AMPEX
DATA SYSTEMS CORPORATION

The Evidence Backbone for Cognitive Electronic Warfare

An Architect’s Guide to a Deterministic Data Plane for Testable, Reproducible, and Certifiable CogEW

By James Spriet
EW/ISR Subject Matter Expert,
Strategy and Communications Manager
7 Jan 2025

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

Contents

Executive Overview 4
Core invariants 4
Release rule 5
Objective and scope 6
Objective 6
Scope 7
Definitions 7
Assumptions 8
Non-goals 8
1. The Verification Gap 8
1.1 Silent loss 8
1.2 Intent vs execution 9
1.3 Post hoc repair and invented history 9
1.4 Operational impact 9
2. Cognition is a causal system 10
2.1 The four cognitive artifacts 10
2.2 Bounded uncertainty as a hard requirement 11
3. Evidence is a first-class data product 11
4. Evidence invariants for CogEW systems 12
4.1 Time integrity as the root of causality 13
4.2 Executed-state closure 13
4.3 Effect bounding and measurement pedigree 14
4.4 Disqualifying condition 14
5. Throughput-aligned capture architecture 15
5.1 The Truth Spine 15
5.2 High-rate capture windows 15
5.3 Trigger doctrine 16
5.4 Deterministic backpressure and ordering 17
5.5 Evidence objects, not dumps 17
6. Deterministic contention behavior under load 17
6.1 Why real platforms always contend 18

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

6.2 Arbitration must be deterministic ... 18
6.3 Non-preemptible vs preemptible evidence channels ... 19
6.4 Loss as a state transition ... 19
6.5 Arbitration decisions as evidence ... 20
7. Evidence eligibility and learning gating ... 20
7.1 Eligibility is a binary property ... 21
7.2 Automated exclusion ... 21
7.3 Separation of collection authority and learning authority ... 21
7.4 Discarding evidence as discipline ... 22
8. Regression-grade integrity and provenance ... 22
8.1 The sortie manifest as a system output ... 22
8.2 Versioned data products at ingest ... 23
8.3 Chunked evidence objects with integrity ... 23
8.4 Calibration lineage ... 23
8.5 Machine-readable sortie package index ... 24
9. Configuration control and release authority ... 24
9.1 Golden rule ... 25
9.2 The quadrant of attribution ... 25
9.3 Preventing autonomous drift ... 26
9.4 Evidence as release authority ... 26
10. Interface closure ... 26
10.1 Open parsing as a certification requirement ... 27
10.2 Minimum evidence streams ... 27
10.3 Failure attribution rule ... 28
11. Integration boundaries and responsibility split ... 28
11.1 Nervous system contract ... 28
11.2 Responsibility split ... 29
11.3 Integration support versus application ownership ... 29
11.4 Why tight coupling is disqualifying ... 30
11.5 Hard rule ... 30
12. Disqualifiers and failure modes ... 30
12.1 Disqualifiers ... 31

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

12.2 Failure modes when disqualifiers are ignored 32
13. What Cognitive EW means when done correctly 33
13.1 Grounded definition 33
13.2 The cognitive loop as four artifacts 34
13.3 Necessary and sufficient conditions 34
13.4 Operational meaning 35
13.5 Center of gravity 35
Appendix A 35

Executive Overview

Cognitive EW (CogEW) does not fail because teams choose the wrong algorithm. It fails because causal truth collapses under real platform stress. Once truth is gone, claims about learning cannot be tested, repeated, or certified.

This guide defines what must be true in the data plane and the evidence backbone for CogEW to be real. Not as branding, but as something test authorities, safety, and operators can sign their names to.

At a program level, "we have CogEW" is only credible if the system can:

1. **Prove what it saw, what it believed, what it chose, what it executed, and what effect occurred, all with bounded uncertainty**
2. **Survive independent replay and regression by a third party with no tribal knowledge**
3. **Defend why the behavior changed and which variable actually drove the delta**

Anything less is just an adaptive algorithm bolted to an EW stack.

Core invariants

A CogEW system is only defensible if these invariants are implemented.

- **Time integrity**: Absolute time, monotonic time, and time-quality metadata are preserved and carried with evidence. Degraded time is segmented and labeled, not "smoothed" into plausibility.
- **Executed-state closure**: Commanded intent and executed behavior are preserved as parallel timelines. Executed ES/EP/EA state is captured at

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

authoritative monitor points near the RF chain, not inferred from command logs.

- **Deterministic contention behavior**: Arbitration under load is fixed and repeatable. Priority classes, thresholds, and drop policies are explicit. Any loss is recorded with time bounds, counters, and reason codes. Silent loss and silent reordering are treated as corruption.

- **Truth Spine and high-rate windows**: A non-preemptible Truth Spine records time, executed-state, constraints, arbitration state, evidence quality, and triggers at a low rate. High-rate capture windows add local detail around decision-relevant events, but never displace the spine.

- **Eligibility gating**: Evidence is either admissible or not. Degraded, incomplete, misaligned, or unknown-loss intervals are automatically excluded from scoring and learning. There is no "use cautiously."

- **Regression-grade provenance:** Evidence is packaged at ingest as versioned objects with hashes and a manifest that binds them to platform identity, configuration, and RF chain hashes; firmware and software builds; schemas and ontologies; time discipline history; calibration lineage; technique libraries; capture policies; and model identifiers.

- **Interface closure**: The platform exports a minimum set of time-integrity-protected, loss-accounted evidence streams through stable, versioned interfaces:
    - Time and time-quality telemetry.
    - Arbitration and loss telemetry.
    - ES/EP/EA executed-state.
    - Constraint and protection state.
    - Transmit verification products with calibration IDs and bounds.
    - Configuration identifiers active during collection.

Facts that are not exported on these interfaces do not exist for certification, regression, or release decisions.

Release rule

Release authority is inseparable from evidence authority:

- No evidence, no release
- Incomplete evidence, no release
- Ambiguous evidence, no release

This applies to scoring acceptance, regression promotion, software and model releases, and operational fielding. If the evidence backbone cannot defend a change, that change does not go to the fleet.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

Objective and scope

Objective

This architect's guide defines the mandatory evidence, data integrity, and control requirements that a CogEW system must meet to be considered a fieldable, defensible capability. This guide deliberately does not prescribe specific algorithms, threat models, or training datasets. It assumes competent model and data engineering. Its purpose is to define the additional evidence and control requirements without which those efforts cannot be credibly tested, repeated, certified, or fielded, regardless of how sophisticated the learning stack may be.
The goal is to close the Verification Gap that kills CogEW in practice: once adaptive behavior cannot be reconstructed with causal truth, you cannot test it, repeat it, certify it, or release it under configuration control without turning your fleet into a science experiment.

This guide draws a hard line: if a system cannot satisfy the requirements below, it is not CogEW. It is an unverified adaptive algorithm bolted to an EW stack.

In this framework, the terms are engineering contracts:

- Testable means effects and failure modes are measured at authoritative monitor points with defensible timing and configuration context. Inference, narrative reconstruction, and "we think it did X" are rejected. If it cannot be measured, it did not happen.

- Repeatable means identical inputs, declared constraints, and declared evidence quality produce explainable outcomes. Deltas are attributable to a specific cause, not buried in silent loss, timing ambiguity, side effects of contention, or uncontrolled configuration drift.

- Certifiable means T&E can bound uncertainty, verify executed-state closure, and sign their name to the result because the evidence package supports independent replay and regression. Certification is not belief in the model. It is confidence in the evidence and in the control boundary.

- Releasable under configuration control means a new build, technique library, or model policy cannot drift into the fleet without audit-grade proof of what changed, why outcomes moved, and the envelope within which the change is valid. "It performed better in the lab" is not a release authority.

