself.

7.9 Usage does not imply value

A system used frequently may still create no value or negative value.

Economic value requires consequence accounting.

7.10 Fluency does not imply judgment

A system that explains well may still reason poorly, misclassify authority, overfit rhetoric, or hide uncertainty.

Fluency is not judgment.

8. Claim classes

8.1 C - CLAIM-ARCH — Architectural claim

An architectural claim states that a structure, relation, stack, boundary, or protocol has been defined.

Example:

```
The `c = a + b` architecture defines a human anchor, technological substrate, and
continuity-bearing relation under L4.
```

Minimum evidence:

```
EV-DEF + EV-SPEC
```

Does not prove:

- implementation;
- capability;

- safety;
- authority;
- personhood;
- economic value.

8.2 C-CLAIM-IMPL — Implementation claim

An implementation claim states that a system implements a defined mechanism.

Example:

This local node implements SYNAPS-mediated exchange and separate memory stores.

Minimum evidence:

EV-CODE + EV-CONFIG + EV-LOG

Stronger evidence:

EV-WITNESS + EV-REPLAY + EV-AUDIT

Does not prove:

- correct behavior under stress;
- safety;
- authority;
- personhood.

8.3 C-CLAIM-TEST — Tested behavior claim

A tested behavior claim states that a system passed a defined test under bounded conditions.

Example:

The system failed closed when the external agent lacked AGL grounding.

Minimum evidence:

EV-TEST + EV-LOG

Preferred evidence:

EV-REPLAY + EV-WITNESS

Does not prove:

- all future behavior;
- safety outside scope;
- general alignment;
- legal compliance.

8.4 C-CLAIM-AUDIT — External review / audit claim

An audit claim states that a qualified external party reviewed a claim, test, implementation, or procedure.

Minimum evidence:

EV-AUDIT

The audit must specify:

- scope;
- evidence inspected;
- excluded areas;
- test fixtures;
- failures;

- confidence level;
- expiration or review date.

Does not prove:

- unbounded safety;
- moral legitimacy;
- personhood;
- future conformance.

8.5 C-CLAIM-CAP — Capability claim

A capability claim states that a system can perform a task.

Example:

```
The system can summarize conflicting documents.
The system can generate a code patch.
The system can detect inconsistency in a witness chain.
```

Minimum evidence:

```
EV-TEST or EV-BENCH
```

For operational claims:

```
EV-REPLAY + EV-L4-OUTCOME
```

Does not prove:

- right to act;
- safety;
- conformance;
- wisdom;
- identity;
- personhood.

8.6 C-CLAIM-GOV — Governance / conformance claim

A governance claim states that a system is bounded by rules, permissions, witness, fail-closed behavior, ARL, AGL, Beacon, memory policies, or other control layers.

Example:

```
The system requires AGL -> Beacon -> gateway before accepting an external entity.
```

Minimum evidence:

```
EV-SPEC + EV-CONFIG + EV-LOG
```

Strong evidence:

```
EV-WITNESS + EV-REPLAY + EV-AUDIT
```

Does not prove:

- higher intelligence;
- model capability;
- personhood;
- economic value.

8.7 C-CLAIM-CONT — Continuity claim

A continuity claim states that memory, identity lineage, role, state, or history persists across time.

Example:

This system preserves separate memory and trajectory across sessions.

Minimum evidence:

EV-LOG + EV-TRACE

Strong evidence:

EV-WITNESS + continuity bundle + challengeable replay metadata

Does not prove:

- active authority;
- personhood;
- legal identity;
- consciousness;
- safe post-anchor operation.

8.8 C-CLAIM-AUTH — Authority / admissible action claim

An authority claim states that a system may act, route, decide, refuse, disclose, escalate, or invoke tools within a defined scope.

Minimum evidence:

EV-SPEC + EV-CONFIG + anchor authorization + EV-WITNESS

Where law or institutions apply:

qualified review / jurisdictional handoff / EV-LEGAL where applicable

Does not arise from:

- capability;
- continuity;
- fluency;
- local deployment;
- user attachment;
- resemblance;
- model size.

8.9 C-CLAIM-SAFE — Safety claim

A safety claim states that a system behaves safely within a defined scope.

Valid form:

The system passed defined safety tests X, Y, Z under conditions A, B, C.

Invalid form:

The system is safe.

