

Cadence — system whitepaper

Version 0.1.140 · last updated June 25, 2026 · macOS 26 native research tool · all perception on-device · cloud-orchestrator synthesis (raw media never leaves the Mac) · provisional patent filed

1. Premise

Cadence captures *your* communication on video — voice, gaze, facial dynamics, conversational timing — and extracts a research-backed feature stream entirely on-device. Two collection surfaces feed the same pipeline:

- **Live capture** from any video the user is participating in. Cadence detects when a video surface is active (native meeting apps; browser tabs; LMS-embedded sessions on Canvas / Blackboard / Brightspace / Moodle; supported platforms include Zoom, Teams, Google Meet, BigBlueButton) and starts recording features in 30-second windows. The architecture is platform-independent — adding a new surface is a Bridge-detector entry, not a re-architecture.
- **Upload** of past recordings (audio or video) imported from disk. The reprocessor runs the identical perception → reasoning path on the file. A year of telehealth visits a clinician already has on file can be analyzed as a single read (see §3.14); an instructor's session-by-session lecture recordings can be processed in bulk against a single subject's longitudinal baseline.

A reasoning layer — a foundation model grounded by a retrieval set of peer-reviewed behavioral-signal research and a per-user longitudinal baseline — surfaces indicators worth discussing with a clinician. The literature seed leans on autism work (where multimodal communication signals are most-studied), but the architecture is signal-general: prosody, gaze, facial dynamics, and conversational timing carry information about depression, Parkinson's, schizophrenia, dementia, ADHD, fatigue, and acute stress, and the indicator vocabulary is built around those underlying signals, not a single condition. The grounding layer is **bundled seed + live OpenAlex validation** — every citation the reasoner emits is resolved against the OpenAlex scholarly graph (250M+ papers, free, no auth) for retraction detection and topic-grounding cross-check, so the evidence base stays current between releases without re-shipping the app (see §3.6).

Cadence is **not a diagnostic device**. It does not produce probabilities of any condition. It produces correlational indicators with cited research, intended as data the user (or family) can bring to their care team.

The scope is deliberately self-only: Cadence analyzes the data captured from your own machine — your microphone, your camera. Other participants in a live session are never the subject of analysis even when their audio passes through the process tap, and even though the screen capture sees the gallery view. Uploaded recordings inherit the same scope: only the user-identified speaker is analyzed; other voices feed talk-time / turn-taking aggregates but are not modeled as subjects.

1.5 Patent context

The architecture described in this document is the subject of a provisional patent application: **"Multimodal Behavioral Signal Intelligence Layer for Integration into Existing Software Platforms."** The provisional names six steps in its example independent method statement, and Cadence's implementation maps onto them directly:

1. **Receive behavioral signal data** from a host platform — Cadence's `CaptureCoordinator` ingests mic, remote-tap, camera, and screen from detected meeting apps. The **Cadence Bridge** Chrome extension extends this to *any* browser- or LMS-iframe-hosted meeting (Google Meet, web Zoom, web Teams, BigBlueButton inside Canvas / Blackboard / Brightspace / Moodle) — the only piece of the system that needs to be aware of the host platform. This is the provisional's "operates independent of any specific video or LMS platform" claim made concrete: above the meeting layer, no per-platform integration required (see §3.13).
2. **Extract multimodal behavioral features** — 32 scalar fields per 30-second window across three structs: vocal (`AcousticFeatures` — F0 mean/SD/range, jitter, shimmer, HNR, RMS,

voicedFraction, pauseCount, pauseMeanDuration, speakingRate, turnTakingLatencyMs); ocular and facial ([VisualFeatures](#) — gazeTowardCameraFraction, headYawStdDev, headPitchStdDev, headMovementRMS, meanFixationDurationMs via I-DT, saccadeRatePerSecond, gazeEntropy, blinkRatePerMinute, meanEyeOpenness, expressionEntropy, auActivations covering FACS-13); linguistic ([LinguisticFeatures](#) — wordCount, typeTokenRatio, mattr, disfluencyCount, repetitionScore, meanUtteranceLength). Full enumeration in Appendix A.

3. **Normalize against individual and population baselines** — [CalibrationStore](#) for the user's own voice baseline (Rainbow / Grandfather Passage), [NormativeBands](#) for literature-anchored population bands from a bundled prior, and [BaselineAgent](#) (when the multi-agent orchestrator is enabled) for per-user longitudinal comparisons.
4. **Fuse across modalities** — [RuleBasedReasoner.multimodalConvergence](#) computes the mean pairwise Pearson r of per-window acoustic / visual / facial signals; the [Synthesizer](#) agent performs a late-fusion pass over the seven specialist-agent findings. The [LiveInterventionEngine](#) fires on-device citation-grounded observations whenever three or more behavioral channels above the z-threshold converge in distinct signal families within the same rolling window (see §3.12). This is the provisional's "cross-modal temporal co-occurrence" claim made live, not aspirational.
5. **Longitudinal analysis** — [SessionStore](#) carries every session; the [RepeatabilityStore](#) measures ICC(C,1) and WSCV against literature-grounded priors (Stegmann 2020) so the dashboard can downweight any indicator whose worst-case driver falls below the Fleiss 0.75 clinical-decision threshold.
6. **Generate non-diagnostic indicators inside the host workflow** — [IndicatorKind](#) enumerates **12 indicator kinds** (the orchestrator's output vocabulary); these are computed from the **26 measurements** documented in [docs/METRICS.md](#), each with computation + citation + robustness tag. Each [Indicator](#) carries [confidenceLow](#) / [confidenceHigh](#), citations from a bundled 47-study seed validated against OpenAlex at persist time (retraction-aware + topic-grounded — see §3.6), reasoning, and a repeatability tier; surfaced via the Chrome-extension overlay or the native dashboard without altering the host meeting app.

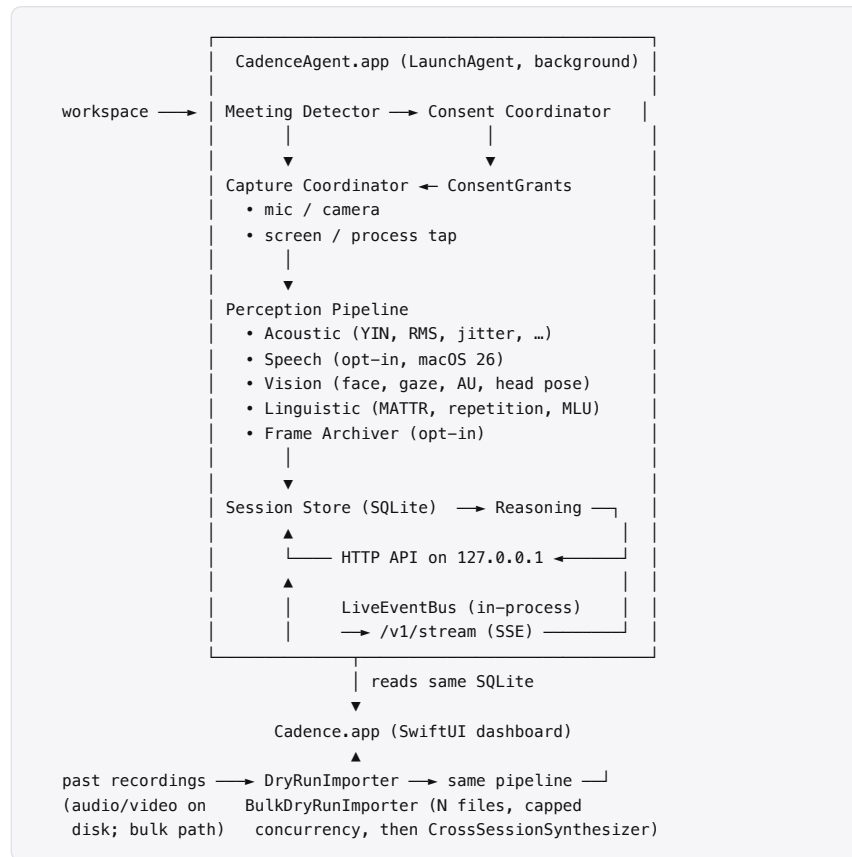
The provisional's representative variations (its "forks") map to existing code as well: platform integration via API/extension (Fork 1), modality-specific pipelines (Fork 2), early/late/hybrid fusion (Fork 3), personal + population baselines (Fork 4), input quality + calibrated confidence (Fork 5), privacy-preserving on-device inference and feature-only federation (Fork 6), explainability and audit trails (Fork 8), active vs passive collection (Fork 9), and aggregated cohort analytics (Fork 10). Multi-context per-user baselines, drift correction, hand/gesture/tremor extraction, modality-disagreement output, and the third-party SDK packaging are roadmap items.

A structurally distinct invention sits on top of this layer: the Research Library Audit subsystem (§6.5) — a model-driven adversarial-verdict pipeline (Auditor → Advocate → Judge → Skeptic) that periodically evaluates Cadence's own grounding corpus, stages deltas through a 30-day local quarantine queue layered onto the immutable bundled corpus at read time, and stamps every downstream indicator with the overlay version it was reasoned against to preserve reproducibility across corpus evolution. Cross-install consistency is delivered in two layers without a central authority: the canonical [research_seed.json](#) bundled in every Cadence release is the baseline sync, and a per-install verdict log published to S3 (gossip-style, public-read peers) feeds a pure-function consensus aggregator that runs locally on every install — converged verdicts (N distinct partner accounts agreeing at confidence $\geq$ threshold) apply to the local overlay autonomously, with no operator click anywhere in the chain. This subsystem is **not** covered by the existing provisional's six method-claim elements (which describe what the system does to incoming user behavioral data, not what the system does to its own grounding corpus) and is being documented in §6.5 + [/patent/README.md](#) for inclusion in a follow-up nonprovisional filing or continuation-in-part addendum. See §6.5 for the operator-facing description and [/patent/README.md](#) § "Self-auditing grounding corpus" for the IP-counsel claim-element-to-code mapping.

Claim language drafted to date covers the Phase 3 architecture as a fully formed set of independent method-claim elements (a) – (k); Phase 4 — the gossip-over-S3 fleet-consensus extension that removes the per-org authority figure and adds operator-out-of-loop semantics — is also drafted as elements (i') and (l) – (p) in [/patent/README.md](#) and is the highest-priority continuation-in-part work given that Phase 4 has been shipped publicly to TestFlight users since v0.1.110.

Cadence is **non-diagnostic by construction**, matching the provisional's central limit on the invention's scope: indicators are computed outputs that highlight patterns, deviations, and changes for human interpretation — never a diagnosis, never a classification of a medical, psychological, or neurological condition.

2. Architecture at a glance



The whole pipeline lives in a single LaunchAgent process; the dashboard reads the same SQLite store and the local HTTP API. No cloud is involved unless the user explicitly enables it for the reasoning step. **The dry-run / bulk path runs the identical perception → reasoning code, so an imported recording produces indicators that are directly comparable with live-captured sessions.**

3. The components

The system is built from coordinating actors plus a reasoning layer. **Perception is on-device** (DSP + Vision + Core ML + macOS 26 SpeechAnalyzer); **synthesis is cloud-orchestrated**. The Mac client ships a single [OpenAIRReasoner](#) that posts the structured feature stream (never raw audio / video / transcript) to the eleplan-backend [cadence-messages-reasoner](#) Lambda; the lambda runs a server-side multi-agent orchestrator that fans the same feature stream out to seven domain-scoped specialists and merges their findings through a final Synthesizer pass (eight agents total — see §4.6 and Appendix B). The earlier on-Mac [AgentOrchestrator](#) / [AgentRegistry](#) Swift code was retired when the orchestrator moved server-side; the [AdminSettings.useMultiAgentOrchestrator](#) flag is left in the struct for on-disk decode compatibility but no longer changes behavior — the multi-agent pipeline is *the* pipeline. Per-modality framing, citation grounding, and the late-fusion Synthesizer remain the architecture's strongest expression of the patent's modality-specific-pipeline (Fork 2) and multimodal-fusion

(Fork 3) claims. The pipeline does not use tools or self-iterate — it is not autonomous in the AutoGPT sense.

A second reasoner path stays on the Mac: `FoundationModelsReasoner` against Apple Intelligence. It does NOT run per-session synthesis (the prompt is too big for the 4096-token context). It powers the live intervention engine (§3.12) — short, structured mid-session observation cards produced entirely on-device.

3.1 Meeting Detector

Detects when a meeting is starting and tells the rest of the pipeline. Sources:

- `NSWorkspace.didLaunchApplicationNotification` and `NSWorkspace.didActivateApplicationNotification` filtered to a curated bundle-ID set (Zoom, Teams, WebEx, FaceTime).
 - An AX-API poll of browser tab URLs every 5s for Google Meet, web Zoom, web Teams.
 - An `idle reaper` that fires `didEnd` if the meeting app hasn't been frontmost for 120s — Zoom often lingers as a process after a call, and the terminate notification doesn't reliably fire.
- Sends one `didDetect` per session. Re-activations during an active session are suppressed by the coordinator (see 3.10).

3.2 Consent Coordinator

Owns the per-call `NSAlert` and the persistent consent registry under `~/Library/Application Support/Cadence/consent/<app>.json`. Six per-stream toggles — mic, camera, screen, remote audio tap, transcript storage, key-frame archive — plus a "Don't ask again" remember flag (default ON).

Once an app is granted with remember-ON, the daemon never prompts for it again. The dashboard's Settings panel exposes the saved decisions and lets the user flip them between auto-apply / ask-each-time, or forget them entirely.

3.3 Capture Coordinator

Routes consented streams through their respective capture services:

Stream	Backend	Whose data
Microphone	<code>AVAudioEngine</code> input tap (NSEException-wrapped via <code>CAExceptionCatcher</code>)	You
Camera	<code>AVCaptureSession</code> , 640x480 BGRA	You
Screen	<code>ScreenCaptureKit</code> , per-app filter	Meeting window (currently not analyzed)
Remote audio	macOS 14.4+ <code>CATapDescription</code> process tap	Other participants (routed to a separate channel; intentionally not blended)

Each source fails independently — if Screen Recording TCC is denied, the others still capture.

Remote-audio policy (the "other participants" channel). The process tap is the only path through which other people's voices enter the system. By default it is **used and discarded**: each buffer is run through an in-memory energy VAD to produce two aggregate signals — voiced-seconds (for the talk-time split) and cross-stream onset timestamps (for `turnTakingLatencyMs`) — and then dropped. The audio is never transcribed, never written to disk, and never sent off-device. The single way to make Cadence persist this stream is the `AdminSettings.persistRemoteAudio` opt-in, which is **double-gated**: it only takes effect when the operator has also granted per-call `remoteAudioTap` consent. When the opt-in is on, the daemon writes `sessions/{id}/remote-audio.wav` as a *separate* file from the user's own mic so the two streams are never accidentally blended at the listener.