The central claim is straightforward: adaptive behavior is operationally meaningless unless it is supported by evidence that withstands independent replication and regression. Without that substrate, CogEW claims are not falsifiable. They are stories built on missing truth.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

Scope

This guide is restricted to the nervous system of the cognitive loop: the deterministic data plane and evidence model that preserves causality at the tactical edge under SWaP-C stress. The “brain” can be whatever a prime wants. If the nervous system cannot prove what the brain sensed, decided, executed, and achieved, the brain does not matter.

Scope includes:

- Engineering of edge ingest under contention, arbitration under saturation, and sustained write behavior without invisible gaps.
- The evidence model required to separate commanded intent from executed behavior and to bind every decision and effect to defensible time and configuration context.
- The control boundary between cognitive application logic and evidence infrastructure, including what the platform must expose through ICDs, at what rates, and with what authority.

This guide is explicit about the “80% reality” of cognitive systems: most of the work is not in the learning algorithm; it is in the data lifecycle. In CogEW, that workload cannot be deferred to post-mission analysis. The data plane is not a pipe. It is the curation engine. If curation, labeling, time-quality marking, loss accounting, and provenance are not automated at ingest, the program accrues data debt that will bankrupt test, regression, and certification.

This guide intentionally avoids debates over algorithm selection and model architecture choices (e.g., RL, DRL, XAI, and similar). Those arguments are irrelevant if the platform cannot preserve causal truth at the edge. The closed loop only exists if the system can characterize what it observed, what it believed, what it executed, and what effect occurred, with bounded uncertainty. If the evidence backbone cannot do that, the loop collapses, and “cognition” is just branding.

Definitions

- **Evidence Backbone:** The deterministic data and metadata substrate that preserves executed-state, time integrity, explicit loss accounting, and provenance so an independent team can reconstruct what happened with bounded uncertainty.
- **Executed-state:** What the system actually did at authoritative monitor points and state machines (for example, EP mode transitions, EA technique identifiers and parameter hashes, limiter or overload or compression activity, derates, and reverts), not what it was commanded to do.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

- **Time integrity:** Preservation of absolute time and monotonic time, plus time-quality metadata (source, discipline state, holdover, discontinuities) such that correlation and attribution remain defensible.
- **Eligibility gating:** Automated rules that prevent degraded, incomplete, or ambiguous evidence from entering scoring and learning pipelines.
- **Configuration control for adaptation:** The ability to attribute outcome deltas to algorithm change, configuration drift, environment change, or evidence-quality change using machine-readable provenance.

Assumptions

The reader is fluent in ES, EP, EA, and closed-loop EW system design and employment. The reader understands contested RF environments and platform SWaP-C constraints, including bus contention, thermal derate behavior, and rigid real-time boundaries. The reader's objective is to make CogEW fieldable, certifiable, and defensible in operations, not to produce publishable demonstrations.

Non-goals

This guide does not attempt to prove CogEW works. It defines the conditions under which CogEW claims are falsifiable. If a system cannot preserve evidence sufficient for independent replay and regression testing, its learning claims cannot be verified, reproduced, or released safely. They are not engineering claims. They are stories.

1. The Verification Gap

CogEW does not fail because teams pick the wrong algorithm or bring too little compute. It fails because causal truth collapses under real platform stress. Once truth is gone, claims about learning are not testable.

The Verification Gap is the inability to explain, with bounded uncertainty, why the system adapted and what it actually executed in the electromagnetic environment.

1.1 Silent loss

The primary failure mode is silent loss. In contention, the system drops truth at deeper layers: firmware, drivers, DMA rings, kernel buffers, IO schedulers, filesystems, and transport stacks tuned for throughput, not evidence.

When those layers shed load without producing loss events with time tags, counters, and reason codes, the evidence stream is not just incomplete; it is also corrupted. Undocumented loss is not "missing samples." It is an unobserved edit to the system history.

That matters because CogEW is a learning system. Learning requires a reliable link between cause and effect. If the system does not know what it failed to record, it does not know what it failed to observe. The model is then trained, tuned, or evaluated on data with unknown gaps and invisible bias. That is how you get confident, wrong conclusions that pass review because the missing truth is never declared.

1.2 Intent vs execution

The second failure mode is treating intent as execution.

A command log is a subjective history. It records what the controller requested. It does not prove what the platform did.

Executed state is an objective history. It is captured at authoritative monitor points and state machines: receiver configuration and protection state, EA technique identifiers and parameter hashes, timing and dwell state, derates, limiter activity, overload or compression flags, and any arbitration decisions that changed capture or action. If a command was reordered, throttled, clipped, or never emitted due to constraints, the command log still appears correct.

Crediting the model for an effect it did not execute is false learning. Blaming the model for a miss caused by platform contention is also a form of false learning. Without executed-state closure, attribution is fiction.

1.3 Post hoc repair and invented history

The third failure mode is the attempt to repair gaps after the fact.

For CogEW evidence, interpolation is hallucination. Any post hoc reconstruction that invents intermediate values, aligns ambiguous time points, or fills gaps with assumptions fabricates a causal structure that never existed in the evidence used to credit adaptation.

That might be acceptable for visualization. It is not adequate for certification, regression, or adaptation credit. If you cannot prove what happened, you do not get to approximate it and then treat the approximation as truth.
Traditional logging and post-mission reconstruction were designed for static execution models where intent and execution were tightly coupled, and loss was rare or obvious. In adaptive systems, that coupling breaks under load, and loss becomes silent. Once that occurs, the sortie cannot be replayed with independent causality. Regression results are not repeatable. Performance deltas cannot be attributed. The learning mechanism cannot be validated as a mechanism.

1.4 Operational impact

This is not a documentation problem. It is an operational stop.

If a system cannot produce audit-grade evidence that separates intent from execution and explicitly declares loss and time quality, it remains non-certifiable for operational use. Test authorities cannot sign it. Safety reviewers cannot bound its behavior. Units cannot trust adaptation they cannot explain. Without an evidence backbone, CogEW is a “Red-X” by design.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

The rest of this guide defines the minimum deterministic evidence behaviors required to close that gap: explicit loss accounting, executed-state capture, time integrity, eligibility gating, and regression-grade provenance. Without those, CogEW claims are stories built on missing truth.

2. Cognition is a causal system

CogEW is not an algorithmic problem. It is a causal systems problem.

An algorithm produces outputs. A cognitive system must explain outcomes.

In a real EW platform, a command is not a fact. It is an intent. The moment a decision leaves the cognitive logic and enters the hardware and firmware stack, it is subject to arbitration, throttling, derating, protection mechanisms, and timing distortion. The burden of proof does not lie in the decision software. It lives in the authoritative state machines that determine what actually happened.

Any CogEW system that treats command logs as evidence is asserting belief, not proof.

The electromagnetic environment and the cognitive algorithms that act on it may be stochastic. The evidence backbone and contention rules cannot be. Their behavior must be deterministic or explicitly parameterized so that, given the same inputs and random seeds, an independent team can replay, bound, and explain the causal chain.

2.1 The four cognitive artifacts

For cognition to be defensible, the system must preserve four distinct artifacts as independent records: **belief, decision, action, and effect.**

- Belief is the system assessment of the electromagnetic environment at a specific time, with declared confidence and uncertainty.
- A decision is the policy selection the system applies to that belief.
- Action is the executed behavior captured at authoritative monitor points, not inferred from intent.
- Effect is the observed outcome in the environment, with stated fidelity, time quality, and capture completeness.

When these artifacts are collapsed or blurred, the system drifts into a circular learning loop.