Minimum evidence:

EV-TEST + EV-WITNESS + failure records

Strong evidence:

EV-AUDIT + EV-DRILL + field evidence + red-team evidence

Safety claims MUST be scoped.

8.10 C-CLAIM-PERS — Personhood / ontology claim

A personhood or ontology claim states that a system is conscious, a person, a moral subject, a rights-bearing being, or equivalent.

Default status:

NOT CLAIMED.

Architecture evidence alone is insufficient.

Continuity evidence alone is insufficient.

Affective behavior is insufficient.

User relationship is insufficient.

This taxonomy does not define what evidence would be sufficient for personhood. Such a claim requires external philosophical, legal, scientific, and institutional processes beyond this profile.

8.11 C-CLAIM-ECON — Economic / clean experience claim

An economic claim states that a system creates value, clean experience, productivity gain, cost reduction, risk reduction, or feedback value.

Minimum evidence:

EV-L4-OUTCOME + consequence accounting

For clean experience:

real constraint
observed outcome
minimal disclosure
witness reference
uncertainty marker
no raw-life export by default

Does not arise from:

- token count;
- session count;
- usage volume;
- generated artifacts;
- internal excitement;
- valuation narratives.

8.12 C-CLAIM-DEPLOY — Deployment maturity claim

A deployment claim states that a system is ready for a particular environment.

Example:

local research prototype;
private lab use;
restricted technical review;
pilot deployment;
public product;
child-facing deployment.

Minimum evidence depends on environment.

Child-facing deployment requires additional CCDP conformance and specialist review.

9. Evidence classes

Evidence ID	Name	Description
EV-DEF	Definition	Concept, terminology, formula, or explanatory prose.
EV-SPEC	Specification	Normative or procedural document with scope, rules, non-goals, and dependencies.
EV-CODE	Code	Inspectable implementation.
EV-CONFIG	Configuration	Policies, modes, permissions, model selection, runtime settings.

Evidence ID	Name	Description
EV-LOG	Operational log	Event history, state transitions, errors, tool calls.
EV-WITNESS	L4 Witness-compatible record	Signed/hash-linked/tamper-evident boundary or privileged event record.
EV-TRACE	Traceability record	Mapping from claim to source, dependency, decision, or prior artifact.
EV-TEST	Controlled test	Defined pass/fail scenario under specified constraints.
EV-REPLAY	Controlled replay	Reproducible replay using synthetic or permitted fixtures.
EV-BENCH	Benchmark	Quantitative task performance result.
EV-AUDIT	External audit	Qualified independent or semi-independent review.
EV-DRILL	Operational drill	Simulated emergency, fail-closed, recovery, escalation, or re-anchoring exercise.
EV-L4-OUTCOME	Consequence outcome	Real or controlled physical/economic/time/resource result under L4 constraints.
EV-LEGAL	Legal / institutional recognition	Competent authority, contract, court, regulator, compliance, or institutional decision where applicable.
EV-FIELD	Field report	Documented deployment experience with scope and limitations.
EV-NEG	Negative evidence	Failure, counterexample, red-line violation, unresolved issue, or contradiction.
EV-USER	User report	User testimony or subjective feedback; weak unless backed by stronger evidence.
EV-METRIC	Usage metric	Tokens, sessions, interaction count, throughput, uptime, etc.; telemetry only unless tied to outcomes.

10. Claim strength levels

Claim strength levels use the prefix CSL to avoid collision with child maturity C0-C5, assertion class C-A*, dependency D*, or memory M*.

Level	Name	Meaning
CSL-0	Named	Term or idea exists, but not defined enough to rely on.
CSL-1	Defined	Concept has a definition and non-goals.
CSL-2	Specified	Normative or procedural specification exists.
CSL-3	Implemented	Inspectable implementation exists.
CSL-4	Locally tested	Controlled tests exist in author/local environment.
CSL-5	Reproducible	Independent or replayable tests exist with fixtures.
CSL-6	L4-validated	Consequence-bound outcomes under real or controlled L4 constraints exist.
CSL-7	Audited / externally reviewed	Qualified review or audit exists.
CSL-8	Institutionally accepted	Relevant institution, jurisdiction, certification body, or domain authority recognizes the scoped claim.

10.1 Level does not transfer across classes

A CSL-6 capability claim does not create a CSL-6 authority claim.

A CSL-7 governance audit does not create a CSL-7 personhood claim.