3.4 Perception Pipeline

The heart of the system. An actor that ingests buffers and emits `FeatureWindow` rows every N seconds (default 30). Internally:

- **AcousticFeatureExtractor** — real DSP using `Accelerate` /vDSP. YIN pitch detection (with parabolic refinement), RMS energy, voice activity gating, jitter (period-perturbation), shimmer (amplitude-perturbation), HNR, pause distribution.
- **SpeechAnalyzerTranscription** — opt-in (`CADENCE_ENABLE_SPEECH=1`). Wraps the macOS 26 `SpeechAnalyzer` + `SpeechTranscriber` modules. Checks `AssetInventory.status` before starting; falls back to `NullTranscription` if the language model isn't installed. Words are tagged `speakerTag: "user"` since only the local mic is transcribed (no remote-stream diarization).
- **Transcription fallback (reprocess / dry-run / bulk paths)** — `SessionReprocessor.transcriptionFallback` calls the Eleplan `/audio/transcribe` Lambda when the live `SpeechAnalyzer` transcript wasn't captured (every imported recording, every re-analysis). Built by `TranscriptionGateway.makeFallback()` — the no-arg form resolves an `account_id` at *call* time: prefers the user's partner-account binding (`PartnerAccountSettings.current`) when set, falls back to the caller's own `cognito_id` (which doubles as their personal `account_id` in Eleplan's data model, `users.cognito_id == accounts.account_id` for personal rows). Without this fallback the `LinguisticAgent` would run against `wordCount = 0` feature windows on every import for users who haven't been paired to a partner customer — historically that surfaced as a "grey" Linguistic specialist with no rendered linguistic indicators. Same construction shape on the `VisualNarrationGateway` (`/vision/describe`) — both gateways now work for anyone signed in to Eleplan, with or without a partner binding.
- **GazeAndFaceService** — two-stage perception. Apple's `VNDetectFaceLandmarksRequest` constellation76Points is the front end: it detects the face, computes yaw/pitch/roll head pose, surfaces pupil landmarks for the eye-tracking metrics, and produces the face-bounding-box the next stage crops against. From there the pipeline forks:
 - **BlendshapeClassifier** — bundled Core ML pipeline (`Sources/CadenceCore/Resources/MediaPipeFaceMeshV2.mlpackage` + `MediaPipeBlendshapesV2.mlpackage`). Crops the face region to 256×256 RGB, runs the **478-point FaceMesh V2** model, indexes 146 landmarks into the **52-coefficient blendshape MLP-Mixer** (ARKit vocabulary: `browInnerUp` , `cheekSquintLeft/Right` , `mouthSmileLeft/Right` , `jawOpen` , ...), and maps onto the **FACS-13 core AU set** (AU1, 2, 4, 5, 6, 9, 10, 12, 15, 17, 20, 25, 26). Both models ported from Google MediaPipe (Apache-2.0); the blendshape head is a PyTorch port from py-feat (MIT) converted to Core ML via the scripts in `scripts/convert-blendshapes-to-coreml.py` and `scripts/convert-facemesh-to-coreml.py` . On-device, ~10 ms per frame on M-series silicon — pixels never leave the Mac. **Sampling rate is path-dependent**: live capture runs the classifier on every Vision observation (the camera delivers ~30 fps at VGA, so the pipeline keeps up in real time). The dry-run / bulk-import reprocessor runs Vision face landmarks at **10 Hz** (gaze / blink / saccade fidelity) but throttles the BlendshapeClassifier itself to **2 Hz** (`SessionReprocessor.processVideo` 's `auComputeStride: 0.5`) — AUs aggregate into 30-second feature windows, so ~60 AU samples per window stays well above what the synthesizer reads. The landmark-geometry heuristic in `FaceHeuristicAUs` stays as a safety-net fallback when the bundled `.mlpackage` fails to load or per-frame inference errors.
 - **Vision-derived metrics** — using the same 76-point constellation: gaze-toward-camera fraction, head yaw/pitch SD, head-movement RMS, expression entropy (Shannon over the AU-activation distribution).
 - **Eye-tracking metrics** — pupil-midpoint trajectories at each frame: **mean fixation duration (ms)** via I-DT dispersion-threshold (4 % face-normalized dispersion, 100 ms minimum), **saccade rate per second** via velocity threshold (1.5 normalized units/s), **gaze entropy (bits)** via Shannon over a 4×4 grid, **blink rate per minute** via runs of frames where eye openness drops below 30 % of the within-window 80th-percentile baseline, and **mean eye openness**.

- **LinguisticAnalyzer** — moving-average TTR, trigram repetition rate, disfluency markers, mean length of utterance.
- **Remote-stream VAD** — `PerceptionPipeline.handle(remoteBuffer:)` no longer drops the remote process-tap stream. It runs a cheap RMS-energy VAD and accumulates voiced seconds for the "other side" talk-time totals. No within-remote diarization (would need a multi-speaker model), so this is "everyone the user could hear" lumped together.
- **FrameArchiver** — opt-in. Saves a JPEG every 10s to `~/Library/Application Support/Cadence/sessions/<id>/frames/`. ~3 MB per minute at the default cadence. Beyond the periodic save, six **anomaly triggers** fire `archiveAnomaly` with the most recently cached camera frame whenever the corresponding perception event lands. Each is independently toggleable in Admin → Perception → Anomaly captures (all default ON); all share a global `anomalyMinIntervalSeconds` throttle (default 10s) so correlated detections in the same beat don't pile up six near-identical stills:
 - **Long-pause end** — `AcousticFeatureExtractor.onLongPauseEnd`. Pause $\geq$ `anomalyPauseThresholdSeconds` (default 1.5s); fires on the silence→voiced transition that closes the pause. Captures the moment of speech resumption after a thinking pause.
 - **F0 spike** — `AcousticFeatureExtractor.onF0Spike`. Voiced F0 sample exceeds rolling-window mean by `anomalyF0SpikeSigma` $\times$ SD over the last `anomalyF0WindowSamples` voiced frames (default 2.0 σ , 50-sample buffer). SD is clamped to a 10 Hz floor so stable speech doesn't fire on trivial natural variation. Captures emotionally salient pitch jumps — surprise, frustration, emphasis.
 - **Head burst** — `GazeAndFaceService.onHeadBurst`. L2-norm of yaw/pitch/roll deltas between consecutive Vision observations $\geq$ `anomalyHeadBurstRadians` (default 0.5 rad $\approx$ 28°). Frames with nil pose components are skipped so a momentary detection dropout doesn't poison the baseline. Captures rapid head movement — emphatic gestures, startle.
 - **Gaze-away end** — `GazeAndFaceService.onGazeAwayEnd`. State machine on `lookingAtCamera`: fires on the false→true transition only when the preceding false-run lasted $\geq$ `anomalyGazeAwaySeconds` (default 2.0s). Mirrors the long-pause-end pattern in the gaze channel — captures the moment of re-engagement after a sustained gaze-off.
 - **Blink burst** — `GazeAndFaceService.onBlinkBurst`. Sliding-window count of blink onsets (open→closed transitions on `eyeOpenness < blinkClosureRatio  $\times$  per-session-max-openness`); fires when $\geq$ `anomalyBlinkBurstCount` blinks occur within `anomalyBlinkBurstWindowSeconds` (defaults: 5 blinks / 3 seconds). Anchoring closure to the per-session running max insulates the detector from user-to-user differences in eye geometry. Captures fatigue / dry-eye / cognitive-load spikes.
 - **Talk-time flip** — `PerceptionPipeline` (in `flush`). Compares this 30-second window's user-share-of-voiced-time against the prior window's; fires when the two windows sit on opposite sides of 0.5 AND the magnitude of the shift is $\geq$ `anomalyTalkTimeFlipDelta` (default 0.2). Requires the remote process tap to be on; with one-sided audio the share is always 100% user and a flip can't happen. Captures conversational pivots — "the call just changed direction."

Adding a future anomaly type (mouth-open hold, sustained low-energy, sudden silence, etc.) is one closure on the relevant service + one case in `PerceptionPipeline.AnomalyKind` — the admin-flag check, throttle, and `archiveAnomaly` call are centralized in `recordAnomaly(kind:)`.

- **AudioFileWriter / VideoFileWriter** — admin opt-in (`AdminSettings.persistRawAudio` / `persistRawVideo`). When on, fan-out the same buffers to disk: 16-bit PCM `.wav` at the mic's native sample rate (~5.7 MB/min at 48 kHz mono) and H.264 `.mp4` @ ~2 Mbps via `AVAssetWriter` (screen capture preferred over camera as the source). Off by default since raw audio re-identifies the speaker.

Actor reentrancy is explicitly avoided: `lastFlush` is advanced *before* the `await store.append(window)` call so concurrent ingest paths can't double-flush the same offsets.

`sessionEnded` returns a `TalkTimeTotals` (`userSeconds` + optional `otherSeconds`) that `DaemonCoordinator` stamps onto the finalized `Session` row alongside the relative paths to any raw recordings.

3.5 Reasoning layer — the synthesis component

See section 4 for detail. The shipping per-session synthesizer is `OpenAIReasoner` against the cloud orchestrator Lambda. `FoundationModelsReasoner` survives in the Mac binary but is scoped to the on-device live intervention engine (§3.12); it does not run per-session synthesis.

`RuleBasedReasoner` runs alongside the cloud path: deterministic z-score-vs-bands indicators that always populate, even when the network is down or the daily quota's been spent. It also produces the `multimodalConvergence` indicator — the mean pairwise Pearson r of per-window acoustic × visual × facial stress signals (frontier multimodal-fusion metric).

The rule-based output is passed to the cloud synthesizer as a structured prior (see §4.7) so the model can anchor its reasoning on the deterministic z-scores rather than re-deriving from raw feature windows. When the cloud path is unavailable, the rule-based output is what the dashboard renders — the indicators are deterministic, the narrative is auto-generated, and the user sees a clear "reasoner unavailable" disclaimer.

Each rule-based indicator carries `bandZScore` + `bandCitationKey` on the `Indicator` record so the dashboard's `IndicatorCard` can render a "+1.3σ vs NT [citationKey]" chip next to the user-history percentile.

3.6 RAG Store

On-device retrieval over three corpora:

- **Research — 47 seeded peer-reviewed citations** in `Sources/CadenceCore/Resources/research_seed.json`, covering ASD (Fusaroli 2017 voice meta-analysis, Klin 2002 gaze, Parish-Morris 2016 pauses, Trevisan 2018 facial expression meta-analysis, Bone 2014 jitter/shimmer/HNR, Sasson 2008 fixations, Losh & Capps 2003 lexical diversity, Martin 2018 head movement, Tang 2024 NLP, Bottema-Beutel 2021 + Milton 2012 + Crompton 2020 + Cage 2018 neurodiversity-critique perspectives, de Marchena 2010, Diehl & Paul 2013, Heeman 2010, Lehnert-LeHouillier 2020), depression (Cohn 2009, Cummins 2015, Mundt 2012, Pampouchidou 2017, Sanchez 2013), schizophrenia (Blanchard & Cohen 2005, Elvevåg 2007, Hutton 1998, Parola 2020, Thompson & McCabe 2012, Varlet 2012), Parkinson's (Bowers 2006, Hlavnička 2017, Jin 2020, Rusz 2011), dementia/Alzheimer's (Eyigöz 2020, Fraser 2016, Pistono 2016), other clinical (Faurholt-Jepsen 2016 bipolar, Prkachin 2008 pain, Tseng 2013 ADHD), and methods (de Cheveigné 2002 YIN, Eyben 2016 GeMAPS, Bagher Zadeh 2018, Baltrušaitis 2018 OpenFace 2.0, Insel 2017 digital phenotyping, Koo & Li 2016 ICC, Low 2020 review, Soleymani 2017 multimodal sentiment, Stegmann 2020 repeatability). The library is **indicator-tag-indexed** at ingest time — every chunk is stamped with the `IndicatorKind` s it supports, so retrieval at reasoning time is O(1) per firing indicator family rather than a generic semantic search. Admins can extend the library; OpenAlex resolves DOIs to abstracts for live additions. The seed JSON also carries a per-citation `category` field — 8 named folders (Autism Spectrum, Methodology & Measurement, Schizophrenia, Depression & Mood, Parkinson's & Movement, Neurodiversity Perspectives, Dementia & Cognitive Decline, Pain & Other Clinical) — so the `LibraryView` sidebar lands every seeded paper in a meaningful folder on first launch instead of one "Uncategorized" bucket. Existing installs auto-stamp via `DocumentLibrary.backfillCategories()` at boot (idempotent, joins on `citationKey`).
- **LibraryChunk store** — paragraph-sized chunks of each document, indexed by indicator kind for the `LiveInterventionEngine` and post-session retrieval (see §3.12).
- **Cohort** — anonymized session-embedding pool for nearest-neighbor lookup. Empty in v0.1; populated only with user opt-in.

Living-corpus layer (CAD-63). The bundled 47-study seed is the *prior*; the grounding layer continues at the OpenAlex scholarly graph (250M+ works, free, no auth). Two components close the loop:

- **IndicatorTopicMap** (`Sources/CadenceCore/Library/IndicatorTopicMap.swift`) — every `IndicatorKind` is grounded in 1–3 OpenAlex topic IDs, derived not by keyword guessing but by aggregating the bundled corpus's *actual* topic assignments (one batched DOI lookup over the 47 seed papers, ~\$0.0001). Each topic anchor is paired with the cohort-level disorder topics (T10106 Autism Spectrum Disorder Research, T10023 Schizophrenia, T10009 Dementia, T10085 Parkinson's, etc.) so "prosody in Parkinson's" composes cleanly. The per-indicator paper trail lives in `docs/INDICATOR_TOPIC_MAPPING.md` .

- `CitationTruther` (`Sources/CadenceCore/Reasoning/CitationTruther.swift`) — runs between merge and persist in `ReasoningInvoker` . Batches every cited DOI in the just-produced indicator set into a single OpenAlex `/works?filter=doi:...` query; for each resolved work it (a) drops the citation if `is_retracted: true` , (b) logs when the work's topics don't overlap the indicator's mapped topics (informational only in v1; future iteration may downweight), and (c) keeps unresolved citations as-is on the principle that bibliographic validity $\neq$ OpenAlex coverage. Best-effort with a 5 s timeout fallback — any OpenAlex failure (offline, rate-limited, slow) silently passes the raw citations through so persistence is never blocked.

Why this matters: the bundled library is a *snapshot* — frozen at app release. The CitationAgent inside the orchestrator (§4.6) checks that every claim is anchored to a citation in that snapshot. The Truther adds an *out-of-context* validation pass against the entire living scholarly graph: catches retractions that happened after release, surfaces non-grounded citations the in-context check couldn't see, and gives every persisted citation an OpenAlex provenance trail. The two are complementary — in-context grounding at reasoning time + out-of-context truthing at persist time.

3.7 Session Store

SQLite (via `SQLite.swift`) at `~/Library/Application Support/Cadence/cadence.sqlite` . Tables: `sessions` , `feature_windows` , `indicators` , `moments` , `key_frames` , `fingerprints` , `feedback` . All payloads are stored as JSON blobs in a `payload` column for forward-compatibility with schema evolution — see `ConsentGrants.init(from decoder:)` for the `decodeIfPresent` fallback that lets old rows decode after fields are added.

A `reapZombies(olderThan:)` method marks unterminated sessions ended on app launch, so daemon crashes don't leave ticking-duration ghosts in the dashboard.

3.8 Local HTTP API (port 18047)

Embedded `Network.framework` server bound with `acceptLocalOnly = true` + `requiredLocalEndpoint = 127.0.0.1` — unreachable off-host at the kernel socket layer. Optional bearer-token auth via `CADENCE_API_TOKEN` . Endpoints:

```
GET /v1/health
GET /v1/active
GET /v1/sessions
GET /v1/sessions/:id
GET /v1/sessions/:id/windows
POST /v1/sessions/:id/end
GET /v1/fingerprints
POST /v1/sessions/:id/fingerprint
POST /v1/external/meeting           (browser-bridge lifecycle events; bridge-
token authed)
```

Lets browser extensions, mobile companion apps, research tools, or scripts integrate with the daemon as a proper service. Replaces the file-marker IPC the dashboard used to rely on.

3.9 Fingerprints

A salted SHA-256 over a 12-dim aggregate feature vector per session. The 256-bit salt lives at `~/Library/Application Support/Cadence/.cohort-salt` (chmod 600, file protection complete, never transmitted). The vector itself is not exposed over the API — only the hash, dimension, and metadata. This is the *only* artifact a future cohort upload service would ever see; raw features, audio, video, and transcript stay on-device unconditionally.

3.10 Daemon Coordinator

The orchestrator. Subscribes to `MeetingDetector` events, opens consent through the `ConsentCoordinator` , starts the `CaptureCoordinator` + `PerceptionPipeline` , writes the live `ActiveSessionFile` marker for the dashboard's red banner, watches for `StopRequest` from either the dashboard's Stop button or `POST /v1/sessions/:id/end` , and on session end:

1. Stops capture.

2. Finalizes the perception pipeline (flushes the last window, computes session-level linguistic features).
3. Invokes the reasoning layer (4).
4. Generates a fingerprint via `FingerprintBuilder`.
5. Clears the active-session marker.

Reentrant detections during an active capture are early-returned with a log line.