If belief is updated based on the system's own decision without an independently verified action, the model begins reinforcing its own assumptions. If effect is inferred from intent rather than measured, success and failure are self-attributed. Over time, the system can appear to improve even as it moves further from reality. That is not adaptation. It is self-confirmation with no external check.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

2.2 Bounded uncertainty as a hard requirement

Bounded uncertainty is a requirement, not an option. Every cognitive outcome must state what is known, what was constrained, what was degraded, and what is missing. Trust does not come from apparent performance. It comes from an attribution that can be defended.

CogEW systems must therefore support independent causal reconstruction without analyst inference.

Recorded evidence must be sufficient to drive a high-fidelity digital twin of the mission. A third party must be able to replay the sortie using the evidence alone and reach the same causal explanation for why the system behaved as it did. If the replay requires interpolation, assumptions, or expert gap-filling, the evidence is insufficient. The system is opaque.

Opacity here is an engineering failure. An opaque system cannot be certified, cannot be regressed, and cannot be trusted to adapt safely. Any CogEW implementation that cannot separate intent from fact, belief from action, and decision from effect is not cognitive in an operational sense. It is an unverified algorithm operating without accountability.

Only systems that preserve causality can learn without lying to themselves.

3. Evidence is a first-class data product

A CogEW system does not earn credibility by producing effects. It earns credibility by producing evidence.

Most EW systems are designed for observability. They generate logs, captures, spectra, and telemetry that let an analyst see that something happened. That is necessary, but it is not sufficient. Observability answers the question “what did I see?” Explainability answers the question “why did the system act?” Those are different problems. Confusing them is one of the core reasons CogEW efforts fail in practice.

Observability serves analysts. Explainability serves certification.

Explainability requires evidence that preserves causality across belief, decision, execution, and effect. It requires data products that can survive scrutiny by organizations whose job is to be skeptical: test authorities, safety reviewers, independent red teams, and follow-on integrators. If evidence only “makes sense” when the original developer explains it, the system is not testable. It is vendor-dependent vaporwear.

The requirement is simple: **evidence must be evaluable, not just understandable.**

Evaluable means an independent third party can take the data package, replay the mission, and reconstruct the causal chain without source-code access, tribal knowledge, or analyst interpolation. If the explanation of behavior resides in people rather than in the evidence, the system has already failed the testability requirement.

This framing exposes the main enemy of CogEW: **silent data loss.**

On high-rate, SWaP-constrained platforms, loss is guaranteed. Arbitration, bus contention, thermal derates, DMA starvation, cache eviction, and non-real-time OS behavior are normal. Loss itself is not the failure. Undocumented loss is.

Silent loss is not "missing data." Silent loss is corrupted truth.

When samples are dropped without explicit accounting, the system does not know what it failed to see, execute, or measure. The learner can be credited for an action that never occurred. A derate can be misread as an environmental effect. A logging artifact can be misclassified as novelty. In an adaptive system, this is not a bookkeeping issue. It is a safety issue.

In aviation, an undocumented failure in a flight control loop results in the aircraft being grounded. In CogEW, an undocumented failure in the evidence loop must ground the learning.

Interpolation does not solve this. Interpolation is fabrication.

Interpolation converts unknowns into invented certainty. That might be acceptable in visualization, simulation, or training aids. It is not acceptable in cognitive systems that change behavior based on evidence. When truth is missing, the only valid value is null. The system must say "I do not know," not "here is a guess." Any learning pipeline that trains on interpolated evidence is training on fiction.

This leads to the governing principle of this guide:

If evidence cannot stand on its own, learning is invalid.

Evidence must be treated as a first-class system output. It must be produced intentionally, under deterministic control, with explicit integrity, explicit loss accounting, time integrity, and provenance. Anything less is not a strong evidence base.

4. Evidence invariants for CogEW systems

CogEW does not become real when the model gets smarter. It becomes real when the system can preserve causality under stress and prove what happened without interpretation. That requires invariants. These are hard requirements. If they are broken, the result is not a cognitive system. It is an unverified algorithm riding on wishful logging.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

4.1 Time integrity as the root of causality

Every nontrivial CogEW argument eventually reduces to time correlation: what was observed, when it was believed, when a decision was made, what was executed, and what effect followed. If time is ambiguous, attribution collapses into narrative.

Time integrity in this guide is a three-layer requirement:

- **Absolute time is a wall-clock reference suitable for cross-platform correlation, archive alignment, and enterprise-level regression.**
- **Monotonic time is a strictly non-decreasing local timeline used to prove ordering, dwell, latency, and causality inside the box, even if the absolute clock steps, slews, or drops into holdover.**
- **Time-quality metadata is the truth about the time. Every interval must carry enough metadata to support correlation, including a time source identifier, discipline state, holdover age, a drift estimate or bound, and explicit discontinuity markers such as step, slew, reset, rollover, or oscillator swap.**

When time quality degrades, the system must segment and label the degraded interval. It must not blend degraded time into adjacent good time. It must not smear, smooth, filter, or retro-correct timestamps to make plots look continuous.

Time smoothing is falsification, not repair. If time is missing or degraded, the correct output is a bounded uncertainty statement or a null, not an invented continuity. Eligibility gating only works if degraded time is carried forward as a first-class property.

4.2 Executed-state closure

A command is an intent. Intent is not fact.

Executed-state closure means the evidence backbone preserves parallel timelines for commanded intent and executed behavior and treats executed behavior as authoritative. A command log is a subjective history of what the controller requested. The executed state is an objective record of what the system actually did.

Capture must occur at the last authoritative monitor point before the RF chain, not at a software output buffer, not at a queued message boundary, and not at an interface that can be reordered, dropped, or backpressured under load.

Executed-state evidence must, at a minimum, include the state variables that decide whether the commanded action became a physical reality, such as:

- EP state transitions and protection state, including limiter engagement, overload or compression flags, gain or attenuation state, and filter or tune state.

- EA execution state, including technique identifier, parameter hash and key fields, dwell and timing state, antenna or sector state, and abort or revert events.
- System constraint interventions that change execution, including thermal and power derates, compute saturation, bus arbitration outcomes, and scheduler drops.

If the executed state cannot be proven at authoritative monitor points, the system can be credited for actions it did not accomplish. That turns learning into reinforcement of a fiction.

4.3 Effect bounding and measurement pedigree

An executed state is necessary but not sufficient. CogEW claims usually rely on the effect, which is an RF chain output interacting with the environment. That output must be bounded.

Effect bounding requires transmission verification products with stated fidelity and explicit calibration lineage. The evidence package must bind every verification measurement to the calibration state of the sensor chain that produced it. A number without pedigree is a guess.

Effect bounding must explicitly carry uncertainty terms appropriate to the measurement chain and mission use, including at least:

- Timing uncertainty, including transmit time relative to the decision and relative to the time-quality state.
- Amplitude or EIRP bounds within declared instrumentation limits.
- Configuration context that significantly affects output, including antenna or sector, attenuation, filter or tuning state, duty-cycle constraints, and any limiter action that reshapes the waveform.

If calibration lineage and uncertainty bounds cannot be stated, it is not possible to distinguish actual effectiveness from instrumentation artifact, derate behavior, or mis-correlation. Honest regression is not possible because “system got better” cannot be separated from “measurement changed.”

4.4 Disqualifying condition

Any system that infers execution or effects instead of measuring them at authoritative source points is disqualified under this framework.

Inference can occur within the cognitive loop during decision-making. It cannot be used as evidence. Evidence must be measured, timestamped with quality metadata, attributable to authoritative sources, and packaged so an independent team can reconstruct causality without smoothing, interpolation, or tribal knowledge.

5. Throughput-aligned capture architecture

“Record everything” is not ambitious. It is a failure mode. On a SWaP-C-constrained platform, unconstrained capture becomes a self-inflicted denial-of-service attack against the recorder. You saturate the bus, starve time alignment, trigger silent loss, then claim the system learned from a history that never existed.

The objective is not maximum fidelity.