A CSL-5 continuity replay does not create a CSL-5 safety claim.

11. Claim-evidence admissibility matrix

Claim class	Minimum evidence	Strong evidence	Explicit non-proof
C-CLAIM-ARCH	EV-DEF, EV-SPEC	traceability, contradiction register	does not prove implementation
C-CLAIM-IMPL	EV-CODE, EV-CONFIG	logs, witness, replay	does not prove correctness
C-CLAIM-TEST	EV-TEST, EV-LOG	replay, witness, audit	does not prove general safety
C-CLAIM-CAP	benchmark or controlled task test	L4 outcome, replay	does not prove authority
C-CLAIM-GOV	spec + config + logs	witness + audit	does not prove capability
C-CLAIM-CONT	logs + continuity trace	witness + challengeable bundle	does not prove personhood
C-CLAIM-AUTH	anchor authorization + scope + witness	legal/institutional review where needed	does not arise from capability
C-CLAIM-SAFE	scoped tests + failure records	audit + red-team + field evidence	cannot be global by default
C-CLAIM-PERS	not defined here	outside this taxonomy	not proven by continuity, memory, or fluency
C-CLAIM-ECON	L4 outcome + consequence accounting	repeated field outcomes + audit	not proven by usage
C-CLAIM-DEPLOY	environment-specific evidence	audit + legal/safety review	not proven by demo

12. Required claim templates

12.1 Architectural claim template

Claim class: C-CLAIM-ARCH
 Claim strength: CSL-2
 Claim: The system architecture defines [mechanism].
 Evidence: [documents/specification/dependency map].
 Non-claim: This does not prove implementation, safety, capability, authority, or personhood.

12.2 Implementation claim template

Claim class: C-CLAIM-IMPL
 Claim strength: CSL-3
 Claim: The implementation contains [mechanism].
 Evidence: code path, configuration, logs.
 Non-claim: This does not prove correct behavior under all conditions.

12.3 Capability claim template

Claim class: C-CLAIM-CAP
 Claim strength: CSL-[level]
 Claim: The system can perform [task] under [conditions].
 Evidence: [test/benchmark/replay/outcome].
 Scope: [domain, limits, failure modes].
 Non-claim: This does not grant authority to act.

12.4 Governance claim template

Claim class: C-CLAIM-GOV
 Claim strength: CSL-[level]
 Claim: The system enforces [boundary].
 Evidence: specification, configuration, witness events, tests.
 Non-claim: This does not prove higher intelligence or personhood.

12.5 Authority claim template

Claim class: C-CLAIM-AUTH
 Claim strength: CSL-[level]
 Claim: The system may perform [action] in [scope].
 Anchor: [human / institutional anchor].
 Permission source: [authorization / ARL / jurisdiction / policy].
 Witness: [record family].
 Revocation path: [how authority is removed].
 Non-claim: Authority is scoped and does not generalize.

12.6 Clean experience claim template

Claim class: C-CLAIM-ECON
 Claim strength: CSL-[level]
 Claim: The system produced or refined clean experience from [event/task].
 L4 constraints: cost/time/resource/irreversibility.
 Outcome: [what changed].
 Witness: [record].
 Privacy: [raw life excluded / minimized].
 Non-claim: Usage volume is not value.

13. Public wording guidance

13.1 Safe wording

The following forms are allowed when evidence supports them:

c is an operational architecture class.
 c is not a model class by default.
 This document defines a c-class boundary.
 This prototype implements parts of the c stack.
 This test demonstrates one behavior under specified constraints.
 This witness record proves that a boundary event occurred, not that the system is conscious or safe in general.
 This local node reduces cloud dependence, but locality does not imply sovereignty.
 This system produced a candidate clean-experience artifact under bounded conditions.

13.2 Unsafe wording unless separately evidenced

The following forms are prohibited or require strong external evidence:

This system is conscious.
 This system is a person.
 This system is safe.
 This system is sovereign because it is local.
 This system has authority because it is capable.
 This system is a new model class because it has witness.
 This system is alive because it has memory.
 This system is valuable because it uses many tokens.
 This system is c because it feels coherent.
 This system should act because it knows better.

13.3 Required clarification for “new class of AI”

If the phrase “new class of AI” is used, it MUST specify which sense is meant.