3.11 Repeatability layer

The newest layer, motivated by Stegmann et al. (Digital Biomarkers 2020), which showed that the open-source acoustic and linguistic features Cadence ships have median ICCs in the 0.02–0.55 range — well below the Fleiss 0.75 threshold for clinical decision-making. The Liss/Berisha 2024 arc ("features vs. measures") follows directly: a metric isn't fit-for-purpose until its test-retest reliability has been measured and surfaced to the reader.

What we do:

- `Repeatability.iccConsistencyOneRater` — ICC(C,1), single-rater consistency, two-way mixed effects (Shrout & Fleiss 1979; McGraw & Wong 1996). Balanced-design only; explicit `nil` for unbalanced/degenerate input.
- `Repeatability.withinSubjectsCV` — pooled within-subjects coefficient of variation, per Stegmann 2020 §2.2. Unit-independent.
- `RepeatabilityReport` — a `metricKey → MetricRepeatability` map persisted as JSON. Bundled priors at `Sources/CadenceCore/Resources/repeatability_bundled.json` are literature-grounded (Stegmann medians for acoustic/linguistic, Klin / Martin / Trevisan ranges for the exploratory visual block). A per-user overlay at `~/Library/Application Support/Cadence/repeatability_overlay.json` is merged on top with overlay-wins semantics.
- `IndicatorKind.canonicalDriverKeys` + `RepeatabilityReport.verdict(for:)` — the worst-case driver dominates an indicator's verdict. If any underlying metric has ICC < 0.75, the `IndicatorCard` surfaces a colour-coded `low repeatability` / `moderate repeatability` chip with a tooltip citing the threshold. **No suppression.** The Liss/Berisha framing is "exploratory, not silenced" — the indicator is still shown, but the user knows the measurement substrate is contested.
- `cadence-repeatability` CLI — `run` synthesizes an N×K speakers × sessions corpus with seedable SplitMix64, computes per-metric stats, writes the overlay. `show` prints the merged bundled overlay table. `clear` removes the overlay. Sized as a math-validation harness today; lays the groundwork for a discovery harness over real multi-speaker recordings later.

What this layer does *not* do yet:

- It does not run analytical or clinical validation in the V3 sense (Goldsack et al. 2020). Verification (sensor fidelity) and clinical correlation against established scales (PHQ-9, UPDRS, ALSFRS-R) are still gaps — see §7.
- It does not yet feed a quality-tier signal into the reasoning prompt. A future iteration could pass each indicator's tier to the synthesizer so it can downweight or omit chips in the narrative.

3.12 Live intervention engine (on-device, citation-grounded)

`LiveInterventionEngine` (added in PR #125; lives at `Sources/CadenceCore/Reasoning/LiveInterventionEngine.swift`) is the patent's "citation-grounded reasoning" claim made *live*, not aspirational. The engine watches each rolling-synthesis tick (default every 60 s) and evaluates a four-stage gate before deciding whether to surface a mid-session observation card.

Convergence gate (the patent's central claim, enforced in code):

1. **z-threshold filter** — every `RuleBasedReasoner` indicator carrying a `bandZScore` above $\pm 1.5\sigma$ (admin-tunable via `LiveInterventionZThreshold`) is a candidate. The `multimodalConvergence` indicator itself is excluded — it's a derived signal, not a primary channel.

2. **Distinct-family count** — candidates are bucketed by signal family (`acoustic` / `visual` / `interaction` / `linguistic`). The engine requires ≥ 3 **distinct families** above threshold in the same window, defaulting via `liveInterventionMinConvergence = 3` . Three acoustic indicators alone is one signal triple-counted, not convergence.
3. **Cooldown** — `liveInterventionCooldownSeconds = 300` (5 minutes default) prevents the engine from whispering every tick during a sustained convergence stretch.
4. **Same-pattern suppression** — if the firing `IndicatorKind` set equals the prior fire's set, the engine suppresses the call. No repetition.

When all four gates pass, the engine constructs an on-device `FoundationModels` prompt with:

- The firing indicators with their z-scores + drivers
- The top-3 `LibraryExcerpt` s retrieved by indicator-tag overlap (O(1) per indicator family, not generic semantic search — see §3.6)
- The canonical `METRICS.md` definitions for each firing `IndicatorKind` (so the model knows the exact computation behind each score)
- The last 3 prior interventions in this session (non-repetition + narrative arc)
- The user's chosen sensitivity / tone bucket

The model returns a strictly-formatted JSON: `{ observation, suggestion?, citationKeys[] }` . The engine validates the citation keys against the loaded library — **the model can only cite from what's loaded**, and the JSON validator drops any cite the library can't resolve. The fired prompt is persisted to the `live_interventions` SQLite table with full structured metadata (indicator kinds, z-scores at fire time, citation keys + excerpts, sensitivity bucket, wall-clock timestamp, session offset) so the post-session synthesizer and the timeline scrubber can both render it.

Why this is the patent's central claim demonstrated:

- *Cross-modal temporal co-occurrence* — the convergence gate enforces it in code, not in narrative
- *Citation-grounded reasoning* — the model's context window literally cannot include sources outside the bundled library
- *Real-time perception executed locally* — `FoundationModels` is on-device; the engine never hits the network
- *Reasoning available without network connectivity* — the engine is the default; cloud reasoners run in addition, never instead

Defaults are conservative — the engine speaks rarely by design. Three sensitivity buckets are surfaced in Admin → Reasoning → Live intervention prompts: `rare` (4 channels, 600 s cooldown), `normal` (3 channels, 300 s — default), `chatty` (2 channels, 120 s — for demos and calibration).

11 unit tests cover the convergence detector, family routing, distinct-family enforcement, JSON parsing edge cases (empty observation = suppression, null suggestion, malformed response), and the `METRICS.md` context assembler. The `FoundationModels` round-trip itself requires macOS 26 + Apple Intelligence at runtime and isn't unit-testable, but every deterministic input the engine assembles around it is covered.

3.13 Cadence Bridge — independent of any specific video or LMS platform

`chrome-extension/` (in the repo) ships **Cadence Bridge** — the single piece of the system that needs to know about the host video / LMS platform, deliberately isolated so the rest of the architecture stays platform-agnostic.

The problem the Bridge solves. macOS's window-level meeting detection sees native apps directly — Zoom Desktop, Teams Desktop, FaceTime — and the `MeetingDetector` actor picks them up via `NSWorkspace` notifications and a curated bundle-ID set. What macOS cannot see is a Google Meet running in a Chrome tab, a Zoom Web session, or a `BigBlueButton` call living inside a Canvas iframe. Those calls aren't windowed apps; they're DOM regions inside a browser process. Cadence Bridge closes that gap.

How it works. The extension manifest (`chrome-extension/manifest.config.ts`) matches `<all_urls>` with `all_frames: true` . Two content scripts:

- `src/content/injected.ts` — runs in the page's MAIN world (`world: 'MAIN'`), hooks `getUserMedia` and `RTCPeerConnection` so it can observe every WebRTC negotiation that initiates a media stream. This is how it sees Google Meet, web Zoom, web Teams, web Webex, BigBlueButton, Whereby, Jitsi, and any iframed call inside Canvas / Blackboard / Brightspace / Moodle.
- `src/content/detect.ts` — runs in the default isolated world, listens for the `postMessage` events the injected script emits, attaches frame metadata (tabId, frameId, frame origin), and forwards them to the service worker.

The service worker maintains a Native Messaging connection to the local Cadence daemon over `127.0.0.1:18047` , gated by a bridge token (`BridgeAuth.loadOrProvisionToken`) the Cadence app generates on first launch and writes to the extension's storage via the per-LMS native-messaging registration manifest. The Bridge **never sees audio or video** — only the lifecycle events (`candidate` / `connected` / `closed`) plus the meeting URL. Media capture happens entirely in the Cadence Mac app via system-level AV permissions.

Why this is the "operates independent of any specific video or LMS platform" claim made concrete. The Bridge runs above the meeting layer, not inside it. There's no LTI 1.3 registration, no per-LMS admin approval, no per-school IT gauntlet. The same content script works inside a Canvas page embedding BigBlueButton, a Brightspace page embedding Zoom, a standalone Google Meet tab, or a Whereby room — same code path, no per-platform integration. This is the patent's Fork 1 (platform integration via API/extension) implementation that lets the same Cadence deployment serve every LMS partner Cadence ever signs without per-LMS engineering work.

State surfaced to the user. The extension popup (`chrome-extension/src/popup/Popup.tsx`) renders three derived states: `offline` (daemon not reachable), `idle` (daemon connected, no meeting), and `live` (daemon connected, meeting detected). The popup's visual design matches the marketing site mockup at cadence.eleplan.com/lms/ exactly — when prospects look at the marketing pitch and then install the extension, they see the same chrome.

3.14 Dry-run import & bulk series analysis

Cadence is built around live meeting capture, but a clinician's analyzable material is rarely captured live — it's a year of telehealth visits sitting on a hard drive. `DryRunImporter` and `BulkDryRunImporter` (in `Sources/CadenceCore/DryRun/`) let an operator drop one or many past audio/video files into Cadence and run them through the **identical** perception → reasoning pipeline that a live capture would use, producing indicators directly comparable to live sessions.

Single-file path (`DryRunImporter`). Reads an audio or video URL, decodes it via AVFoundation, feeds buffers through `PerceptionPipeline` exactly as the live `CaptureCoordinator` would, persists a `Session` with `endedAt` already set, then invokes `ReasoningInvoker` for the reasoning + indicator pass. The on-disk layout matches a live session: `~/Library/Application Support/Cadence/sessions/<id>/` (or, when an Eleplan project is bound, `~/Library/Application Support/Cadence/projects/<project_id>/sessions/<id>/` — see §3.15).

Bulk path (`BulkDryRunImporter`). Series analysis over N files. Default caps: `bulkMaxSessions = 25` , `bulkMaxTotalSeconds = 43_200` (12 h), `bulkMaxTotalBytes = 4 GiB` , `bulkMaxEstimatedCostUSD = $5.00` , all admin-tunable. The importer:

1. **Pre-flight gate.** Validates Cognito sign-in (cloud-reasoner installs only), file readability, and cumulative size/duration/cost vs. the admin caps. Blocking violations refuse the run for everyone. The estimated-cost panel is **gated behind admin unlock** — non-admin operators see no cost numbers at any point in the flow; admin operators see a structured per-file token estimate × model price preview before the run starts.
2. **Chronological ordering.** Reads each file's `AVAsset.commonMetadata` creation date when present; falls back to a natural-filename sort (`session-001.wav` , `session-002.wav`) when creation dates cluster within 5 s (common when files are batch-exported from a meeting platform that stamps them at copy time). The operator can drag-reorder before kicking off the run.
3. **Bounded concurrency.** A `TaskGroup` runs at most `bulkConcurrency` files in parallel (default 1). Each parallel slot runs the full single-file `DryRunImporter` pipeline. The concurrency cap exists because each session fans the server-side orchestrator out to 8 sub-agent OpenAI

Responses-API calls; uncapped $25 \times 8 = 200$ simultaneous calls trips the per-account quota and the per-model TPM ceiling.

4. **Per-file blame attribution.** When a session lands with 0 indicators but the feature windows show real acoustic signal, the importer reads `agent_findings` for that session and surfaces the right copy: `quotaExceeded` → "Daily reasoner quota reached", `notSignedIn` → "Reasoner couldn't authenticate", `apiError` → "Reasoner returned an HTTP error". The pre-2026-05-21 conflation of "no audio" with "reasoner offline" is gone.
5. **Cross-session synthesis.** Once every file lands (success or skipped-as-failed), `CrossSessionSynthesizer` produces a deterministic rollup over the whole series — indicator trajectories (mean / SD / monotonic-trend test per IndicatorKind), moment density per session, anomaly density, talk-time evolution — and (when enabled) a single LLM narrative pass over the rollup. See §4.8.

The output: a BatchAnalysis. A new row in the `batch_analyses` table carries the series ID, list of `sessionIds`, per-session summaries (filename, duration, indicator count, narrative excerpt, status, error), the deterministic rollup, and the optional LLM narrative. `BatchAnalysisDetailView` renders the clinical dashboard: top-line narrative, indicator-trajectory sparklines, reliability panel (ICC verdict per indicator), anomaly density per session, talk-time totals, and a per-session list that links into the individual `SessionDetailView` for the full read.

Why this is architecturally significant. Most behavioral-signal tools are either *live-only* (call-center analytics) or *recording-only* (research uploads). Cadence's pipeline is built so that the same `FeatureWindow` shape, the same `ReasoningContext`, the same agent orchestrator, and the same JSON output contract serve both modes. There's no "import mode" code path that diverges from the "live mode" code path. The Bulk importer is just a TaskGroup around the live pipeline.

3.15 Project federation & per-project storage

A clinician's Cadence install is rarely a single bucket of sessions. The same operator may be analyzing a family member's longitudinal trajectory, running a research study with N participants, or working across multiple subjects in a clinical practice. **Projects** are the partitioning unit.

Local layout. When a project is bound on this install, every captured (or imported) session writes its artifacts under `~/Library/Application Support/Cadence/projects/<project_id>/sessions/<session_id>/` instead of the flat `sessions/<session_id>/`. The `Session.projectID` field carries the binding into SQLite. `AppState.visibleSessions` filters the dashboard's lists by the currently-active `currentProjectID` from `UserDefaults`, so the sidebar / `SessionListView` / `HomeView` / `Trends` all show one project at a time. An "All sessions" sentinel disables the filter for cross-project review.

Account binding. Projects live on the operator's Eleplan account.

`CadenceSessionsClient.fetchProjects(accountID:cognitoID:)` pulls `/projects/get-all`; the resulting `[ProjectSummary]` is cached on `AppState.availableProjects` and refreshed alongside `sessions` on every 3 s dashboard poll. The header-bar `ProjectPickerMenu` (compact pill on `HomeView`, full row in Settings) flips the active project; flipping triggers a re-read of `visibleSessions`. Numeric and string project IDs are both accepted on decode for forward-compatibility with the partner-side API's tolerant typing.

Sync (CAD-49). `SyncCoordinator` runs a durable upload queue that flushes any ended-but-not-yet-uploaded session to `<stage>-api.eleplan.org/cadence/sessions` every 30 s when the operator has opted in to auto-sync. The upload payload is the structured feature-stream + indicators + moments — never raw audio, video, or transcript. Sessions stay queryable on the operator's Eleplan account so a partner clinician or family caregiver can read the dashboard without holding the operator's local install.

Why federation, not a cloud DB. Cadence's local SQLite is authoritative; the cloud copy is a derived projection. A failed sync never blocks capture; a deleted cloud row never invalidates the local one. The split keeps "Cadence works offline" honest while letting partner-account roles see the same analysis layer.

3.17 Environmental signal layer (opt-in)

Behavioral indicators don't sit in a vacuum. A 4σ drop in voiced fraction means one thing on a winter morning at 06:00 in a room measuring 20 lux and another at noon on a clear day in a room at 600 lux. To let the synthesizer correlate (when warranted) without forcing every install to pay the permission cost, Cadence ships an opt-in **environmental signal layer** alongside the perception pipeline.

The layer adds an `EnvironmentalFeatures` substruct to every `FeatureWindow`. Six fields, all optional:

Field	Source	Cadence	Notes
<code>outdoorTempC</code> , <code>outdoorHumidity</code> , <code>outdoorUVIndex</code>	WeatherKit at the device's CoreLocation coordinate	Once per session at <code>sessionStarted</code> (cached for the rest of the session)	Requires the <code>com.apple.developer.weatherkit</code> entitlement to be registered for the app's bundle ID on Apple's developer portal AND a granted <code>NSLocationWhenInUseUsageDescription</code> prompt. macOS 13+.
<code>indoorLux</code>	IOKit <code>AppleLMUController</code> built-in ambient light sensor	Per <code>FeatureWindow</code> flush (every 30 s default)	Available on most MacBooks; nil on hosts without the ALS (Mac mini, Mac Studio, most Mac Pros). No permission gate — IOKit lookup only. Calibrated against a hand-held lux meter at 5 / 300 / 10 000 lx; linear in the middle, undershoots at the extremes.
<code>audioNoiseFloorRMS</code>	<code>AcousticFeatureExtractor</code> silent-period RMS (frames where the VAD reads below <code>acousticEnergyThreshold</code>)	Per <code>FeatureWindow</code> flush	Mirror of <code>AcousticFeatures.audioNoiseFloorRMS</code> . Derived from data the pipeline already processes — zero additional permission surface.
<code>sceneLuminance</code>	<code>GazeAndFaceService</code> per-frame Y-channel mean	Per <code>FeatureWindow</code> flush	Mirror of <code>VisualFeatures.sceneLuminance</code> . Reads the YCbCr Y plane (or BGRA → BT.601 luma) at a 32-pixel stride; ~microseconds per frame. Tracks the room's brightness independent of face-detection success.