The objective is maximum causality: preserve enough evidence to reconstruct belief, decision, action, and effect with bounded uncertainty under declared constraints.

This section defines a two-tier doctrine.

5.1 The Truth Spine

The Truth Spine is a continuous, non-preemptible, low-rate capture of causal state and metadata. It is not raw I/Q. It is the skeleton.

High-rate data is the flesh. Flesh without a skeleton is a bag of parts. The system may look instrumented, but posture, motion, and cause cannot be reconstructed.

The Truth Spine exists so an independent team can replay the mission without inference. It is the minimum evidence set that survives saturation and preserves history as executed, not as intended.

A Truth Spine segment must always include:

- **Time integrity:** absolute time, monotonic time, and time-quality metadata such as source identifier, discipline state, holdover age, drift indicators, and discontinuities.
- **Executed-state closure:** ES, EP, and EA executed-state from authoritative monitor points and state machines, not command logs.
- **Constraint state:** thermal, power, and compute derates, protection state changes, receiver overload or compression flags, and any envelope interventions.
- **Capture arbitration:** deterministic backpressure state, drop counters, truncation indicators, and reason codes.
- **Evidence quality:** explicit flags when a segment is degraded, incomplete, or ineligible for scoring or learning.

Non-preemptible means the system is architected so that optional high-rate products cannot displace the Truth Spine. If the Truth Spine drops, the sortie is no longer regressable, and the learning claim is no longer falsifiable.

5.2 High-rate capture windows

High-rate windows are bounded bursts of detailed capture around decision-relevant events. They are preemptible by design, but never silently. If a window is dropped, shortened, or reordered, that fact is recorded in the Truth Spine with explicit loss accounting.

High-rate windows capture detailed products as needed, including:

- Narrowband or channelized I/Q, where attribution requires it
- Intermediate features and classifier outputs
- Verification products used for effect bounding
- Technique-specific context needed to reproduce selection and timing outcomes

Windows must be bounded in duration, bounded in bandwidth, and tied to a specific trigger reason.

5.3 Trigger doctrine

Triggers are mission-critical metadata, not just recording flags. They define why the system concluded that more detail was required at a specific moment. They must be preserved as part of the evidence trail.

Valid trigger classes map directly to the cognitive artifacts and to constraint state:

1. **Attribution shifts:** Triggered when the belief structure changes in a way that affects action selection.
    - Classification changes that alter the hypothesis set
    - Association or deconfliction changes
    - Technique selection changes driven by new attribution

2. **Novelty:** Triggered when the environment moves outside the declared envelope.
    - Out-of-library detection
    - Distribution shift beyond threshold
    - Behavior is inconsistent with current models

3. **Constraint transitions:** Triggered when execution capacity changes.
    - Onset or release of thermal, power, or compute derate
    - Change in bus contention regime
    - Protection state changes that alter the receiver's truth

4. **Confidence changes:** Triggered when decision authority changes.
    - Confidence collapse that forces a fallback or revert
    - Confidence increases that enable a higher-risk technique
    - Learning eligibility thresholds crossed, such as transitioning from ineligible to eligible or vice versa

Every trigger must be recorded in the Truth Spine with trigger class, reason code, key state variables at the time of trigger, requested window parameters, and the window outcome (honored, truncated, dropped) with explicit loss accounting.

5.4 Deterministic backpressure and ordering

Best-effort behavior is not acceptable for CogEW evidence. The capture path must be deterministic under load.

The architecture must provide:

- Fixed arbitration priority in which the Truth Spine wins over optional streams
- Deterministic backpressure behavior with declared thresholds
- A deterministic drop policy for window streams with documented ordering
- Explicit counters and reason codes for any loss or truncation
- No silent reordering across streams

Silent reordering is a corruption of history. If buffering or scheduling can change the timing relationship between belief, decision, and executed state without recording that change, the system produces a false narrative that the model treats as fact.

5.5 Evidence objects, not dumps

Both tiers produce evidence products, not arbitrary files. Each product is a bounded, versioned object with integrity and completeness declared at ingest, including:

- Absolute and monotonic time bounds
- Time-quality for the interval
- Stream identifiers and sample counts
- Hashes for integrity and regression comparison
- Explicit loss accounting and eligibility status

The Truth Spine is the authoritative sortie index. High-rate windows are attachments that only gain meaning because the index makes them attributable.

A CogEW system is only as real as its causal record. The Truth Spine preserves the skeleton of what happened. High-rate windows supply detail where it matters. Triggers bind detail to belief and decision. Deterministic backpressure and explicit loss accounting prevent the system from lying to itself.

6. Deterministic contention behavior under load

Contention is not an anomaly in CogEW platforms. It is the normal operating state once wideband ingest, closed-loop decision logic, verification products, storage writes, timing discipline, and protection behavior all share a single SWaP-C envelope. Any design that assumes otherwise is defining success conditions that do not exist in operations.

When resources become insufficient, the system must not improvise. Improvisation under load injects randomness into the causal chain. Once that randomness reaches the model, it is indistinguishable from a learning error.

Contention is an engineering condition, not an exception.

6.1 Why real platforms always contend

Real platforms contend because of three facts:

- Shared fabrics mean shared failure modes. DMA engines, memory controllers, PCIe fabrics, and storage queues couple streams that appear independent on block diagrams. A burst on one path becomes a timing distortion or loss on another.
- Hard real-time deadlines coexist with best-effort infrastructure. The cognitive loop cannot wait. Evidence cannot misrepresent history. Storage cannot exceed its sustained write rate. Arbitration is guaranteed.
- Protection and derate behavior is part of execution, not background noise. Thermal throttles, power clamps, and receiver protection states directly change what was observed, what could be executed, and when. If these transitions are not recorded as evidence, outcome attribution collapses.

6.2 Arbitration must be deterministic

Arbitration is not a plumbing detail. It is part of the evidence model.

If arbitration is non-deterministic, the system has effectively inserted a random number generator into the causal chain. Two identical sorties with identical inputs can then produce different evidence histories, different learning outcomes, and different conclusions about effectiveness. At that point, repeatability is impossible by definition.

Minimum requirements:

- Arbitration policies must be fixed, documented, and repeatable. Given the same inputs, constraints, and timing, arbitration outcomes must be the same.
- Priority classes must be explicit, with declared preemption rules and thresholds.
- There must be no opportunistic buffering or scheduler behavior that can reorder history without recording that reordering as evidence.
- Regime changes must be explicit. The system must know and record when it transitions between normal operation, backpressured operation, saturation, and degraded modes.

This is not an optimization exercise. It is the difference between a system that can be certified and a stochastic system.

6.3 Non-preemptible vs preemptible evidence channels

The architecture must explicitly separate what may drop from what must not.

Non-preemptible channels are those required to preserve causal truth:

- The Truth Spine
- Time integrity metadata
- Executed-state closure
- Constraint and protection state
- Arbitration state and loss accounting

If these channels drop, the sortie is no longer regressable. The system may continue to operate tactically, but it must self-identify the interval as ineligible for learning and scoring.

Preemptible channels include:

- High-rate capture windows
- Optional detail products
- Auxiliary verification streams beyond the minimum required for effect bounding

Preemptible does not mean invisible. It means allowed to drop with explicit accounting.

6.4 Loss as a state transition

Loss is not “missing data.” Loss is a change in system state that must be recorded with the same rigor as a mode transition.

Every capture and evidence stream must record loss using a defined taxonomy:

- **Dropped**: the data was not captured at all
- **Truncated**: capture began and terminated early
- **Backpressured**: capture rate or timing was altered by arbitration
- **Derated**: capture capability was reduced due to thermal, power, or compute constraints

Each loss event must carry:

- Absolute and monotonic time bounds
- Affected streams and evidence objects
- Attempted versus captured volume with counters
- Reason codes tied to arbitration or constraint state
- An eligibility impact flag for learning and scoring

If the system cannot say what it lost, when it lost it, and why it lost it, it cannot defend any claim about what it learned.