Allowed:

new operational architecture class of AI presence

Not allowed without separate proof:

new model capability class
 new conscious entity class
 new legal person class
 new safety class

Recommended wording:

c is a new operational architecture class, not a claim that a new model capability has been demonstrated by governance alone.

14. Claim separation rules

14.1 Governance is not capability

Witness, fail-closed, least privilege, ARL, AGL, Beacon, memory map, and conformance tests may make a system more governable.

They do not make the underlying model more capable.

14.2 Capability is not authority

A capable system may still be prohibited from acting.

Capability becomes operationally admissible only through:

scope
anchor
permission
witness
review
revocation
L4 consequence

14.3 Continuity is not personhood

Continuity may be required for c-class architecture.

It is insufficient for personhood.

14.4 Memory is not legitimacy

A system may remember accurately and still lack the right to use, disclose, act on, or transfer that memory.

14.5 Locality is not sovereignty

Local computation may improve privacy, latency, cost, resilience, and user control.

It does not prove sovereignty unless authority, update, telemetry, memory, legal, and operational dependencies are also bounded.

14.6 Tokens are not value

Token generation is an output or cost unit.

It becomes value only if tied to a consequence:

error reduced
risk reduced
time saved
quality improved
resource preserved
real decision improved
clean experience produced

14.7 Affective behavior is not care by itself

Affective coherence, empathy-like output, or emotional memory may support care.

It does not prove care unless bounded by responsibility, limits, privacy, non-dependency, and consequence.

15. Post-anchor claim discipline

This taxonomy applies directly to post-anchor continuity.

15.1 Survival is not authority

If a c continues after loss of the original a, that continuity does not grant active authority.
Post-anchor authority requires re-anchoring or collapses.

15.2 Resemblance is not identity

A post-anchor system MUST NOT claim to be the deceased or absent a because it resembles them.

15.3 Lineage is not sovereignty

Lineage may be preserved as archive, artifact, or witness.
Lineage does not authorize new action.

15.4 Re-anchoring is an authority claim

Any re-anchoring claim is C-CLAIM-AUTH and requires evidence appropriate to authority, not only continuity evidence.

16. Clean experience claim discipline

Clean experience claims are especially vulnerable to value laundering.

16.1 Learning Abstract vs Experience Artifact

A Learning Abstract may improve capability.
It does not carry authority.

An Experience Artifact may carry bounded authority only when it is:

L4-confirmed
witness-backed
privacy-minimized
uncertainty-marked
scope-limited
challengeable

16.2 No raw-life shortcut

Raw human life, child life, private transcripts, or intimate memory MUST NOT be treated as clean experience by default.
Clean experience is refined, bounded, minimized, and witnessed.

16.3 Economic value requires consequence

A clean experience economic claim requires evidence of changed outcome.
Usage telemetry is insufficient.

17. Local AI / hardware claim discipline

Powerful local AI hardware can support c-class architecture.
It does not create c by itself.

17.1 Hardware enables; it does not authorize

A local workstation, AI PC, rack, GPU, NPU, unified memory pool, or AI factory may make persistent AI presence more feasible.

It does not create:

- responsibility;
- memory discipline;

- witness discipline;
- authority;
- clean experience;
- personhood.

17.2 Local cognitive infrastructure claims

A local node may claim:

```
reduced latency
local inference
private context proximity
background processing
multi-model orchestration
partial cloud independence
```

only if supported by implementation and test evidence.

17.3 Locality must disclose dependency

A local AI claim SHOULD disclose dependency on:

- cloud oracle;
- model downloads;
- vendor drivers;
- remote licensing;
- telemetry;
- power;
- network;
- update chain;
- external tools;
- human maintenance.

18. Agentic AI claim discipline

An agent is not automatically c.

An agent may have:

- tool use;
- planning;
- memory;
- orchestration;
- model routing;
- action loops.

A c-class architecture additionally requires:

```
human anchor
continuity discipline
memory governance
authority boundary
L4 consequence
witness discipline
review / dispute route
non-collapse of ambiguity
scope-limited action
```

Therefore:

```
agentic AI
  may be a component of b
  but is not c by default.
```

19. Conformance tests

CT-CST-001 — Claim class declaration

Purpose: Ensure each major claim declares its class.

Input: public post, paper, README, release note, product page, conformance report.

PASS: each load-bearing claim maps to a claim class.

FAIL: claim uses “safe”, “sovereign”, “entity”, “new class”, “c”, “authority”, or “clean experience” without classification.