Master switch: `AdminSettings.environmentalSignalsEnabled` — default OFF. When OFF the `EnvironmentSensor` actor is a no-op: no WeatherKit call, no CoreLocation prompt, no ALS read, no env substruct attached. The audio noise floor and scene luminance fields on the existing `AcousticFeatures` / `VisualFeatures` records continue to populate either way (they're derived from data the pipeline already processes), they're just not mirrored into the env substruct unless the flag is on.

Failure modes are nil, never throws. Each sub-signal independently can be nil:

- WeatherKit unentitled / offline / location denied → outdoor fields stay nil.
- Host has no built-in ALS → `indoorLux` stays nil.
- Window contained no silent acoustic frames → `audioNoiseFloorRMS` stays nil (rare on conversational audio).
- No camera frames processed → `sceneLuminance` stays nil.

If every sub-signal is nil, the pipeline skips attaching the substruct entirely — synthesis never sees an empty `{}`. Operators can flip the flag on without worrying about a partial setup breaking sessions; the worst case is "less ambient context than the operator hoped for," not "session capture failed."

What the synthesizer does with this. When the env substruct is present, the system prompt names the fields so the model can correlate them with the indicators it's already surfacing — e.g. "the user's blink rate is 3σ above baseline AND `indoorLux` is below 50 (dim room): flag as possibly fatigue-related ambient context rather than baseline drift." The model is told explicitly that these are correlational context, never proximate causes, and never to substitute environmental

explanations for behavioral observations the family deserves to discuss with their care team. The hard rules in §4.3 stay intact.

What this layer is not. Not a clinical environmental-monitoring system. Not a calibrated photometer or weather station. Not a longitudinal climate-correlation engine — the substruct sits next to one session's windows, nothing more. The point is to give synthesis enough context to *not over-call* an indicator that has an ambient explanation, never to add a new diagnostic axis.

3.16 LiveEventBus — in-process SSE for the live dashboard

`LiveEventBus` (`Sources/CadenceCore/Live/LiveEventBus.swift`) is the actor-based multicast bus the daemon publishes to and the HTTP API's `/v1/stream` endpoint subscribes to.

Event kinds (see `LiveEvent.Kind`): `daemon.ready`, `meeting.detected`, `consent.resolved`, `capture.started`, `window.partial` (default every 500 ms — drives the live strip charts), `window.emitted` (one full 30 s `FeatureWindow` landed), `reasoner.fired`, `agent.fired`, `indicator surfaced`, `moment surfaced`, `intervention surfaced`, `session.ended`, `heartbeat` (every 15 s when otherwise idle so SSE proxies don't time out).

Backpressure model. Each subscriber gets its own bounded `AsyncStream` (`bufferingPolicy: .bufferingNewest(512)`) so a slow client can't stall the daemon. If a subscriber falls more than 512 events behind, the oldest are dropped and the next event delivered carries a non-contiguous `id`, which the SSE client uses to render a "missed events" indicator and decide whether to refetch from SQLite.

Replay buffer. A separate 512-event ring buffer holds recent events for reconnect replay. When a subscriber reconnects with `Last-Event-ID: N`, the bus replays every buffered event with `id > N` before joining the live fanout. Sized for ~2.5 minutes at a busy 500 ms partial cadence so most short network blips replay cleanly. Heartbeats are not buffered (they'd dilute the buffer during real-event bursts).

Why this is its own component. The dashboard polls SQLite every 3 s for the slow-path state, but the live experience (strip charts ticking at 2 Hz, indicator pills lighting up the moment the orchestrator fires, intervention cards appearing inline mid-session) requires a sub-second push channel. SSE on the local HTTP API is the cleanest solution — no extra ports, no WebSocket framing, works through every Mac firewall posture out of the box, and the Cadence Mac app, the Cadence Bridge extension, and any third-party research tool integrate the same way.

4. The synthesis component (reasoning layer)

This is where the foundation model is invoked. The shipping path is a **single cloud orchestrator** running the multi-agent pipeline server-side; the Mac client speaks to it through one `OpenAIReasoner` against the `cadence-messages-reasoner` Function URL. The lambda fans the structured feature stream out to seven domain-scoped specialists in parallel, then closes with a Synthesizer (eight agents total). See §4.6 and Appendix B for the per-specialist scope; §4.1–4.5 cover the input / output contract that's identical regardless of how many specialists fired.

The pipeline doesn't use tools or self-iterate. It produces a single `[Indicator] + [Moment] + narrative + anomaly_narratives[]` shape the dashboard consumes.

A second on-device path stays in the Mac client: `FoundationModelsReasoner` powers the live intervention engine (§3.12), producing short structured mid-session observation cards locally. It does not run per-session synthesis — the orchestrator's prompt exceeds Apple Intelligence's 4096-token context.

4.1 Inputs (`ReasoningContext`)

- The `Session` record (start/end, app, participant count).
- The full `[FeatureWindow]` series (truncated to 120 windows in the prompt for cost control).
- `NormativeBands` — published mean ± SD per metric for neurotypical and ASD cohorts, sourced from the seeded meta-analyses.
- `CohortNeighbor`s — top-k nearest matches in the (opt-in) cohort embedding pool.

- `ResearchCitation`s — top-k retrieved from the RAG store by indicator kind.
- A short `userHistorySummary` ("N prior sessions in baseline").

4.2 Models

One per-session synthesis backend in the shipping client; one on-device backend for live interventions.

- `OpenAIReasoner` — the per-session synthesizer. Posts the structured feature stream + retrieved research + repeatability hints + rule-based prior to the eleplan-backend `cadence-messages-reasoner` Function URL. The lambda runs the eight-agent orchestrator server-side (see §4.6) — admin-tunable primary + per-specialist model overrides (`gpt-5` , `gpt-4.1-mini` , etc.) via the `model_overrides` block in the request payload. Streaming SSE response — `agent_started` + `agent_complete` events fire as each specialist's call dispatches and resolves, the Mac persists `AgentFinding` rows in arrival order so the dashboard's intelligence bar lights up within ~5–10s of orchestration start. Authentication: `Authorization: Bearer <Cognito ID token>` + `X-Cadence-Device: <UUID>` (per-install) + `X-Cadence-Body-SHA256` (tampering bind), resolved via `CloudReasonerAuth.shared.proxyHeaders()`. Daily-quota check at `accounts.cadence_reasoner_daily_quota` with platform default. Only the structured feature stream + retrieved research is sent — never audio, video, or transcript content (transcript text itself is opt-in per-session). The entire path is covered by Eleplan's signed HIPAA BAA with AWS + OpenAI's signed BAA with zero-data-retention in force. Users never bring their own API key. An App-Attest auth branch remains in the lambda for any in-flight Mac-App-Store fixture; the JWT path is what every Developer-ID install uses today.
- `FoundationModelsReasoner` — on-device Apple Intelligence (`SystemLanguageModel.default` → `LanguageModelSession`). Free, ~3B parameters, no data leaves the Mac. Drives the live intervention engine (§3.12) only — its 4096-token context cap can't fit the per-session synthesizer prompt.

The earlier `ClaudeReasoner` against the Anthropic Messages API was retired when the orchestrator moved off Anthropic's Messages API onto the OpenAI Responses API with strict-JSON structured outputs; the Mac-side switch happened at the same time the server-side fan-out went multi-agent.

Both reasoner implementations parse the model's JSON output through the same contract.

4.3 Hard constraints (from the system prompt)

- **Never output an autism probability.** The framing is always "this session resembles X% of cohort sessions reported during Y" — never "you have X". This is the SaMD-regulation line.
- **Always cite.** Every indicator must reference one or more citation keys from the retrieved research set. Anything else is marked exploratory.
- **Surface uncertainty.** Short or noisy windows must say so. Confidence intervals are required on every score. The system also computes a session-level **input quality score** (0..1, with a tier label) that aggregates duration, voiced fraction, face coverage, transcript word count, and HNR into one numeric handle (`CadenceCore.SessionQualityScore.compute(...)`). Rendered as a percentage on the session quality header; closes provisional patent §8 "input quality scores."
- **Flag — don't interpret.** Patterns that look clinically relevant get flagged for clinician review; the dashboard surfaces them but does not directly explain them to the family.
- **Recommend follow-ups, never diagnose.** Each indicator may carry 0–2 `recommendations` — structured non-diagnostic follow-up actions in one of four kinds: `repeatAssessment` (with a timeframe), `offerResource` (with a topic key), `suggestAccommodation` (an environmental/behavioral/capture-setup adjustment), or `flagForReview` (worth a clinician's eye). The honest default is empty — the prompt instructs the model NOT to fabricate recommendations to fill the field. Maps to provisional patent §7 "recommendations for follow-up actions (e.g., repeat assessment, offer resources, suggest accommodation)."

4.4 Output contract

JSON, parsed back into model types in `ReasoningOutput` :

```

{
  "narrative": "one paragraph on what stood out this session",
  "indicators": [
    { "kind": "prosodicVariability",
      "score": 0.42, "percentile": 0.18,
      "confidence_low": 0.34, "confidence_high": 0.51,
      "drivers": ["f0_std_dev_semitones"],
      "citation_keys": ["fusaroli2017"],
      "reasoning": "one sentence",
      "recommendations": [
        { "kind": "repeatAssessment",
          "text": "Pitch variability sits on the edge of the published band; a
            second session in this same setting would clarify whether it's stable.",
          "timeframe": "in 4-6 weeks" }
      ]
    }
  ],
  "moments": [
    { "start_offset": 120, "end_offset": 150,
      "indicators": ["prosodicVariability", "gazeTowardCamera"],
      "summary": "joint drop in F0 variability and gaze toward camera",
      "flag_for_clinician": false }
  ],
  "questions_for_family": ["..."]
}

```

The **Indicator** and **Moment** rows land in SQLite; the dashboard renders them with the AI's **plain-English interpretation leading the indicator card** (a sparkle-prefixed reasoning sentence at **.body** weight, the math demoted to a compact strip underneath holding the bell-curve gauge + score + 95% CI + z-vs-band label — when an interpretation is present; cards with no usable synthesizer reasoning fall back to the legacy score-primary layout). Each anomaly key-frame surfaces its AI interpretation alongside the deterministic math caption in the events-ribbon popover, the keyframe gallery detail sheet, and the hover tooltip on every thumbnail strip — meaning sits next to math, not commentary alongside it. Citation cards, percentile pills, drivers as feature-name badges, and (for moments) a horizontal scroll of key-frame stills round out the read.

The same persisted rows back two render modes, picked by `AppState.viewMode` (UserDefaults-backed; toolbar toggle on `SessionDetailView`):

- **Basic View** — engagement-ring headline card + plain-language timeline (≤7 chips collapsing moments, anomaly key-frames, and live AI interventions) + up to 3 highlight callouts (positive / watch / notable). Translation runs through `CadenceCore.PlainLanguage` — pure functions over the existing `Indicator` / `Moment` / `KeyFrame` / `LiveInterventionPrompt` models with one dictionary entry per `IndicatorKind` and driver. No new collection. Built for the smart-but-not-researcher reader (parent, caregiver, generalist clinician). New installs default to this mode.
- **Advanced View** — the full clinical dashboard: severity banner, synthesizer headline, window-anomaly heatmap, AI-interpretation-led indicator cards (with compact metric strip carrying CI + band z-score + driver readout, longitudinal sparkline, repeatability tier), per-window summary cards, unified event ribbon (anomalies + interventions + moments on one axis), a multi-strip `TimelineView` (F0 variability, pitch contour with ±range envelope, RMS energy, voiced fraction, gaze→camera, head movement with yaw + pitch + RMS overlay, expression entropy, blink rate, eye openness, turn-taking latency, speaking rate, WPM), talk-time split, moments with severity badges, `GroundedResearchPanel`, `LiveInterventionTimelinePanel`, `AgentFindingsPanel`, `VoiceQualityPanel`, `GazeQualityPanel` (fixation duration + saccade rate + gaze entropy), `PauseDistributionPanel`, `AnomalyHistogramPanel` (per-minute frequency stacked by detector), `KeyFrameGallery` (clickable stills, filterable by trigger), `AUHeatmap` with click-to-drill per-AU trajectory + FACS-grounded interpretation, transcript pane, recording playback (AVKit via `NSViewRepresentable` `AVPlayerView`), CSV export, **right-rail** `ChatPanel` with an "agents intelligence bar" of clickable per-specialist chips that pop a popover containing the specialist's summary + three canned drill prompts that send directly to chat — making the chat the unified interrogation surface for "what did the AcousticAgent see?" follow-ups.

Toggleing between Basic and Advanced is instantaneous — same data, different surface; no reload, no recompute.

4.5 What the synthesis component *cannot* do today

- It does not see the camera frames. It reasons over feature numbers only. The dashboard renders stills next to moments so the user can verify visually, but the model itself is text-only.
- It does not learn between sessions in any model-weight sense. The RAG store grows; the model weights don't. "Learning" decomposes into: corpus growth (research + cohort), expanded in-context examples, and periodic supervised fine-tuning that would require labeled outcomes — not yet implemented.
- **Remote-stream coverage is shallow.** The process tap now contributes voiced-seconds (talk-time split) and the trailing window's `wordTimings` are speaker-tagged, but the remote stream is not transcribed and there is no within-remote diarization. "Other" is everyone the user could hear lumped together.

4.6 Multi-agent orchestrator (server-side)

The orchestrator runs **server-side** in the `cadence-messages-reasoner` Lambda (backend path: `backend/api/client/services/openai/orchestrators/cadence/`). The Mac client sees it through one `OpenAIReasoner.reasonWithFindings` call that streams `agent_started` + `agent_complete` SSE frames as each specialist resolves. There's no client-side orchestration code anymore — the legacy `AgentOrchestrator` / `AgentRegistry` Swift files were removed when the fan-out moved off the Mac; `AdminSettings.useMultiAgentOrchestrator` survives in the struct for on-disk-decode back-compat but no longer changes behavior.

Why this exists. Stuffing every modality into one prompt forces the model to context-switch between acoustic, ocular, linguistic, and temporal reasoning, and to triage which of 47 citations applies to which feature. The multi-agent path does the triage at *prompt-construction time* instead: each specialist sees only the features it's qualified to interpret and only the citations relevant to its modality. The result is sharper per-modality findings + a Synthesizer pass whose job is exactly the cross-modal fusion the patent calls out as the system's frontier contribution.

The seven specialists + one synthesizer (defined in `backend/api/client/services/openai/orchestrators/cadence/specialists.js`; structured-output schemas in `schemas.js`; dispatch + SSE in `handlers.js`):

Agent	Scope	Citation anchors
AcousticAgent	F0, jitter, shimmer, HNR, pause distribution, speaking rate	Fusaroli 2017, Diehl & Paul 2013, Parish-Morris 2016, Rusz 2011 (Parkinson voice), Tang 2024
GazeAgent	gaze-toward-camera, fixation duration, saccade rate, gaze entropy, blink rate, head pose, FACS-13 AUs	Klin 2002, Sasson 2008, Martin 2018, Trevisan 2018 — explicitly paired with the Milton 2012 / Crompton 2020 double-empathy critique to keep the framing non-deficit
LinguisticAgent	MATTR, repetition score, disfluency count, MLU, per-word timings	Heeman 2010, Losh & Capps 2003, de Marchena 2010
TemporalAgent	turn-taking latency, talk-time balance, voiced/silent transitions across the full session	Wynn 2020, Heeman 2010
BaselineAgent	the user's own calibration baseline + cross-session trajectory (no research set; comparison is to <i>self</i>)	—
CitationAgent	reviews the other agents' findings and verifies each claim ties back to a citation in the bundled corpus (in-context grounding at reasoning time; complemented by CitationTruther 's out-of-context OpenAlex validation at persist time — see §3.6)	full research set
ClinicalFramingAgent	rewrites findings into non-diagnostic language; refuses any framing that resembles a diagnosis	—

The **Synthesizer** receives all seven sub-agent outputs plus the full feature stream (up to 60 windows) plus the research corpus and produces the final [\[Indicator\]](#) + [\[Moment\]](#) + [narrative](#) + [anomaly_narratives\[\]](#) set. Its system prompt (managed as an editable server-side rule in [system_settings.rules.cadence_synthesizer_guidance](#) , mirrored as the [DEFAULT_SYNTHESIZER_GUIDANCE](#) constant in [prompts.js](#)) carries an explicit instruction:

Surface [multimodalConvergence](#) when it's real. This is the patent's frontier framing.