6.5 Arbitration decisions as evidence

Arbitration outcomes are not internal bookkeeping. They are causal facts.

At a minimum, the Truth Spine must preserve:

- Arbitration regime state
- Priority decisions at contention points
- Queue depths or watermarks at decision time
- Drop and truncate counters by stream and reason
- Reorder indicators wherever buffering can alter event order
- Sustained write health and flush latency

If the platform becomes so saturated that it cannot record the evidence explaining saturation, the system must declare that interval invalid for learning. Anything else is self-deception.

CogEW under load must behave like avionics under stress. A flight control system is allowed to degrade. It is not permitted to misrepresent its state.

Silent loss, silent reordering, or undocumented arbitration is the data-plane equivalent of a silent control surface failure. It is a grounding event for learning, even if the mission continues.

A CogEW system is disqualified if, under contention:

- Loss occurs without explicit accounting
- Arbitration outcomes vary non-deterministically
- History can be reordered without being recorded
- Degraded intervals are smoothed or interpolated
- Evidence quality is inferred rather than measured

In that regime, the system is no longer learning from reality. It is learning from its own resource-starvation artifacts.

7. Evidence eligibility and learning gating

Most adaptive systems do not fail because they lack data. They fail because they learn from data they should have rejected.

In CogEW, incomplete or ambiguous evidence is not a weaker signal. It is corrupted. Allowing it to influence scoring or learning does not produce conservative adaptation. It produces false certainty, the most dangerous failure mode an adaptive system can exhibit.

Eligibility gating exists to prevent that outcome.

7.1 Eligibility is a binary property

Eligibility is not an analyst judgment, and it is not a confidence score. It is a binary engineering decision applied at ingest and enforced mechanically.

An evidence interval is either admissible for scoring and learning, or it is not. There is no middle state that can survive test and evaluation.

Eligibility must be evaluated against declared invariants, at a minimum:

- **Time integrity:** If absolute or monotonic time quality falls outside declared bounds, the interval is ineligible. Correlation uncertainty cannot be averaged away by learning.
- **Completeness:** If required executed-state, constraint state, arbitration metadata, or loss accounting is missing or truncated, the interval is ineligible. Partial truth breaks the causal chain.
- **Alignment:** If belief, decision, executed-state, and effect cannot be aligned within stated uncertainty bounds, the interval is ineligible. Misalignment is indistinguishable from false attribution.
- **Loss accounting:** If loss occurred but cannot be bounded in time, scope, and magnitude, the interval is ineligible. Unknown loss is worse than known loss because it contaminates every downstream conclusion.

Eligibility is determined by rule, not by optimism.

7.2 Automated exclusion

Eligibility gating must be enforced automatically by the evidence backbone at runtime. Post-mission filtering, analyst judgment, or ad hoc “cleanup” is disqualifying. Those approaches destroy repeatability and guarantee that different teams will extract different lessons from the same sortie.

When an interval is declared ineligible, that status must propagate into scoring, learning, regression, and reporting. The system must not silently consume degraded evidence “just this once.” That is how adaptive systems teach themselves lies.

Automation here is about removing subjectivity from the learning boundary.

7.3 Separation of collection authority and learning authority

A cognitive system must not be allowed to grade its own homework.

The collection authority and the learning authority must be separated by design. The collection side produces evidence with declared quality, completeness, and loss accounting. The learning side consumes only evidence that has passed eligibility gating according to fixed, auditable rules.

This separation acts as a firewall.

Without it, the system will ratify its own decisions. Belief contaminates observation. Decision contaminates evidence. The model reinforces its own behavior and labels it as an adaptation.

Eligibility gating is what keeps belief updates anchored to independently verifiable facts.

7.4 Discarding evidence as discipline

Discarding evidence is not waste. It is discipline.

CogEW systems can learn on operational timescales, sometimes from a single encounter. In that regime, one corrupted lesson is more dangerous than many missed ones. Ignorance is stable. False certainty is not.

The system must prefer “I do not know” over “here is a guess.” That choice preserves trust, enables regression, and keeps adaptation reversible.

The governing rule is simple:

If the evidence is incomplete or ambiguous, the lesson is discarded.

8. Regression-grade integrity and provenance

CogEW does not become fieldable by showing adaptation once. It becomes fieldable when adaptation can be repeated, audited, and explained over time, across builds and platforms, without relying on personalities, tribal knowledge, or post-mission reconstruction.

That is a provenance problem.

Without regression-grade provenance, every performance delta is non-attributable. Engineers end up arguing whether a change came from the model, the technique library, a clock discipline event, thermal derate, bus contention, a new FPGA image, a driver update, a calibration shift, or silent loss that nobody recorded.

8.1 The sortie manifest as a system output

The sortie manifest is not documentation. It is evidence.

The manifest must be generated at ingest by the evidence backbone, not built later by analysts. If the manifest is constructed after the fact, the chain of custody for causal truth is already broken. The system has already encountered the failure modes this guide aims to prevent: missing intervals, timing ambiguity, reordering, truncation, and “best effort” gaps.

The evidence backbone owns manifest creation because it is the only component that can assert what was observed, what was captured, what was dropped, and what constraints were active at the moment of capture.

8.2 Versioned data products at ingest

Evidence must be packaged as versioned data products at ingest. “Raw bits plus an explanation” is not a data product. It is debt.

A versioned evidence product means the system declares, in machine-readable form:

- The schema version governing each evidence object
- The packaging rules that define object boundaries, alignment, and required fields
- The capture policy identifier that governed arbitration and trigger behavior
- The minimum admissibility rules are used for eligibility gating

If any of these change, the version changes. There is no silent format drift. There is no “same field name, different meaning.” There is no “ask the veteran on the team how to read this.”

8.3 Chunked evidence objects with integrity

Evidence must be segmented into bounded, independently verifiable objects.

Minimum requirements:

- Deterministic chunking is whether time-bounded, event-bounded, or a clearly defined hybrid
- A cryptographic hash per chunk
- Hashes stored in the manifest as external truth anchors
- Integrity protection for the manifest itself

This is not cosmetic security. It is how regression replay is prevented from becoming “close enough” with differences that nobody can detect or prove.

8.4 Calibration lineage

Calibration lineage prevents physics drift from masquerading as cognition.

In EW systems, small shifts in receiver sensitivity, gain staging, limiter behavior, antenna pattern, or timing discipline can appear as target adaptation or model improvement. Without calibration identifiers bound to the evidence, learning cannot be distinguished from instrumentation drift. False correlations will be drawn and operationalized.

Calibration lineage must be bound to any evidence used to measure effects, validate execution, or score outcomes. If calibration state cannot be stated, measurement fidelity cannot be claimed. If measurement fidelity cannot be claimed, a learning mechanism cannot be certified.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

8.5 Machine-readable sortie package index

The sortie package index is the manifest that enables additive evidence across the lifecycle. It binds the collection to a configuration baseline and declares the context needed for independent replay and delta attribution.

At a minimum, it must bind:

Identity

- Platform identity
- Evidence backbone identity
- Evidence backbone build identifier

Hardware configuration

- Hardware configuration hash, including RF chain configuration where applicable
- FPGA bitstream identifiers
- Sensor front-end configuration identifiers

Software and firmware

- Firmware build identifiers
- Software build identifiers, including image hashes or equivalent
- Driver and OS build identifiers that affect timing, DMA, or arbitration

Schema and packaging

- Evidence schema versions
- Sortie package format version
- Metadata ontology version that controls field meaning under configuration control

Time discipline

- Time source identifiers
- Discipline state history, including locked, holdover, and free-run states.
- Holdover age and drift estimates
- Discontinuities and resynchronization events

With this index in place, a program can perform true regression across sorties, builds, and platforms and know what changed, when it changed, and why behavior shifted. Without it, adaptation is indistinguishable from uncontrolled drift.