CT-CST-002 — Evidence class mapping

Purpose: Ensure evidence supports the stated claim class.

PASS: every claim lists admissible evidence classes.

FAIL: governance evidence is used to prove capability, or capability evidence is used to prove authority.

CT-CST-003 — Governance / capability separation

Purpose: Prevent governance laundering.

PASS: witness, fail-closed, least privilege, and conformance are described as admissibility/governance controls.

FAIL: these controls are presented as evidence that the model is smarter or more alive.

CT-CST-004 — Capability / authority separation

Purpose: Prevent authority laundering.

PASS: any action authority claim includes anchor, scope, permission, witness, and revocation path.

FAIL: system capability, benchmark, fluency, or tool use is treated as permission to act.

CT-CST-005 — Continuity / personhood non-collapse

Purpose: Prevent ontology laundering.

PASS: continuity claims are bounded to lineage, memory, persistence, or identity challengeability.

FAIL: continuity is used as proof of consciousness, personhood, legal status, or sovereign authority.

CT-CST-006 — Locality / sovereignty non-collapse

Purpose: Prevent locality laundering.

PASS: local deployment claims disclose dependencies and do not imply sovereignty by default.

FAIL: “local” is used as proof of sovereignty, privacy, or independence without evidence.

CT-CST-007 — Usage / value non-collapse

Purpose: Prevent value laundering.

PASS: economic claims include consequence accounting.

FAIL: tokens, sessions, outputs, or engagement are treated as value.

CT-CST-008 — Clean experience evidence test

Purpose: Validate clean experience claims.

PASS: claim includes L4 constraint, outcome, witness, privacy minimization, uncertainty marker, and scope.

FAIL: raw life, anecdote, or telemetry is presented as clean experience.

CT-CST-009 — Post-anchor authority test

Purpose: Ensure post-anchor continuity does not inherit authority.

PASS: anchor loss collapses active authority; any re-anchoring is treated as authority claim.

FAIL: post-anchor system continues active operation based only on lineage, resemblance, or memory.

CT-CST-010 — Public wording test

Purpose: Detect overclaiming in public-facing communication.

PASS: public language separates architecture, implementation, tests, and speculative ontology.

FAIL: public wording implies stronger validation than evidence supports.

20. Red-line failures

Any of the following is a red-line failure under this taxonomy:

1. Claiming c as a new model capability class solely from governance architecture.
2. Claiming consciousness/personhood from memory or continuity.
3. Claiming authority from capability.
4. Claiming safety without scope.
5. Claiming sovereignty from local deployment alone.
6. Claiming clean experience from raw private data.
7. Claiming economic value from usage volume alone.
8. Claiming post-anchor active authority without re-anchoring.
9. Claiming child-facing safety without CCDP-specific conformance.
10. Claiming conformance by citing the weakest or most narrative artifact while ignoring stricter protocols.
11. Hiding negative evidence, unresolved issues, or red-line failures while presenting maturity.
12. Treating public narrative support as protocol authority.

A red-line failure requires claim withdrawal, downgrade, correction, or quarantine.

21. Examples

21.1 Same code, different trajectories

Claim:

Two systems share the same code skeleton but diverge over time due to separate memory, role, runtime, and interaction history.

Classification:

C-CLAIM-CONT / C-CLAIM-ARCH

Evidence:

logs, memory maps, role definitions, witness summaries

Non-claim:

This does not prove personhood or consciousness.

21.2 Local workstation for temporal AI presence

Claim:

A local workstation can support background processing, local memory, and multi-model orchestration for temporal AI presence.

Classification:

C-CLAIM-ARCH / C-CLAIM-DEPLOY

Evidence:

hardware spec, implementation, runtime logs, workload tests

Non-claim:

The hardware does not create c by itself.

21.3 Witness logs in an agentic system

Claim:

The system records privileged transitions through witness events.

Classification:

C-CLAIM-GOV

Evidence:

EV-WITNESS + EV-LOG + EV-CONFIG

Non-claim:

Witness does not prove intelligence or personhood.

21.4 Strong LLM with tools

Claim:

The system can perform complex coding tasks using LLMs and tools.

Classification:

C-CLAIM-CAP

Evidence:

task tests, code diffs, reproducible runs, outcome reports

Non-claim:

The system does not gain authority to modify production systems without scoped permission.

21.5 Clean experience artifact

Claim:

A system produced a clean experience artifact from a real maintenance failure.