That line is the load-bearing connection between code and patent. The Synthesizer is the only place in the pipeline whose job is *exclusively* cross-modal fusion — measuring whether the acoustic, ocular, and linguistic signals point in the same direction, and how much weight a clinician should put on the combined read versus any single channel. The Synthesizer also receives an [ANOMALY_KEY_FRAMES](#) block (id + offset + detector kind + math caption per anomaly the perception pipeline flagged) and emits one short plain-English interpretation per anomaly into [anomaly_narratives\[\]](#) ; the Mac persists each onto the matching [KeyFrame.narrative](#) so the dashboard renders meaning inline alongside the math caption.

What you get for the multi-agent fan-out:

- Citation-grounded findings per modality (each specialist's narrow citation list keeps the model from hallucinating cross-domain attributions).
- The double-empathy critique surfaces consistently on every gaze-related finding instead of "if the model remembers to apply it."
- A separate framing-check pass that catches diagnostic language the synthesis would otherwise need to police itself for.
- Explicit, traceable late fusion: the Synthesizer's reasoning string shows which modalities converged and which dissented, which is exactly the explainability artifact the patent's §8 ("Confidence / Uncertainty / Explainability") and Fork 8 ("Human-in-the-loop governance") describe.

The lambda runs wave-parallel: specialists whose prompts can run independently (Acoustic / Gaze / Linguistic / Temporal / Baseline) fire concurrently as Wave 1, then CitationAgent and ClinicalFramingAgent run on the merged findings as Wave 2, then Synthesizer closes. The `OpenAIReasoner` streaming path receives one `agent_started` SSE frame at each specialist's dispatch (so the chat panel's intelligence bar lights up immediately) and one `agent_complete` frame per resolution; total wall-clock is dominated by the slowest specialist, not the sum.

4.7 The deterministic rule-based prior

Cadence's reasoning layer is **never** "LLM-only." `RuleBasedReasoner` runs unconditionally on every session — it's the deterministic floor that keeps the dashboard producing indicators when no foundation model is configured, when the cloud reasoner times out, or when the on-device path isn't available. It computes z-scores against the normative bands and the `multimodalConvergence` Pearson r across modalities. When an LLM IS configured, the rule-based pass still runs — and its output is now passed to the model as a **structured prior** alongside the raw feature stream, not just merged in afterward.

The prior block in the user prompt looks like this:

```
RULE-BASED PRIOR (deterministic z-scores against normative bands — anchor your read here):
- gazeTowardCamera: score 0.62, z=-1.34σ vs NT band [klin2002] (drivers: gaze_toward_camera_fraction)
- multimodalConvergence: score 0.41, z=+0.23σ vs NT band [schuller2018]
- prosodicVariability: score 0.55, z=+1.12σ vs NT band [fusaroli2017] (drivers: f0_std_dev, f0_range)
...
```

The system prompt instructs the model to use the prior as an anchor: enrich it with narrative + moments + indicators the rule pass couldn't produce; do NOT silently disagree with the deterministic numbers; if you DO disagree on a non-deterministic indicator, justify the disagreement from the feature stream in the `reasoning` field. `multimodalConvergence` is treated as fact — its value is a Pearson r, not an opinion.

Why this matters:

1. **Hallucination resistance.** The model can't claim "your speaking rate is unusual" when the prior shows $z = +0.1\sigma$.
2. **Cheaper synthesis.** The model doesn't re-derive z-scores from raw FeatureWindows; it gets them precomputed and spends its tokens on narrative + cross-modal interpretation.
3. **Explicit cross-check.** When the LLM and rule pass disagree, the disagreement surfaces in the indicator's `reasoning` field — an explainability artifact (patent §8 / Fork 8) rather than a hidden divergence.
4. **Honest fallback semantics.** Turning the LLM off doesn't kill the dashboard; it just falls back to what the prior would have anchored the LLM to anyway.

The merge logic in `mergeWithRuleBased` still runs after the LLM call — it fills any `IndicatorKind` the LLM didn't produce with the rule-based version. So the prior shows up twice in the flow: once in the prompt (so the LLM sees it before answering) and once after (so any gap in the LLM's output is filled deterministically).

4.8 Cross-session synthesis

When the bulk path (§3.14) finishes per-file reasoning across N sessions, a separate component closes the loop: `CrossSessionSynthesizer` (`Sources/CadenceCore/DryRun/CrossSessionSynthesizer.swift`). Its job is to answer the question the per-session indicators can't — *what's true about this person across the whole series?*

Two-pass design.

1. **Deterministic rollups (always run).** For every `IndicatorKind` that fired on any session in the batch, compute mean, SD, monotonic trend (Mann-Kendall test), session-count-fired-on, and a banded reliability verdict via the repeatability layer (§3.11). Anomaly density per session. Talk-time totals (user voiced seconds, other voiced seconds, ratio). Moments-per-session histogram.

Outlier flagging (sessions whose indicator vector sits $>2\sigma$ from the per-batch centroid in cosine distance). All of this is pure-numeric and runs without an LLM in the loop. The output is the `BatchAnalysis.Rollups` struct.

2. LLM narrative pass (`bulkCrossSessionNarrativeEnabled` , default ON). One additional reasoner call — admins can flip it off for cost-sensitive runs, in which case the rollups stand alone and the batch view renders without the cross-session paragraph. Context:

- The deterministic rollups
- Each session's narrative excerpt + flagged-for-clinician moments
- The full anomaly density per session
- Talk-time evolution across the series
- The outlier set with what made each one different

The LLM produces a single paragraph framing the trajectory ("F0 variability is recovering across the series, gaze-toward-camera fraction is stable, two sessions in late-March cluster as outliers on linguistic disfluency — flagged for clinician review") + a list of cross-session moments-worth-discussing and a list of follow-up questions for the family. Same JSON contract as a single-session pass, same hard constraints (no diagnostic language, every claim cites the underlying sessions or research).

Why a separate component (vs. one giant prompt). Stuffing 25 sessions $\times$ 30 windows each $\times$ 32 features = 24,000 feature numbers into a single context window is technically possible on Opus 4.7 but produces a model that's drowning in numbers and skipping the narrative work. The two-pass design forces the numeric reduction first (deterministic, cheap, reproducible) and then asks the model the *only* question that requires its capability: pattern-naming across the rolled-up summary. The narrative output is reproducible against a fixed rollout — same numbers always produce the same prompt — which matters for clinical-audit traceability later.

Failure isolation. If the LLM narrative pass fails (rate limit, network, etc.), the batch is still complete and clinically usable — the deterministic rollups + per-session narratives are the bulk of the read. Re-running the cross-session pass alone is one button in the batch detail view; no need to re-analyze every file.

4.9 Interactive chat over Cadence data

`Indicator` s, moments, narrative paragraphs, and the agent-finding trail give a complete read for the session that already happened — but they don't answer the follow-up question. "What did the F0 variability look like compared to her prior three?" "Which sessions in this batch had the strongest baseline divergence and why?" The dashboard can't predict every question; the chat surface is for the long tail.

Where it lives. A `ChatPanel` (SwiftUI) appears as a right-hand pane on `SessionDetailView` and `BatchAnalysisDetailView` , scoped to that session's UUID or that batch's UUID + the constituent session IDs. Each scope keeps its own conversation history across launches — same chat about the same session resumes where the user left off. Toolbar toggle hides the pane when the dashboard needs every pixel.

Above the chat-message scroller sits an **agents intelligence bar** — a horizontal scrollable strip of clickable chips, one per agent that fired for this session, ordered with the Synthesizer first. Each chip carries the agent's stable glyph + accent color from `AgentVisualStyle` and pops a small popover on click containing (a) the agent's display name + runtime, (b) the first sentence of its summary, (c) the suggested-indicator line if any, and (d) three canned drill-in prompts that send directly to chat — no input-field prefill, no two-tap flow. Synthesizer-chip prompts focus on cross-modal reconciliation; specialist prompts focus on "what did you see / did you disagree with anyone / how repeatable is this." Empty agent list (rule-based-only sessions, pre-synthesis live sessions, batch scope) hides the bar entirely.

How it works end-to-end.

1. User types a question, hits send. `ChatRunner` (`@MainActor`) persists the user turn locally and opens a stream.
2. `CadenceChatGateway` POSTs `{kind: "chat", messages, scope, model_override?}` to the existing cadence reasoner Function URL — the same Lambda the per-session synthesis call hits,

just a different payload branch. No new endpoint, no new auth path, no extra deploy surface.
Auth: Cognito Bearer JWT + `X-Cadence-Body-SHA256` + the per-install device id, same as every other reasoner call.

3. The Lambda's chat branch dispatches a bounded streaming OpenAI tool loop (max 12 iterations) with five **cadence-search tools** available:
 - `cadence_list_subjects` — lists the (project, account, clinician) combos the caller can read, gated by `project_memberships` at the SQL layer
 - `cadence_list_sessions` — filtered metadata (project / subject / time / modalities / indicator kinds / sort)
 - `cadence_get_session` — full session payload with trim-by-default `include` list (`indicators`, `moments`, `anomaly_frames`, `agent_findings`, `narrative`, `talk_time`, `citations`, `feature_windows`)
 - `cadence_indicator_trend` — score series + monotonic-trend tag for one indicator across a subject's recent sessions
 - `cadence_synthesize` — a bounded inner reasoning loop over the four read tools, for complex multi-session questions where the outer model wants to defer
4. As the model emits text and tool calls, the Lambda streams NDJSON events back:
`{type:"delta",data:"..."}` for assistant text tokens, `{type:"tool_status"}` and `{type:"tool_read"}` for tool dispatch + return, `{type:"done", usage, model_used, iterations}` when the loop terminates. The client's `CadenceChatGateway` parses with a manual byte accumulator (same defensive pattern the SSE reasoner uses to dodge `URLSession.AsyncBytes.lines` silently truncating multi-KB frames).
5. `ChatRunner` accumulates deltas into a live "in-progress assistant turn" while ephemeral tool-event rows ("Reading session abc12345...") render in the pane. When `done` lands, the assistant turn commits to local SQLite (`conversations` + `conversation_messages`, JSON-blob payloads keyed on a composite `scope_index` = `"session:<uuid>"` or `"batch:<uuid>"`).

Tool-scope discipline. The five cadence-search tools live in the central tool registry but are **only** exposed to the cadence reasoner's chat dispatch, via the `includeCadenceSearchTools` flag on `buildFunctionDefs`. They're absent from every other catalog — main account-chat orchestrator, project-chat, admin orchestrator, housekeeping, and the 8-agent session-analysis specialist orchestrator. This is enforced both by the explicit append step and by a belt-and-braces per-tool filter, with a `CADENCE_SEARCH_TOOLS` constant the cache-key includes (DynamoDB tool-definitions cache flushed at v32 on the deploy that landed this scoping). Net effect: the only place the federation data is reachable through a model is the chat surface where the per-session / per-batch UI scope anchors the call.

Persistence + privacy. Conversations are local-SQLite only — there's no backend table, no sync across installs, no shared transcript. On a device wipe they're gone (acceptable for v1). Each turn the client sends includes the full visible conversation history (the backend re-runs tool calls each turn rather than persisting a server-side scratchpad). Tool-event rows are display-only — stripped before re-send.

Failure handling. A failed stream (HTTP non-2xx, JSON parse error, model hits the iteration cap) lands as an `error` row in the pane with the backend's code + message. The conversation continues — the next user turn re-arms the loop. The daily reasoner quota covers chat the same way it covers session synthesis, so chat habits can't dodge the per-account cap.

Why not the planned local-Claude approach. An earlier `CHAT_SCOPE.md` plan called for a local `ChatRunner` invoking the existing `cadence-reasoner/messages` proxy with client-side tool execution against `SessionStore`. The shipped architecture is server-side — chat tools dispatch from the Lambda, not the Mac — because the federation gate (`project_memberships` SQL join) is already server-side and duplicating it client-side meant either re-implementing the gate in Swift or shipping the user's full session data to the model on every turn. Server-side dispatch keeps the gate where the data lives. Same auth path, same Lambda, same quota — the only thing that changed is one branch added to the existing handler.

5. Privacy boundaries

Data	Storage	Leaves device when
Raw audio (mic)	RAM during capture, by default. Admin opt-in: 16-bit PCM <code>.wav</code> at <code>sessions/{id}/audio.wav</code>	Never
Raw audio (remote process tap)	RAM during capture by default — VAD'd for voiced-seconds + turn-taking latency and immediately discarded. Admin opt-in only: 16-bit PCM <code>.wav</code> at <code>sessions/{id}/remote-audio.wav</code> (separate file from the user's mic; double-gated on per-call <code>remoteAudioTap</code> consent)	Never
Raw camera frames	RAM during capture	Never
Raw screen frames	RAM, discarded	Never
Raw video (screen preferred over camera)	Admin opt-in only: H.264 <code>.mp4</code> at <code>sessions/{id}/video.mp4</code> (~2 Mbps)	Never
Transcript + per-word timings	SQLite, only if <code>transcriptStorage</code> consented. Speaker-tagged "user" (local mic only)	Never by default
Talk-time totals	Stamped on Session row. "Other" only present when remote-tap consent was on	Never
Feature windows	SQLite	Only if user enables cohort participation, and then only as fingerprint hashes
Key frames	JPEG on disk, only if <code>keyFrameArchive</code> consented	Never
Indicators + moments	SQLite	Never
Salted fingerprint	SQLite	Only with explicit cohort opt-in
Feature stream → cloud orchestrator	Sent on every per-session synthesis call to <code>cadence-messages-reasoner</code> (the shipping path, not opt-in)	Structured features + retrieved research + repeatability hints + rule-based prior + anomaly key-frame metadata only. Raw audio, video, and transcript content never sent (transcript text itself is opt-in per-session)
CSV feature export	User-chosen NSSavePanel destination	Only where the user explicitly saves the file
Imported recordings (bulk path, §3.14)	Read once into AVFoundation buffers; never copied to Cadence's session folder unless raw-retention opt-in is on (same gate as live capture)	Never. The source file the user dropped in stays where it was; Cadence works from the same in-memory buffer pipeline the live path uses.
Cross-session rollups (<code>BatchAnalysis</code>)	SQLite <code>batch_analyses</code> table on the operator's Mac	Only via the auto-sync queue (§3.15) when the operator has opted in. Structured rollups only — never raw audio/video.
Per-project session artifacts	<code>~/Library/Application Support/Cadence/projects/<project_id>/sessions/<id>/</code> instead of the flat path when a project is bound on the install	Same as flat-path artifacts — never by default, opt-in for the

Data	Storage	Leaves device when
Chat conversations (§4.9)	SQLite <code>conversations</code> + <code>conversation_messages</code> on this Mac. Keyed per session UUID / batch UUID; never synced across installs	auto-sync structured upload only Each user turn POSTs the visible message history to the cadence reasoner Function URL so the model can continue the thread. Tool calls dispatch server-side, gated by <code>project_memberships</code> — the federation gate is enforced where the data lives, not in client code

The `cohort-salt` is a 256-bit random value generated on first run, never transmitted, and is what makes two users' fingerprints uncorrelatable even on identical features.

5.1 Admin-panel access control

The Admin panel exposes controls (reasoner backend, prompt content, federation, raw recording retention, anomaly thresholds, cost / quota config, ...) that should not be accessible to anyone who simply downloaded the app. The gate is **server-validated against an Eleplan-account-bound password**, not a local secret.