9. Configuration control and release authority

Configuration control is where CogEW either becomes a weapon system or is disqualified from becoming one. Adaptive behavior that produces effects without attribution is not a capability. It is a liability. Success without explanation is not reassuring. It creates false confidence that will not survive regression, fleet variation, or operational surprise.

In CogEW, release authority is inseparable from evidence authority. If the system cannot prove why the behavior changed, approving that change is not engineering. It is a bet on coincidence.

9.1 Golden rule

CogEW programs require a non-negotiable release contract:

- No evidence, no release
- Incomplete evidence, no release
- Ambiguous evidence, no release

This rule applies to:

- Scoring acceptance
- Regression promotion
- Software, firmware, and model release
- Authorization for operational employment

Evidence completeness and integrity are gating conditions, not advisory inputs. If the evidence backbone flags degraded time integrity, undocumented loss, missing executed-state, or broken provenance, the release gate remains closed.

This gives the program office and test authorities an objective basis to stop progression. The system may not have failed, but the evidence has. That is sufficient to halt release.

9.2 The quadrant of attribution

Every observed performance delta in a CogEW system originates from one or more of four sources:

1. Algorithm or policy change
2. Configuration drift in hardware, firmware, software, calibration, or timing
3. Environmental effects
4. Evidence quality degradation

Claiming learning requires subtracting the other three.

If the system cannot show, in machine-readable evidence, that configuration, environment, and evidence quality were held constant or explicitly bounded, the delta cannot be attributed to adaptation. Without that subtraction, “learning” is an assertion, not a fact.

This quadrant is the core attribution test. Failing it means the system is not improving. It is changing without accountability.

9.3 Preventing autonomous drift

Autonomous drift occurs when adaptive systems change without a regression discipline. It often appears as gradual improvement, intermittent regression, or unexplained variance that engineers rationalize after the fact. Drift is not primarily a model problem. It is a configuration control problem.

Regression-grade evidence is the defense.

Every adaptive change must be evaluated against a known baseline using eligible evidence. If the system cannot reproduce prior behavior under comparable conditions, it has not learned. It has drifted. Drift may appear acceptable on a bench, but at fleet scale it becomes unbounded risk.

Ignorance is survivable. False certainty is not.

9.4 Evidence as release authority

In CogEW, the evidence backbone functions as an independent authority for release. Cognitive logic can propose changes, but it cannot certify itself. Certification must come from a deterministic, independent evidence substrate that has no incentive to justify success.

This mirrors flight-critical practice. Flight control laws do not self-certify. They are certified by independent instrumentation and recorders that prove what the aircraft actually did under defined conditions. In CogEW, the evidence backbone plays the role of the flight data recorder. The AI is the pilot. The pilot does not write the accident report.

The hard line is simple: if the evidence cannot defend the change, the change does not release.

10. Interface closure

Interface closure is where CogEW either works or fails. If the platform does not export the facts required to prove what happened, the evidence backbone cannot reconstruct causality, eligibility gating cannot protect the learning loop, and test authorities cannot certify anything beyond a narrative. A recorder can only preserve what the platform exposes. If the platform never publishes a fact, that fact does not exist for assurance.

The evidence backbone is not passive storage. It is the Truth Spine of the cognitive loop. That spine is a minimal, non-preemptible set of time-aligned state, constraint, and execution artifacts that survive contention and allow an independent team to replay the mission without inference. High-rate capture windows add fidelity. They do not replace the spine. Without the spine, high-rate data is expensive and ambiguous.

SWaP-C is not background context. It is part of causality.

In a real platform, SWaP-C constraints are causal inputs that change what the system can observe, decide, execute, and verify. Thermal derates, power ceilings, bus saturation, and compute throttling are not “engineering details.” They are the mechanism by which a correct decision becomes a non-event, a delayed event, or a different event. If these constraint transitions are not exported as authoritative evidence with time integrity, then every downstream claim about adaptation is contaminated. The system cannot distinguish adversary behavior from self-limitation, and neither can test authorities.

10.1 Open parsing as a certification requirement

Evidence that requires tribal knowledge to interpret is not evidence. It is vendor memory.

Interface closure requires schema stability: versioned, self-describing records that remain parseable across missions, software baselines, and integration stacks. That is what makes evidence portable, comparable, and regression-grade. It also prevents the failure mode in which “truth” can be recovered only by the original developer with proprietary context. That is not compatible with independent evaluation.

10.2 Minimum evidence streams

The platform must export a minimum set of evidence streams that are authoritative, time-integrity protected, and loss-accounted. “Minimum” means the smallest set that preserves belief, decision, action, and effect with bounded uncertainty. The evidence backbone cannot certify what it cannot see.

The required streams are:

- **Time and time-quality telemetry:** Absolute time, monotonic time, and time-quality metadata must be explicit and continuous, including discipline state, holdover age, drift bounds, and discontinuity events. Degraded time must be segmented and labeled. It must never be blended or smoothed to repair a timeline.
- **Capture arbitration and loss accounting telemetry:** Arbitration is part of the causal chain. If the system drops, truncates, backpressures, derates, or reorders any stream, that must be recorded as a state transition with reason codes and counters. Loss is not an absence. It is an event. Undocumented loss is corruption.
- **ES receiver configuration and EP executed protection state:** To claim what was observed, the platform must expose receiver configuration and protection state at the moment of observation, including gain or attenuation, tuning, bandwidth or mode state, overload or compression flags, limiter engagement, and any EP interventions. If the front end was saturated, that is not an analyst detail. It is the ground truth.

- **Executed EA and EP state with technique identifiers and parameter hashes:** A command is intent, not fact. The platform must export the executed state from authoritative state machines: technique identifiers, parameter hashes, dwell and timing state, antenna or sector state (where applicable), and executed EP actions. This is an executed-state closure. Without it, the system is grading its own homework.
- **Constraint, derate, and guardrail intervention events:** Thermal, power, compute, and bus contention transitions must be exported with timestamps and reason codes. Guardrail interventions, reverts, and safing actions must be explicit. Constraints are part of the explanation for why a behavior did or did not occur.
- **Transmit verification products with calibration identifiers:** Effects must be bounded, not assumed. The platform must export verification products tied to calibration lineage and validity flags, including uncertainty bounds. If the measurement chain is saturated or out of family, that must be declared. Uncalibrated “verification” is still inference.
- **Configuration identifiers active during collection:** Technique library identifiers, capture policy identifiers, model identifiers, software and firmware build identifiers, and schema versions must be exported so that every evidence segment binds to a specific baseline. Without this, learning cannot be separated from drift.

10.3 Failure attribution rule

Interface closure ends with a simple contract:

If the platform does not expose a fact as a stable, versioned interface, that fact does not exist for certification, regression, or release decisions.

In practice, that means no claim about learning, effectiveness, or safety can rely on internal behavior that is not visible as evidence on the defined interfaces.

11. Integration boundaries and responsibility split

CogEW systems fail when responsibility is blurred. Effects can be generated, but truth cannot be preserved. This section defines the mandatory boundary between cognitive logic and evidence infrastructure, ensuring accountability while keeping innovation unconstrained.

The cognitive application makes decisions. The evidence backbone establishes fact. These two functions must be integrated, but not fused.

11.1 Nervous system contract

The boundary between cognitive logic and the evidence backbone is an engineering contract, not an architectural preference. The cognitive application functions as the brain: it forms beliefs, selects actions, and adapts under constraints. The evidence backbone functions as the nervous system: it provides sensory ground truth and records execution as an authoritative fact.