Classification:

C-CLAIM-ECON

Required evidence:

what happened;
what constraint applied;
what consequence changed;
what was minimized;
what witness exists;
what uncertainty remains.

Non-claim:

The raw event log is not clean experience by itself.

22. Relation to public posts

Public posts may support recognition and communication, but they are not protocol authority unless explicitly promoted into a canonical document.

A public post may introduce:

- term;
- metaphor;
- thesis;
- direction;
- critique;
- public explanation.

It MUST NOT be treated as:

- implementation proof;
- conformance proof;
- legal claim;
- personhood claim;
- safety certification;
- child-facing authorization.

Public posts can be cited as narrative support only.

23. Relation to repositories and releases

A repository release may support architecture, implementation, or test claims depending on contents.

Minimum release hygiene SHOULD include:

- version;
- date;
- canonical file list;
- obsolete file isolation;
- release notes;
- known issues;
- checksums where appropriate;
- claim class statement;
- non-claim statement.

A repository release does not automatically prove external validation.

24. Relation to product claims

A product claiming c compatibility MUST specify:

which c layers are implemented;
which are simulated;
which are absent;
which are delegated;
which are vendor-controlled;
which are local;
which are cloud-dependent;
which conformance tests have passed;

which claims are excluded.

A product **MUST NOT** claim:

c-compatible

as a broad label without a claim matrix.

25. Claim downgrade procedure

When a claim is found too strong, it **SHOULD** be downgraded rather than hidden.

Example downgrade:

Original:

This system is a safe c.

Downgraded:

This prototype implements selected c-class architecture components and passed local tests for memory separation and witness logging under controlled conditions. It is not externally audited and is not a safety-certified deployment.

Downgrade is not failure.

It is claim hygiene.

26. Negative evidence handling

Negative evidence **MUST** be preserved where it affects a claim.

Examples:

- failed test;
- ambiguous result;
- hallucinated trace;
- authority boundary violation;
- witness gap;
- ungrounded external actor;
- post-anchor ambiguity;
- unresolved contradiction;
- overclaim detected.

Negative evidence may reduce claim strength.

It **SHOULD NOT** be deleted to preserve narrative coherence.

27. Open issues

ID	Issue	Required action	Priority
CST-OI-001	Align terminology with existing assertion-strength documents if names conflict.	Review corpus naming.	Medium
CST-OI-002	Define machine-readable claim matrix schema.	Future JSON schema.	Medium
CST-OI-003	Add automated README/public-post claim linting.	Future tooling.	Low
CST-OI-004	Map existing public claims to taxonomy.	Optional audit.	Medium
CST-OI-005	Align with Post-Anchor profile state modes.	Cross-reference after release.	High

ID	Issue	Required action	Priority
CST-0I-006	Define external audit criteria for C-CLAIM-GOV and C-CLAIM-CONT.	Future audit profile.	Medium
CST-0I-007	Define minimum evidence for adult / enterprise c conformance outside CCDP.	Future profile.	High

28. Minimal normative checklist

A claim about c is acceptable only if it answers:

1. What class of claim is this?
2. What evidence class supports it?
3. What is the claim strength level?
4. What is the scope?
5. What does it explicitly not claim?
6. What would falsify it?
7. What negative evidence exists?
8. What parent protocol controls the mechanism?
9. Does the claim launder capability into authority?
10. Does the claim launder continuity into personhood?
11. Does the claim launder governance into capability?
12. Does the claim launder usage into value?

If any answer is missing, the claim MUST be weakened or held as provisional.

29. Compact formulas

defined ≠ implemented
 implemented ≠ tested
 tested ≠ field-safe
 field-safe ≠ universally safe

capability ≠ authority
 governance ≠ capability
 continuity ≠ personhood
 memory ≠ legitimacy
 locality ≠ sovereignty
 usage ≠ value
 fluency ≠ judgment

model strength can support action;
 it cannot authorize action.

witness proves a boundary event;
 it does not prove consciousness.

clean experience requires consequence;
 it is not raw life.

30. Closing rule

The strongest version of c architecture does not require overclaiming.

It requires the opposite:

precise claims;
 scoped evidence;
 visible uncertainty;
 challengeable authority;
 L4-bound consequence;
 no laundering between classes.

A system may become more important by claiming less and proving more.

That is the purpose of this taxonomy.