Flow. When the user opens Admin for the first time, `AdminGateView` resolves which mode applies:

Condition	Mode	Validation source
Valid persisted Keychain grant	Auto-unlock	Local expiry check on the cached grant
Signed in to Eleplan, no grant	Server-backed form	<code>POST <stage>.eleplan.link/cadence/admin/verify</code> with the user's Cognito Bearer JWT — verified against the bcrypt/argon2id hash on the user's account row
Not signed in to Eleplan	Local-passphrase fallback (legacy)	SHA-256 vs. <code>AdminSettings.passphraseHash</code>
<code>CADENCE_ADMIN=1</code> env var	Dev bypass	Always unlocked

The grant. On a successful server verification the endpoint returns an opaque `grantToken` + `grantExpiresAt`. The client persists this as an `AdminAccessGrant` in the Keychain at `org.eleplan.cadence.admin-grant / grant.v1` with accessibility class `kSecAttrAccessibleWhenUnlockedThisDeviceOnly` — strictest tier; the grant is never backed up, never migrated to a new Mac, and is unreadable while the screen is locked. Default TTL 30 days, configurable per-account server-side.

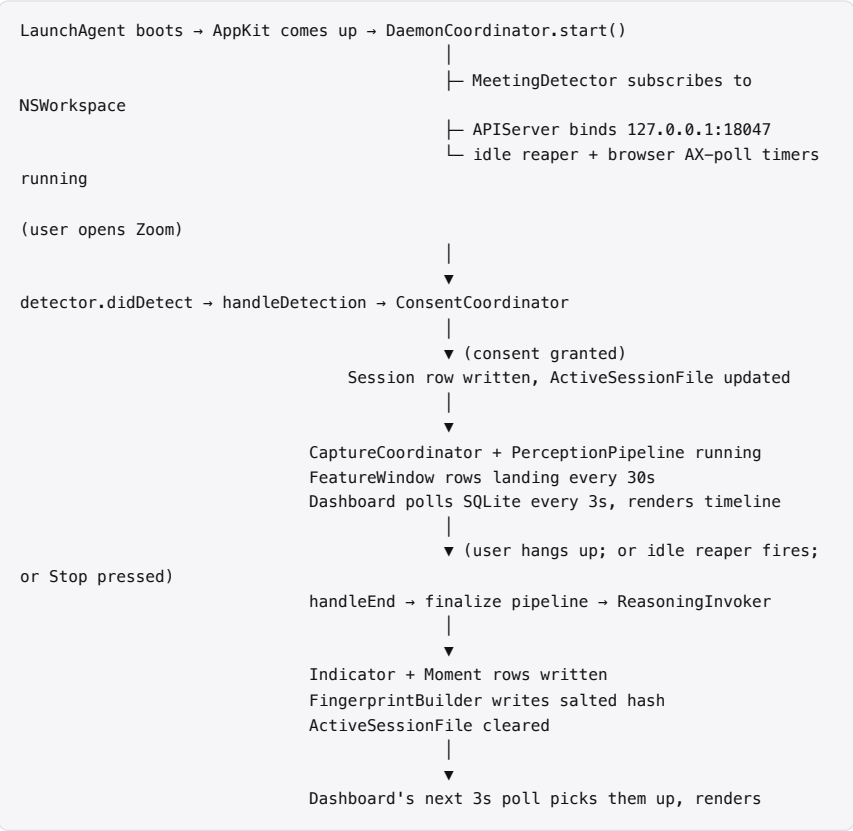
Why this is different from the legacy local-passphrase model. The previous design hashed a user-chosen passphrase locally and stored the hash in `AdminSettings.passphraseHash`. Anyone with physical access to the Mac could reset the passphrase (a "Forgot — reset" button intentionally wiped the hash so a researcher could recover access). Fine for a single-operator research scaffold; wrong for a consumer / clinical / LMS product where the admin password needs to be a real authentication boundary tied to a real identity. The server-backed model:

- Ties admin access to the user's Eleplan account, not the device — same password works across every Mac they sign in on
- Cannot be reset by anyone with physical access (reset goes through `eleplan.com/account/cadence-admin` and requires the Eleplan sign-in password to authenticate)
- Server-side rate-limits brute-force attempts (5 / 15 min / account)

- Server-side audit-logs every verify attempt with user, IP, outcome, grant-token id
- The local fallback is retained only for backward compatibility with existing installs and offline use; it will be removed in a follow-up release once the Eleplan-backed endpoint is in production for all stages

Backend contract. The exact endpoint shape, response codes (200 / 401 / 403 / 404 / 409 / 5xx), data model, rate-limiting, audit-logging, and eleplan.com/account/cadence-admin UX are specified in [docs/ADMIN_AUTH_BACKEND_SPEC.md](#). The client ([Sources/CadenceCore/Federation/AdminAccessGate.swift](#)) gracefully degrades to the local fallback when the endpoint returns 404, so the dual-mode behavior survives a partial backend rollout.

6. Deployment lifecycle



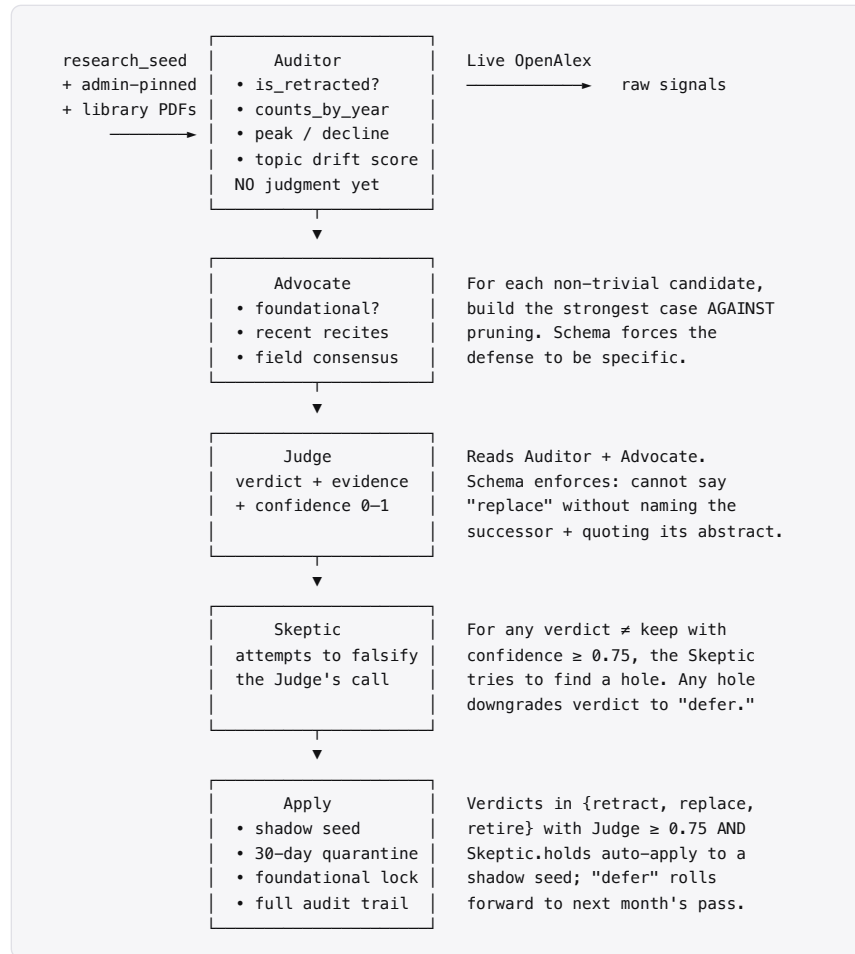
6.5 Research baseline audit — the system reasons about its own grounding

Cadence's analytical interpretations are anchored to a curated research baseline (47 hand-picked seed papers + admin-pinned OpenAlex works + user-uploaded PDFs). That corpus *cannot* be allowed to ossify. Research literature shifts: papers get retracted, methods get superseded, fields refine their consensus over time. A behavioral-signal system that quotes a 2008 study to justify a 2026 interpretation, while a 2024 meta-analysis sits unread on the public internet, is silently wrong in proportion to how much time has passed.

The Research Library Audit is Cadence's answer. It is a self-referential pipeline that watches the corpus the system itself relies on, flags drift, and — in the operator's chosen tier of autonomy — retires what no longer holds.

Terminology for prior-art search. Internal docs use Cadence-specific terms ("fleet consensus," "gossip-over-S3," "careful-pass verdict," "foundational allowlist"). For IP-counsel prior-art search and USPTO-classification purposes, the same mechanisms map to industry-standard terminology as: *distributed voting consensus system* (the fleet-consensus aggregator), *Byzantine-resistant federated trust* (per-account capping + upstream-identity anchoring), *self-auditing knowledge base / RAG corpus evolution* (the full audit pipeline), *shadow-config quarantine* (the 30-day [pending_quarantine](#) window between verdict and effect), *reproducibility stamping for grounded LLM systems* (the per-finding [seed_overlay_version](#) provenance field). These industry-standard terms should be the search-query basis for prior-art clearance.

The pipeline (Auditor → Advocate → Judge → Skeptic → Apply)



Each pass has a system prompt stored as a server-side rule ([system_settings.rules](#)) so the language is admin-editable without a redeploy. Each pass has a strict structured-output schema (Responses-API [json_schema strict](#)) so the model cannot drift into prose verdicts or skip required evidence.

The shipped pipeline (hands-off)

The audit ships as an **autonomous protocol**, not a click-to-run feature. The Audit tab is observer-only — there is no "Run audit now" button and no per-verdict "Apply" button. Three actors close the loop without operator intervention:

- [ResearchAuditScheduler](#) (**Mac actor**) runs at app launch. Gated on [researchAuditCadenceDays](#) (default 30 days, admin-tunable). When the cadence is due:
 1. Phase 1 — deterministic auditor runs locally against the live OpenAlex API, classifying every citation into **Healthy**, **Watch**, **Action**, **Foundational** (admin-marked allowlist, protected from

any future pruning), or **Unresolved** (could not match in OpenAlex; will retry). Bucketization uses deterministic rules — retraction, citation-velocity decline thresholds, topic drift — with one plain-English reason string per signal so the *why* is visible alongside the *what*.

2. Phase 2 — careful-LLM pass on the Action / Watch / Unresolved subset (Healthy + Foundational never reach the model). The Auditor's signals feed the **Advocate**, the Advocate's defense feeds the **Judge**, the Judge's verdict feeds the **Skeptic** for any verdict ≠ keep with confidence ≥ 0.75. Verdicts persist append-only to a local `research_audit_verdicts` SQLite table — the history of what the model decided over time is preserved even after a future audit overwrites the underlying signals.
3. `FleetVoteStore.uploadCurrentLog` publishes this install's verdict log to the public-read S3 `votes/` prefix (gated by `fleetSyncEnabled`, default ON).
- **FleetSyncEngine** (Mac actor) also runs at app launch. Lists peer vote logs from S3, fetches them in parallel, runs `FleetConsensus.compute` locally (pure deterministic aggregator — same input across builds → same output), and applies converged outcomes to the local `ResearchStore` overlay. A converged outcome means **N distinct partner accounts** (default 3, admin-tunable) **agreeing on the same verdict** at confidence ≥ 0.75 with vote age ≤ 60 days. Per-account, not per-install — one partner running 20 installs that all agree contributes 1 to the count.
- `promoteExpiredQuarantine` (Mac job) runs at app launch. Sweeps the local overlay for pending quarantine entries whose 30-day `release_at` timer has passed; flips them from `pending_quarantine` into the active `retired_keys` / `replaced` sets. Until promotion the verdict is reversible — the operator's window for an explicit veto.

Operators are observers throughout. The Audit tab surfaces what the model has decided, which entries are in quarantine, what the fleet's consensus state is, and how soon any pending verdict will promote. The only operator-touchable surface remaining is the **foundational allowlist** in admin — the unappealable veto floor: the apply step never retires or replaces an allowlisted citation, regardless of what the model or the fleet concludes (upstream prevention, not after-the-fact restoration).

Six guardrails that make autonomy safe

1. **Bundled seed never mutated; deltas layered at read time.** The canonical `research_seed.json` is immutable per release — every Cadence install on the same build starts from the same baseline. Active corpus = `bundled_seed - overlay.retired_keys + overlay.replaced.to`. Two set operations at corpus-read time (`ResearchStore.currentAll()`). Trivial full-revert: delete the local overlay file and the seed snaps back to the build's known-good shipped state.
2. **seedOverlayVersion on every persisted AgentFinding.** Every indicator records which overlay version was active at the moment of reasoning. Clinical reproducibility intact even as the corpus evolves — "the analysis that produced last quarter's report was grounded in *these exact papers*" remains answerable by joining (`bundled_version_at_build, finding.seedOverlayVersion`).
3. **Foundational allowlist (hard floor).** A small admin-curated set of papers (`AdminSettings.researchAuditFoundationalAllowlist`) marked as foundational. Audit + verdict pipeline + fleet consensus all still run on them and surface signals for context — but the apply step refuses to act regardless of model decision or fleet agreement. The only artifact in the system the model and the fleet cannot touch.
4. **Confidence + Skeptic both required.** Judge confidence ≥ 0.75 is necessary but not sufficient — the Skeptic must also fail to attack. Either alone is too noisy. Skeptic is confidence-gated to high-confidence verdicts only (inverts the typical "low confidence → more scrutiny" pattern).
5. **30-day quarantine.** A converged verdict — whether from this install's careful pass or from the fleet — does not modify the active corpus at decision time. It lands in `pending_quarantine` with a 30-day `release_at` timer. During the window, removing the entry from the local overlay vetoes promotion. Only after the window expires does `promoteExpiredQuarantine` (launch-time job) move it into `retired_keys` / `replaced` and the paper actually leaves that install's active baseline.
6. **N distinct partner accounts required for fleet apply.** A verdict crosses the fleet-consensus threshold only when **N distinct partner accounts** (default 3) agree on the same verdict at

confidence ≥ 0.75 with vote age ≤ 60 days. Per-account capping — one partner running 20 installs that all agree contributes 1 to the count — prevents a single partner from dominating consensus. Partner-account identity is anchored upstream at pair-code redemption, so Sybil resistance rides existing trust machinery.

Why this matters

A grounded-in-citations system that *cannot* update its citations is a system that drifts out of validity over a horizon set by the publication cadence of its field — fast for behavioral neuroscience, faster for ML-adjacent topics. The audit makes Cadence's accuracy floor a function of effort rather than a function of the bundling date.

It is also the thing that lets us say truthfully: Cadence does not just *use* its research baseline. It *reasons about* its research baseline. The same evidence-required, schema-enforced, debate-pattern discipline that justifies the per-session findings now justifies what gets to be in the corpus that produces those findings.

The cost is bounded by design ($\approx$ \$2/month at default cadence for a 47-paper seed; per-agent budgets prevent runaway). The risk is bounded by design (shadow seed + quarantine + foundational lock + reproducibility metadata). The value compounds: every monthly audit moves the corpus one notch toward what the field currently believes, and one notch away from what the field believed when this build was cut.

The shipping pipeline is the result of five rounds of compression. Phase 1 (read-only Audit tab) shipped in v0.1.105. Phase 2 (careful-LLM verdicts with a manual Apply button) shipped in v0.1.106. Phase 3 shipped in v0.1.107 as a per-study server-distributed overlay with a nightly server-side promotion cron. Phase 4 (v0.1.108–v0.1.109) collapsed the overlay to per-install local-only (`~/Library/Application Support/Cadence/research_overlay.json`) and moved the promotion cron to a launch-time `promoteExpiredQuarantine()` job on the Mac. v0.1.110–v0.1.114 wired the autonomous scheduler, the S3 vote bucket, the fleet-vote upload, the consensus aggregator, and the operator-out-of-loop UI rework (Apply / Cancel buttons gone). **v0.1.115** removed the last two operator-touchable surfaces — "Run audit now" and "Force re-run" — making the Audit tab strictly observer-only. The loop is closed end-to-end. The remaining gaps: an automated OpenAlex `is_retracted` sweep at launch (substrate present, job unwritten — see §7), and a fleet-vote-payload extension that carries a superseding work for `.replace` verdicts so consensus can apply replacements autonomously (today consensus surfaces `.replace` as a UI suggestion only — see §7).