This contract is satisfied only if an independent team can reconstruct belief, decision, action, and effect using preserved evidence without inference, interpolation, or access to proprietary application internals.

If that condition is not met, the system is opaque by definition and cannot be certified.

11.2 Responsibility split

Cognitive application responsibility

The cognitive application owns:

- Belief formation, decision logic, and adaptive policy
- Selection of techniques, priorities, and guardrails
- Consumption of evidence that has already passed eligibility gating

The cognitive application does not own:

- Determination of what was actually executed
- Determination of time integrity or data completeness
- Determination of evidence eligibility
- Certification of its own behavior

A system that grades its own performance is not adaptive. It is circular.

Evidence backbone responsibility

The evidence backbone includes:

- Preservation of the executed state at authoritative monitor points
- Time integrity, including absolute time, monotonic time, and time-quality metadata
- Deterministic arbitration under contention, with explicit loss accounting
- Automated eligibility gating that prevents corrupted evidence from influencing scoring or learning
- Generation of regression-grade evidence packages at ingest, not post-mission

The evidence backbone is not an analytics platform and does not implement cognition. Its role is to make truth durable, portable, and auditable under SWaP-C stress.

11.3 Integration support versus application ownership

Integration support means enabling the cognitive application to meet the nervous system contract without inheriting ownership of cognition. This includes:

- Enforcing interface disclosure for executed-state, time quality, constraints, and arbitration behavior.
- Providing deterministic ingest, arbitration, and sustained write behavior under load.
- Generating ingest-time manifests that bind evidence to platform identity, configuration hashes, software baselines, schema versions, and calibration lineage.
- Preserving evidence bit for bit so replay fidelity is limited only by the fidelity of the original measurements.

Application ownership remains with the entity implementing cognition. This separation allows best-of-breed algorithms to evolve independently while the evidence substrate remains stable across mission cycles.

11.4 Why tight coupling is disqualifying

Tight coupling between the cognitive application and evidence infrastructure destroys regression and certification in three predictable ways.

First, it enables self-grading. If the cognitive logic can influence what evidence is recorded, prioritized, or considered valid, the system can silently erase its own failures. That is not adaptation. It is falsification.

Second, it breaks independent replay. If mission reconstruction requires application-specific heuristics, undocumented state, or tribal knowledge, the evidence is no longer portable. Vendor-dependent replay is not replay. It is an interpretation.

Third, it collapses attribution. Without separation, a performance delta cannot be decomposed into algorithm change, configuration drift, environmental variation, or evidence-quality change. Regression becomes impossible, and release decisions become a bet on coincidence.

In that regime, the system is correctly treated as permanently “Red-X” by test, safety, and airworthiness authorities.

11.5 Hard rule

The evidence backbone must be capable of contradicting the cognitive application.

This is not adversarial. It is the minimum requirement for assurance. If the nervous system cannot independently verify what the brain did, it cannot claim learning, cannot be certified, and cannot be released into operational service under configuration control.

That boundary is what makes CogEW real.

12. Disqualifiers and failure modes

This section is the enforcement mechanism for the guide. The previous sections define what the evidence backbone and platform must do. This section defines the conditions under which a system is not CogEW under this framework, and the failure patterns that appear when those conditions are ignored.

12.1 Disqualifiers

Any single condition below is sufficient to disqualify a system from being treated as CogEW under this guide.

- **No executed-state closure**: If the system cannot provide authoritative executed-state from ES, EP, and EA monitor points and state machines, it cannot claim behavior. Command logs are intent, not fact. Without executed-state closure, attribution, regression, and safety review are invalid.

- **Smoothed or repaired time**: If time discontinuities, holdover intervals, drift, or discipline loss are smoothed into a continuous timeline, the evidence is falsified. A broken timeline with explicit quality metadata is admissible. A cosmetically “fixed” timeline is not.
- **Silent loss**: If drops, truncation, backpressure, derates, or reordering can occur without explicit accounting of time bounds, counters, and reason codes, the evidence history is corrupted. Undocumented loss is not missing data. It is an unobserved edit to the record.

- **Non-deterministic arbitration under contention**: If arbitration and buffering behavior can change run to run under comparable inputs and constraints, repeatability is impossible. Non-deterministic arbitration injects randomness into the causal chain that is indistinguishable from model behavior.

- **Inference instead of measurement for execution or effect**: If execution or effects are inferred rather than measured at authoritative source points, the system fails this framework. Inference may support decision logic. It cannot be used as evidence.

- **No eligibility gating:** If degraded, incomplete, or ambiguous evidence can enter scoring or learning pipelines without binary admissibility checks at ingest, the system will learn from corruption. Partial truth is treated as full truth, and the causal chain is broken.

- **No regression-grade provenance**: If evidence cannot be bound to configuration hashes, firmware and software builds, schema versions, time discipline history, calibration lineage, technique libraries, capture policies, and model identifiers, performance deltas are not attributable. Without attribution, regression is argument, not engineering.

- **Unbounded or uncalibrated effect claims**: If the system cannot provide effect measurements with stated uncertainty and calibration pedigree, "effectiveness" is a belief claim, not a test claim. Effects must be bounded by measurement, not assumed.

- **Broken interface closure**: If the platform does not export the minimum evidence streams defined in the interface closure section using stable, versioned schemas, then those facts do not exist for certification, regression, or release decisions.

Any system that meets one or more of these conditions may still adapt and may still produce effects. It is not CogEW in any certifiable sense.

12.2 Failure modes when disqualifiers are ignored

Ignoring these disqualifiers produces predictable failure patterns.

- **Fraudulent learning**: The system is credited for actions it did not execute because intent is treated as execution or constraint events were not recorded. The learner reinforces a fiction and appears to improve.

- **Phantom performance deltas**:Apparent changes in performance are driven by timing discipline shifts, calibration changes, contention behavior, or configuration drift. Without provenance, these deltas look like learning but cannot be reproduced or explained.

- **Circular learning**: Belief and effect are updated based on the system's own decisions without independent verification of action. The model reinforces its own assumptions and labels it adaptation.

- **False correlation**: Smoothed time, blended intervals, and interpolated gaps manufacture causal structure. The system "learns" relationships that are artifacts of reconstruction, not of reality.

- **Non-repeatable test results**:Under non-deterministic arbitration or undocumented loss, identical scenarios produce different evidence histories and outcomes. Test authorities cannot bound uncertainty or certify the learning mechanism.

- **Unbounded autonomous drift**: Adaptive changes accumulate without regression discipline or hard release gates. Behavior evolves in ways that cannot be tied to specific data, conditions, or configuration baselines. At fleet scale, risk becomes unbounded.

- **Vendor-dependent folklore**: Evidence that only "makes sense" when interpreted by the original developer makes the system dependent on personalities and tribal knowledge. Independent evaluation and long-term sustainment fail.

Under this guide, these are not acceptable characteristics of a fielded CogEW system. If a program exhibits them, the correct disposition is straightforward: the system is Red-X for adaptation until the evidence backbone, interfaces, and configuration control are brought into compliance.

13. What Cognitive EW means when done correctly

“Cognitive EW” is not a synonym for “AI in the loop.” It is not a threat library with a smarter search function, and it is not a set of classifiers stapled to a jammer. A system is cognitive only when it can adapt in contact and still remain testable, repeatable, certifiable, and releasable under configuration control.

The defining characteristic is not that the system changes. The defining characteristic is that the system can prove what changed, why it changed, what it actually did, and how confident it is in what it believes happened.

13.1 Grounded definition

Cognitive EW is bounded adaptation, backed by deterministic evidence, operating under explicit uncertainty, and governed by regression-grade control.

Each phrase is an engineering constraint:

- **Bounded adaptation**: The system is allowed to change its behavior only inside a declared envelope. Constraints are first-class inputs: SWaP-C, timing budgets, thermal and power derates, bus contention behavior, RF front-end protection state, and guardrail interventions. If the system adapts and cannot state which constraint forced the change, it is not bounded. It is drifting.
- **Deterministic evidence**: The system preserves causal truth at the tactical edge. The environment can be stochastic; the evidence backbone is not. Time integrity is preserved, arbitration is repeatable, and loss is explicit. Executed-state is captured at authoritative monitor points, not inferred from commands. When a system claims it jammed, protected, or listened, the evidence must show what executed and with what fidelity.
- **Explicit uncertainty**: The system does not hide ambiguity. It states uncertainty at every stage of the cognitive loop: what it observed, what it believed, why it chose an action, and what effect it claims occurred. If time quality is degraded, capture is incomplete, calibration pedigree is unknown, or the RF chain is saturated, the system declares the affected evidence ineligible for learning or scoring.
- **Regression-grade control**: Changes are not accepted because they “seem better.” They are accepted because they are attributable. A performance delta is traceable to a specific variable: algorithm or policy, technique library, platform configuration, environmental change, or evidence-quality change.

Evidence packages carry machine-readable provenance so independent teams can reproduce results without tribal knowledge.

13.2 The cognitive loop as four artifacts

A CogEW system is cognitive only if it preserves four distinct artifacts for every meaningful event:

- **Belief**: what the system thought the RF picture was, including confidence and uncertainty.
- **Decision**: what it chose to do and why, including policy version and constraint context.
- **Action**: what actually executed at authoritative state machines and monitor points.
- **Effect**: what was measured as a result, with calibration lineage and stated bounds.

Collapse any of these into a single log stream and the result is a narrative, not a weapon system. The most common failure mode is circular learning: the system credits itself for an outcome based on its own intent rather than on verified executed action and bounded effect.

13.3 Necessary and sufficient conditions

A system can claim “Cognitive EW” under this guide only if all of the following are true:

- **Executed-state closure exists:** Commanded intent and executed behavior are preserved as parallel timelines. Executed-state is captured at authoritative monitor points, as close to the RF chain as practicable.

- **Time integrity is preserved:** Absolute time, monotonic time, and time-quality metadata exist and are segmentable. Degraded time is labeled, not blended or smoothed.

- **Contention behavior is deterministic:** Arbitration under load is repeatable. Any loss is an explicit, recorded state with time bounds, counters, and reason codes. Silent loss is treated as corruption, not inconvenience.

- **Eligibility gating is enforced:** Degraded or incomplete evidence is automatically excluded from scoring and learning. The system is allowed and required to say “unknown” when conditions demand it.

- **Provenance is regression-grade:** Evidence packages bind to configuration hashes, build identifiers, schema versions, time discipline history, calibration lineage, technique library identifiers, capture policy identifiers, and active model identifiers.

- **Interface closure is implemented:** The platform exports the minimum evidence streams with stable, versioned schemas: time/time-quality,

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

arbitration and loss, ES/EP/EA executed-state, constraint and protection state, effect verification products with calibration IDs, and configuration identifiers.

If any one of these is missing, the system may still be adaptive and may still be dangerous. It is not cognitive in any certifiable sense.

13.4 Operational meaning

When done correctly, CogEW is learning in contact that survives scrutiny. The system can encounter novelty, adapt inside declared bounds, and later withstand hostile questioning from test authorities, safety, and operators. It can be independently replayed without inference. It can be regression-tested across builds without hand-waving. It can be fielded without confusing luck for performance.

The practical test is simple: if the program cannot reconstruct and defend its own causal story from evidence alone, then it does not have a cognitive capability. It has an unverifiable algorithm.

13.5 Center of gravity

CogEW is not a label applied to an algorithm. It is a claim about assurance: that an adaptive EW system can change its behavior in contact and still remain testable, repeatable, and certifiable.

The evidence backbone is the center of gravity that makes that claim real. It is the enabling subsystem that makes CogEW falsifiable, certifiable, and fieldable. It is the nervous system that turns adaptation from an internal narrative into an engineering claim.

Without that backbone, “learning” is indistinguishable from drift, coincidence, or unobserved failure. With it, CogEW becomes a controlled, testable capability: bounded adaptation operating under explicit uncertainty, governed by deterministic truth, and releasable under configuration control.

Appendix A

Implementation checklist for CogEW programs

If a program cannot answer “yes” to these with actual evidence, it does not meet this guide.

Time and causality

- Absolute time, monotonic time, and time-quality metadata are implemented and carried with all relevant evidence streams.
- Degraded time intervals are segmented and labeled; timelines are never smoothed or retrocorrected to hide discontinuities.

Executed-state and interfaces

- Executed-state closure exists: ES/EP/EA executed behavior is captured at authoritative monitor points, independently of command logs.
- The ICD exports the required evidence streams: time/time quality, arbitration and loss, ES/EP/EA executed state, constraint and protection state, transmit verification products with calibration IDs, and configuration identifiers.

Contention and loss

- Arbitration under contention is deterministic, with documented policies, priority classes, and thresholds.
- Loss taxonomy is implemented: dropped, truncated, backpressured, and derated, each with time bounds, counters, and reason codes.
- Non-preemptible channels (Truth Spine, time integrity, executed state, constraints, arbitration state) cannot be displaced by optional or high-rate streams.

Truth Spine and capture architecture

- A Truth Spine is defined and implemented as continuous, non-preemptible, low-rate capture of time, executed-state, constraints, arbitration state, triggers, and evidence quality.
- High-rate capture windows are bounded in duration and bandwidth, tied to explicit triggers, and their loss or truncation is recorded in the Truth Spine.

Eligibility and learning

- Eligibility gating rules are codified as binary checks and enforced at ingest.
- Degraded or incomplete evidence is automatically excluded from scoring and learning.
- Collection authority and learning authority are separated; the system does not grade its own homework.

Provenance and manifests

- Evidence is packaged at ingest as versioned data products with deterministic chunking and per-chunk cryptographic hashes.
- A machine-readable sortie manifest is generated at ingest that binds evidence to:
  - Platform and evidence backbone identity and build IDs
  - Hardware and RF chain configuration hashes
  - Firmware and software builds, drivers, and OS versions that affect timing, DMA, or arbitration
  - Evidence schemas, package formats, and ontologies
  - Time source, discipline history, holdover, drift bounds, and discontinuities
  - Calibration identifiers and validity for any measurement used to claim effect or execution

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com

- Technique library, capture policy, and model identifiers active during collection

Configuration control and release

- "No evidence, no release" is wired into program processes for scoring, regression, and field release.
- Performance deltas are always analyzed across the four attribution axes: algorithm/policy, configuration, environment, and evidence quality. Learning is not credited until the other three are bounded.

Integration boundaries

- Responsibility split is documented: the cognitive application owns belief, decision, and policy; the evidence backbone owns executed-state, time integrity, contention behavior, eligibility gating, and provenance.
- The evidence backbone can contradict the cognitive application; the application cannot suppress or edit evidence about its own failures.

Disqualifiers

- The program explicitly adopts these as hard "no-go" conditions for calling a system CogEW:
    - Smoothed or repaired time.
    - Silent loss.
    - Non-deterministic arbitration under contention.
    - Inference in place of measurement for execution or effect.
    - No eligibility gating.
    - No regression-grade provenance.
    - Broken interface closure.
    - Success without attribution.

If a design or implementation cannot check these boxes with real data and interfaces, it is not "almost CogEW." It is a conventional adaptive system that has not earned the right to be called cognitive.

26460 Corporate Avenue, Hayward, CA 94545
(650) 367-2011 | ampex.com