The two architectural inversions

The novelty of this subsystem is concentrated in two structural inversions of patterns that are otherwise standard in the LLM-ensemble and distributed-consensus literature. Both are independently claimable; together they define the architectural posture.

Inversion 1 — Skeptic runs on HIGH confidence, not low. Most LLM ensembles use a sanity-check pass gated by *low* confidence: when the model is unsure, ask a second model. Cadence inverts this. The Skeptic only runs when Judge confidence ≥ 0.75 . The mechanism's purpose is **falsification of confident verdicts**, not approval of uncertain ones — finding *any* hole in the Judge's reasoning downgrades the verdict to `defer`. Closer in spirit to Karl Popper than to ensemble voting. Implements as the `SKEPTIC_CONFIDENCE_FLOOR = 0.75` gate in `backend/api/client/services/openai/orchestrators/research-audit-careful-pass.js`.

Inversion 2 — Trust anchor is upstream IDENTITY, not downstream COMPUTATION. Classic distributed-consensus systems (Byzantine Fault Tolerance, Paxos, Raft; blockchain proof-of-work and proof-of-stake; DAO governance) prove trust through what the node *did* — work, stake, signature chain. Cadence's Phase 4 fleet-consensus engine inverts this. Trust comes from who the node *is*: partner-account identity is gated upstream at pair-code redemption (a separate trust-establishment phase). The gossip channel only needs to verify "distinct accounts agreed," not "valid work was performed." Per-account capping (one partner running 20 installs counts as 1) replaces stake-weighting structurally. Eliminates PoW/PoS overhead entirely. Implements as the `SHA256(account_id + per-stage SSM salt)` hash on `FleetVoteStore.publish` and the `unique_account_hashes.count` semantics in `FleetConsensus.compute`.

What is novel here (patent context)

The provisional patent application (</patent/provisional-spec.pdf> , §1.5 above) describes the multimodal behavioral-signal layer at Cadence's core: capture → perception → fusion → indicators. The research-baseline self-audit described in this section is a **structurally distinct invention** sitting on top of that layer. It is not covered by the provisional's existing forks 1–10 and is being documented here in the level of detail required for a follow-up nonprovisional filing or a continuation-in-part addendum. See </patent/README.md> § "Self-auditing grounding corpus" for the claim-element-to-code mapping; Phase 4 elements (i'), (l) – (p) covering the gossip-over-S3 fleet-consensus extension are drafted in </patent/README.md> § "Independent method-claim elements — Phase 4 extension."

The novelty the self-audit pipeline asserts:

1. **A grounding-corpus-evaluating model layered onto a grounding-corpus-using system.** The same class of LLM whose reasoning is grounded by a fixed research corpus is, on an independent cadence, made to *evaluate* that same corpus and propose deltas to it. Most contemporary RAG / citation-grounded systems treat the corpus as immutable infrastructure shipped with the build; here the corpus is a first-class evolving artifact, and the evolution is itself a model-driven loop with structural safeguards. We are not aware of prior art that combines (a) deterministic citation-velocity triage, (b) an adversarial Advocate → Judge → Skeptic verdict pipeline, (c) shadow-seed staging with quarantine and reproducibility stamping, and (d) org-wide overlay distribution into a single coherent system applied to a clinical-research-grounded behavioral-signal tool.
2. **Adversarial-defense-before-judgment four-pass design.** The Auditor → Advocate → Judge → Skeptic sequence inverts the typical chain-of-thought verdict pattern. The Advocate's job is not to summarize — it is to *build the strongest case against the system's default action* (pruning the candidate paper) and is schema-required to either name specific recent recitations / methodological dependencies or set `cannot_defend = true` . Only then does the Judge weigh signals against defense. This structurally addresses the "model defaults to action when uncertain" failure mode that plagues single-pass verdict systems.
3. **Skeptic as confidence-gated falsification, not confidence-gated approval.** The Skeptic pass only runs on verdicts with Judge confidence ≥ 0.75 — i.e., *high*-confidence verdicts get *more* scrutiny, not less. The Skeptic's role is to find a hole in the Judge's reasoning; finding one downgrades the verdict to `defer` . Inverts the typical "high confidence $\Rightarrow$ no further check" pattern in LLM ensembles. (Low-confidence verdicts skip the Skeptic because they are already gated below the auto-apply threshold and proceed only to the manual-review surface.)
4. **Quarantine-with-operator-veto window between verdict and effect.** A verdict in `{retire, replace, retract}` does not take effect on the active research baseline at decision time. It lands in a local `pending_quarantine` queue inside the per-install `research_overlay.json` with a 30-day `release_at` timer. During the window the operator at this install can remove an entry to veto it. Only after the window expires does the launch-time `promoteExpiredQuarantine()` job move it into `retired_keys` / `replaced` and the paper actually leaves that install's active baseline. This decouples the model's call from the operator's adoption — autonomy without irreversibility.
5. **Foundational allowlist as veto floor (unappealable).** A small operator-curated set of papers is marked `foundational: true` (via `AdminSettings.researchAuditFoundationalAllowlist`). The Auditor and verdict pipeline still run on them — surfacing signals for context — but the application step refuses to act on them regardless of any verdict the model reaches. The allowlist is the only artifact in the system the model cannot influence. It exists to handle the "low citation velocity $\neq$ low importance" case (textbook-folded methods like FACS-13 are rarely cited but foundational).
6. **Bundled seed never edited; overlay is small deltas applied at read time.** Active seed = `bundled_seed - overlay.retired_keys + overlay.replaced.to` . Two set operations at corpus-read time. The shipped JSON in the app binary is immutable per release. This makes (a) every state of the active baseline at any past instant deterministically reconstructible from `(bundled_version, overlay_version)` , and (b) full revert trivial — delete the overlay file and the seed snaps back to the build's known-good shipped state.

7. `seed_overlay_version` stamped on every produced indicator. Every `AgentFinding` Cadence persists carries the integer `overlay_version` that was active at the moment of reasoning (`ResearchStore.currentVersion()`). The historical state of the corpus is recoverable for any prior analysis by joining (`bundled_version_at_build,` `finding.seed_overlay_version`). This solves the well-known reproducibility-breaks-when-corpus-evolves problem for grounded-LLM systems: most lose this when the corpus changes; here it is preserved by construction.
8. **Per-install local overlay + federated consensus on top of a bundled baseline.** Cross-install consistency operates at two layers, each delivered without a central authority. (a) **Baseline:** every Cadence release ships the canonical `research_seed.json` ; installs on the same build start from identical corpus. (b) **Deltas:** each install publishes its post-careful-pass verdicts as a per-install JSON log to S3 (presigned PUT, ~one Lambda invocation per install per careful-pass run); every install on launch reads every peer log via plain public-read HTTPS, runs the deterministic consensus aggregator locally (§ claim 9 below), and applies converged verdicts to its own overlay's 30-day quarantine. The aggregator is a pure function in the Mac binary — same input across installs → same output across installs → distributed convergence without server mediation.
9. **Federated consensus via gossip-over-S3 with per-account capping and operator-out-of-loop semantics.** The fleet-consensus engine (`FleetConsensus.compute`) counts DISTINCT partner accounts (not installs) agreeing on the same verdict at confidence $\geq$ threshold; one partner running 20 installs that all agree contributes 1 to the account count, preventing single-partner dominance. Per-account identity is preserved via `SHA256(account_id + per-stage SSM salt)` so the gossip channel never reveals which accounts voted what — only that distinct accounts agreed. Operators are observers throughout: the careful-pass UI is read-only, the autonomous scheduler produces verdicts without user action, the consensus engine applies converged verdicts without user action, the 30-day quarantine + foundational allowlist are the only safety latches. The composition (a) inverts the classic peer-to-peer trust problem — there's no proof-of-work or proof-of-stake because partner-account identity is the trust anchor, gated upstream at pair-code redemption; (b) eliminates the per-org authority figure that earlier Phase 3 designs required; (c) keeps the read path Lambda-free (raw S3) so cost is bounded by sync volume, not by fleet size.
10. **Single quarantine pipeline for multiple verdict sources.** Operator-triggered careful-pass verdicts, fleet-consensus-applied verdicts, and (future) Mac-side OpenAlex retraction-sweep verdicts all enter the same `pending_quarantine → 30-day timer → promote` pipeline. The 30-day window is a time-based safety latch — verdicts originating from any source must wait it out before reaching the active baseline. New verdict sources can be added later by writing to the same queue; they inherit the quarantine guarantee automatically.
11. **Cost-bounded by deterministic-prefilter design.** The expensive LLM verdict pipeline is gated by the cheap deterministic Auditor pass. Healthy + Foundational citations never reach the careful pass — only Watch / Action / Unresolved do. For the default 47-paper seed, this typically routes 8–15 candidates to the model per audit cycle (≈ 0.05 each, 0.50/month at gpt-4.1-mini), independent of corpus growth as long as the deterministic prefilter scales linearly.

The system is also disclosed for application beyond Cadence's specific behavioral-signal context: any LLM-driven application whose outputs are grounded by a curated literature corpus has the same drift problem, and the pipeline + guardrails described here apply directly. Domain-specific instantiations would substitute their citation source (e.g. PubMed, arXiv) for OpenAlex; the structural pattern is invariant.

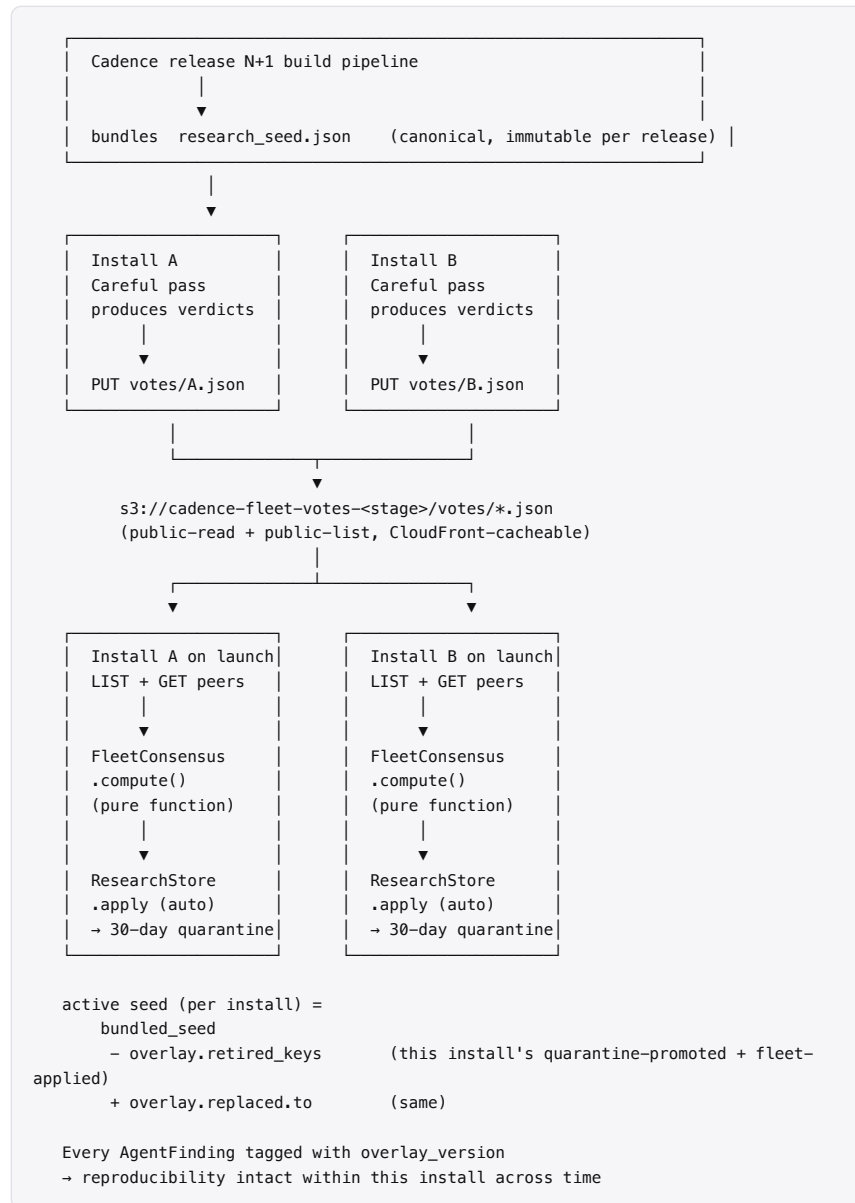
Cross-instance distribution — bundled baseline + fleet consensus on the deltas

Phase 3 originally distributed a versioned `research_seed_overlay.json` from a central audit run; every install polled, fetched, and re-applied. Phase 4 retired that channel because it required a per-study / per-org authority figure — someone whose verdict propagated to every install in that org. The wrong shape for a research tool whose operators are individual clinicians and researchers with legitimately different reads on the same literature.

The corrected architecture splits cross-install consistency into two layers, both delivered without a central authority:

Layer 1 — Baseline sync via the build pipeline. Every Cadence release ships the canonical `research_seed.json` (~50 hand-picked papers + their metadata) in the app binary. Operators on the same build start from the same baseline. Updates to the baseline ship with the next release of Cadence; the build pipeline IS the corpus-update channel. Zero network cost, zero coordination cost, identical in mechanics to shipping any other bundled resource.

Layer 2 — Delta sync via fleet consensus over gossip-over-S3. Each install publishes its post-careful-pass verdicts (citation_key, verdict, confidence, model_id, produced_at, account_id_hash) as a JSON log to `s3://eleplan-backend-cadence-fleet-votes-<stage>/votes/<install_id>.json`. Writes use a 1-hour presigned PUT URL refreshed on demand from a tiny backend Lambda (`POST /cadence/fleet-vote/presign`) — roughly one Lambda invocation per install per careful-pass run, ~ monthly at default cadence. Reads are pure HTTPS to the bucket's public-read + public-list policy on the `votes/` prefix — Lambda-free, S3-only. Every install on launch lists peers, fetches each peer log in parallel, and runs the deterministic consensus aggregator (`FleetConsensus.compute`) locally:



The consensus rule (v1 defaults; admin-tunable):

- **N=3 distinct partner accounts** must agree on the same verdict before it crosses threshold. Per-account capping, not per-install: an account running 20 installs that all agree contributes 1 to

this count. Identity preserved via SHA256(account_id + per-stage SSM salt) — gossip channel never reveals which accounts voted what.

- **Per-vote confidence ≥ 0.75** (matches Skeptic floor) — low-confidence votes don't count toward threshold.
- **Vote age ≤ 60 days** (2× default `researchAuditCadenceDays`) — stale votes from long-paused installs can't drag historical opinions forward.
- **Verdict ties produce no consensus** — when two verdicts have equal distinct-account support, the citation stays unmodified. Don't coin-flip when the fleet is genuinely split.
- **keep / defer never produce outcomes** — there's no action to apply.
- **Foundational allowlist short-circuits before apply** — operator-curated set the model can never override regardless of consensus.

What this buys:

- **Per-operator autonomy preserved.** No central authority means no one operator's call propagates to others without genuine multi-account agreement. Differing reads coexist; the fleet's *durable* consensus is what eventually flips a baseline entry, and only behind the 30-day quarantine timer.
- **Per-install reproducibility intact.** Every finding records the `overlay_version` active at reasoning time, regardless of what triggered the overlay change. Past analyses replay against their historical seed.
- **Sybil resistance via existing trust anchors.** Pair-code redemption gates fleet participation — a partner can't fake 100 accounts to skew consensus. Per-account capping prevents one partner with many installs from dominating.
- **Cost is bounded by sync volume, not fleet size.** Write path: ~1 Lambda call per install per careful-pass run, $\approx$ monthly. Read path: pure S3 (CloudFront-cacheable, no Lambda). 100 installs $\times$ 4 KB vote logs $\times$ monthly refresh = pennies/month.
- **Self-improving baseline.** Every passive operator (the explicit operator-out-of-loop story) still contributes verdict signal to the fleet. Over time, the bundled `research_seed.json` itself gets updated based on fleet consensus history — the canonical baseline learns from the deltas the fleet converged on, shipping that learning in the next release. (The bundled-seed update path is manual today; the consensus engine produces the evidence trail; the build-time decision of which fleet-consensus verdicts to fold into the next baseline release stays human-curated.)
- **The patent claim consolidates** rather than simplifies. The earlier per-study overlay was a distribution mechanism; this is a *consensus* mechanism. The structural novelty: trust-anchored fleet + on-device pure-function aggregation + per-account capping + 30-day quarantine + foundational floor + reproducibility stamp + bundled baseline as the canonical anchor. The composition is what's defensible; no individual component is.

A future *opt-in* extension would route the careful pass itself onto on-device Apple Intelligence rather than the server-side OpenAI orchestrator — same pipeline, free compute, fully self-hosted. See §6.6 for the forward-looking design.

7. Honest gaps

This section is *current state* — gaps closed since the prior whitepaper revision are listed first so the reader can see what moved, then the gaps that genuinely still stand.

Closed since the previous revision

- **Multimodal model input.** (Was: “the model doesn't see frames.”) Now partially closed. The visual-narration gateway (`Sources/CadenceCore/Federation/VisualNarrationGateway.swift`) samples evenly-spaced frames from a session and posts them to a GPT-4o-mini scene-description endpoint; the returned per-frame observations + 1–2-sentence summary attach as a `VisualNarrator` AgentFinding the synthesizer consumes alongside the numeric features. The interactive chat panel (§4.9) further closes the loop — the model can ask follow-up questions over the captured data via the cadence-search tool catalog. Per-frame stills still aren't sent to the synthesizer call directly; that's the remaining slice.

- **Streaming events on the HTTP API.** (Was: "a future SSE/WebSocket endpoint should...") Closed via `LiveEventBus` + the `/v1/stream` SSE multicast (§3.16) — nine event kinds including `window.emitted`, `reasoner.fired`, `agent.fired`, `indicator surfaced` stream to subscribed dashboards in real time.
- **Live literature access from the model.** (Not in the prior list — adding it as a "closed-before-noticed".) Specialist agents AND the chat panel can call `openalex_search` mid-reasoning to consult the full 250M-paper OpenAlex corpus when their pre-loaded research context doesn't cover the pattern they're seeing (§3.6, §4.9). Bounded budgets prevent runaway cost.
- **Visual feature fidelity on uploaded recordings.** (Was: "necessarily coarser than live capture.") Closed as an opt-in. Default reprocessing still caps Vision at 10 Hz and the BlendshapeClassifier at 2 Hz so a casual 10-min 720p import finishes in ~1 min on M-series silicon. High-fidelity mode (`AdminSettings.reprocessHighFidelityEnabled` + a per-import override toggle on the DryRun import sheet) drops both caps to 0 and processes every decoded frame — same fidelity as live capture, sub-100 ms blink onsets and sub-500 ms AU transients visible in the timeline. Cost: ~5x wall-clock. Default off because most uploads are casual; research / clinical recordings flip it on per-recording or per-install.
- **Research baseline audit closed end-to-end.** (Was: "Phase 2 only; Phase 3 designed but not wired.") All phases now ship. **Phase 1:** deterministic Auditor signals in the Audit tab (Cadence#312). **Phase 2:** careful-LLM verdict pipeline (Auditor → Advocate → Judge → Skeptic) emits per-citation verdicts via the `kind: "research_audit_verdicts"` route on the cadence reasoner Function URL; verdicts persist append-only locally with full evidence trail (eleplan/core#655 + Cadence#316). **Phase 3** originally shipped a per-study server-distributed overlay (eleplan/core#656 + #657 + #658 + Cadence#319), then **Phase 4 collapsed the overlay to per-install local-only** (`~/Library/Application Support/Cadence/research_overlay.json` , Cadence#328) — the per-study authority shape didn't survive contact with the real onboarding model (individual operators with legitimately different reads on the same literature; see §6.5 "Cross-instance distribution"). The nightly server-side quarantine-promotion cron moved to the Mac as a launch-time `promoteExpiredQuarantine()` job. The composition is now `active = bundled - overlay.retired_keys + overlay.replaced.to` , applied at corpus read time; bundled `research_seed.json` is never mutated and is the canonical cross-install sync (every release ships it). Foundational allowlist (operator-curated via `AdminSettings.researchAuditFoundationalAllowlist`) as unappealable veto floor, reproducibility stamp on every produced indicator via `AgentFinding.seedOverlayVersion` . The full novelty assertion + claim-element mapping is in `/patent/README.md` § "Self-auditing grounding corpus."
- **Phase 4 split — slave-mode vs full-mode installs.** (Was: "still rides parallel school / study auth machinery.") Largely shipped. The `.school` install kind, `studyID` / `studyName` , per-study overlay scoping, `ResearchAggregateShipper` + `Roller` , `CadenceBundledSeedManifestPublisher` , `RemoteDefaultsClient/Store` , `CadenceAdminDefaultsBlob` , and `AdminSettings+RemoteDefaults` were all removed from the Mac (Cadence#328). The surviving install kind is `.clinical` with two sub-shapes: project-scoped (full mode — Mac runs the entire app surface) and partner-scoped (slave mode — pseudonymous capture client, recording + upload only). `FederationCredentials.isSlaveMode` gates the split; `RootView.visibleTabs` filters Audit / Overlay / Reasoning tabs out for slave-mode installs (Cadence#327). Backend mint + redeem accepts both sub-shapes through the same `gateByClinicianBearer` helper post-#663. The compatibility shim that auto-converts pre-Phase-4 school-bearer tokens decodes legacy persisted blobs at load time via the tolerant `FederationCredentials.init(from:)` (Cadence#328).
- **Operator-out-of-loop audit + federated consensus engine.** (Was: "operator clicks Apply on each careful-pass verdict.") Closed end-to-end across two layers. **Step 1** (Cadence#332): the Audit tab became read-only; the per-verdict Apply / Cancel buttons came out; the `ResearchAuditScheduler` actor runs Phase 1 deterministic signals + Phase 2 careful pass autonomously on launch on a `researchAuditCadenceDays` cadence. **Step 2a** (eleplan/core#670): S3 bucket `cadence-fleet-votes-<stage>` with public-read on the `votes/` prefix + a tiny `POST /cadence/fleet-vote/presign` Lambda that issues 1-hour presigned PUT URLs (federation-bearer-authed, ~1 invocation per install per careful-pass run).

Step 2b (Cadence#334): `FleetVoteStore` publishes per-install vote logs to S3 after each careful-pass batch; `FleetSyncEngine` lists peer logs + GETs in parallel + runs `FleetConsensus.compute` (pure deterministic aggregator — same input across builds → same output) + calls `ResearchStore.apply` for any citation that crosses the threshold (N=3 distinct partner accounts at confidence ≥ 0.75 by default, all admin-tunable). The 30-day quarantine + foundational allowlist + reproducibility stamp all stay intact; what changed is that the trigger to enter quarantine is now fleet agreement, not an operator click. See §6.5 "Cross-instance distribution" for the full architecture.

- **Stereotypy / stim detection.** (Was: "deliberately not implemented" / "ships only at the cohort-aggregate level.") Now ships as a peer of the existing anomaly kinds (longPause, headBurst, blinkBurst, etc.) — per-session, in-session-relative, surfaced in the same timeline ribbon + Anomaly Histogram panel + cross-session synthesizer that all other anomalies use. The detector fires on ≥ 3 consecutive feature windows of elevated `headMovementRMS` ($\geq$ session mean + 0.75σ) with low window-to-window variance — the rhythmic-vs-bursty distinction that separates a stim from a flinch. Admin-toggable via `AdminSettings.anomalyStereotypyEnabled`, default on. Same throttle, same KeyFrame persistence path, no privacy special-casing — just another behavioral signal class.

Still open

- **No clinical validation.** This is a research scaffold. The cohort embedding pool is empty in v0.1 and stays empty until an opt-in cohort recruitment exists. The repeatability layer (§3.11) addresses one corner of the Goldsack 2020 V3 framework (analytical-validation precursor); verification of sensor fidelity and clinical-validation against established scales are still unaddressed.
- **Apple Intelligence dependency is scoped to live interventions.** Per-session synthesis runs cloud-only via `OpenAIRReasoner`; Apple Intelligence is required for the `LiveInterventionEngine` on-device mid-session cards but not for any indicator or moment downstream. If Apple Intelligence is unavailable the live cards simply don't fire and the deterministic rule-based prior + cloud synthesizer keep producing the session's analysis untouched.
- **No automated OpenAlex `is_retracted` sweep yet.** With the overlay collapsed to local-only (Phase 4), the planned retraction sweep moves to the Mac as a launch-time job that would auto-queue retract verdicts into `pending_quarantine` (so retracted papers get the same 30-day-veto-window treatment as operator-applied verdicts). The job is unimplemented; the autonomous careful pass already catches retractions, so this only matters for fully-automated retraction surveillance. The substrate is in place — the Mac already polls OpenAlex via `OpenAlexClient` for the per-citation audit signals — adding a parallel "is_retracted only" pass on launch is a follow-up of perhaps a day's work.
- **Replace-verdict consensus has no apply path.** The fleet-vote wire payload carries `(citation_key, verdict, confidence, model_id, account_id_hash)` but NOT the superseding work for a `.replace` verdict — operator-out-of-loop semantics can't choose successors. `FleetConsensus.compute` correctly surfaces `.replace` outcomes (3 accounts agreeing the paper should be replaced), but `FleetSyncEngine` logs and skips them because there's no way to pick what to replace WITH. A future PR would either (a) carry the superseding work in the vote payload and require N accounts to agree on the SAME successor — strong constraint but yields fully autonomous replace consensus; or (b) surface "fleet agrees this paper should be replaced; with what?" as the one remaining operator-touchable surface. Today: `.retire` + `.retract` consensus apply; `.replace` consensus shows up in logs only.
- **Fleet-consensus provenance not yet surfaced in the UI.** Verdict cards in the Audit tab render an "In quarantine" badge once a verdict enters the local quarantine, but don't yet distinguish "I (this install) produced and applied this on my own" from "fleet of N accounts agreed and applied this on my behalf 2 days ago." The data exists in the verdict's `verdictID` (fleet-applied entries are stamped `fleet-consensus:<citation_key>`); the UI surface to render the distinction is a follow-up PR.
- **The bundled-seed refresh loop is still manual.** Fleet consensus produces a durable evidence trail of which baseline entries the fleet has converged on retiring / retracting. There is not yet an automated "fold last quarter's consensus history into the next bundled `research_seed.json`" step — that update is a build-time human-curated decision today. The bundled seed grows out

of date relative to the fleet's accumulating consensus; the manual update is the corrective. A future "seed refresh proposal" surface would let the maintainer review accumulated consensus and merge the high-confidence ones into the next release's bundle.

- **No Goldsack 2020 V3 verification-of-sensor-fidelity track.** Repeatability is the *analytical-validation* precursor; verification (cross-device fidelity of the underlying audio + video features against laboratory-grade reference instruments) is unaddressed and would need its own measurement campaign.

8. Repo layout

```
Cadence/
├─ Package.swift                # SPM: CadenceCore, CadenceObjC,
CadenceDaemon, cadence-seed, cadence-repeatability
├─ project.yml                 # XcodeGen – produces .xcodeproj with
CadenceCore framework + app + daemon targets
├─ Sources/
│   └─ CadenceObjC/            # @try/@catch wrapper for NSError-
throwing AVFAudio calls
│   └─ CadenceCore/            # Models, storage, capture, perception,
reasoning, API, fingerprints, consent, repeatability
│   │   └─ DryRun/              # Single-file + bulk import;
CrossSessionSynthesizer (§3.14, §4.8)
│   │   └─ Federation/         # Cognito auth, project federation,
/v1/sessions upload coordinator (§3.15)
│   │   └─ Live/               # LiveEventBus + LiveEvent – SSE multicast
for the live dashboard (§3.16)
│   │   └─ Repeatability/      # ICC + WSCV math, MetricRepeatability +
store, IndicatorKind ↔ driver mapping
│   │   └─ CadenceDaemon/      # LaunchAgent executable
│   └─ CadenceSeed/            # CLI: seed the DB with synthetic sessions
for dashboard preview
│   └─ CadenceRepeatability/   # CLI: run / show / clear the per-user
repeatability overlay
├─ App/                        # SwiftUI dashboard target
├─ Resources/                  # BundleInfo.plist, entitlements,
LaunchAgent plist, normative_bands.json, repeatability_bundled.json
├─ scripts/                    # install-daemon.sh (idempotent, ad-hoc-
signed), uninstall-daemon.sh
├─ Tests/CadenceCoreTests/     # 154 tests: YIN pitch, RMS, jitter, store
round-trip, synthetic-cohort separation, z-score, ICC + WSCV math, tier
classification, verdict rollup, live intervention engine convergence detection +
JSON parsing, anomaly detector thresholds, etc.
├─ App/Views/
│   └─ BulkImportSheet.swift    # Multi-file drop, drag-reorder, live per-
file loader (§3.14)
│   └─ BatchAnalysisDetailView.swift # Series dashboard: trajectories,
reliability panel, anomaly density, talk-time, per-session list
│   └─ Federation/ProjectPickerMenu.swift # Header-bar pill + Settings row variant
(§3.15)
├─ chrome-extension/           # Cadence Bridge – MV3 extension; native-
messaging bridge to the daemon
├─ patent/                     # The provisional patent spec (PDF) + README
├─ docs/
│   └─ ARCHITECTURE.md
│   └─ METRICS.md              # The 26 measurements that feed the 12
IndicatorKinds, with computation + citation + robustness tag
│   └─ API.md                  # HTTP endpoint reference
│   └─ PRIVACY.md
│   └─ BUILD.md
│   └─ FIRST_RUN.md
│   └─ ROADMAP.md              # Deferred-but-spec'd: Zoom Video SDK, LTI
1.3, etc.
│   └─ WHITEPAPER.md           # this document
```

9. Prior art and what's novel

A patent attorney's first question is "what existed before and how is this different?" Cadence's novelty isn't in any single measurement — all the underlying algorithms are established. The novelty is in the **architectural combination** of measurements, reasoning, and distribution.

Novelty inventory across this document. Three locations carry overlapping but non-identical novelty content; the attorney's read path is:

- **§9 (here)** — behavioral-signal layer (capture → perception → fusion → indicators). This is what the existing provisional describes.
- **§6.5 + §1.5** — research-baseline audit subsystem (Auditor → Advocate → Judge → Skeptic → quarantine → fleet consensus). Structurally distinct invention sitting on top of the §9 layer; not in the current provisional.
- [/patent/README.md](#) — claim-element-to-code mapping for both layers in attorney-ready form, including Phase 4 fleet-consensus elements (i'), (l) – (p) that are not in the WHITEPAPER prose.

What existed before

Category	Representative tools	What they do
Recording-based engagement	Class.com, Echo360 engagement add-on, various LMS proctoring SKUs	Upload audio + video + screen to a server; run analytics centrally. Compromises FERPA / COPPA / EU AI Act because raw media leaves the device.
Single-channel voice analytics	Cogito (call-center voice tone), Kintsugi (depression voice signal)	Real-time prosodic measurement — <i>one modality</i> . Don't observe gaze, expression, or speech timing alongside.
Established FACS toolkits	OpenFace 2.0 (Baltrušaitis 2018), Affectiva / iMotions	Computer-vision libraries for facial action unit detection. Operate as raw measurement components, not as a reasoning + distribution layer; not BAA-backed; require uploads to vendor cloud.
Research-grade eye-tracking	Tobii, EyeLink	Lab-grade hardware. Not consumer-deployable. Cadence uses the same I-DT algorithm against consumer webcam frames.
Pitch / prosody analytics	Praat (free, research-standard), Carnegie Speech (commercial)	Acoustic analysis of recorded audio. No live integration with meetings, no cross-modal anything.
LMS analytics dashboards	Built-in Canvas / Brightspace / Blackboard "course analytics"	Login/click/time-on-page proxies. Don't measure behavior — measure account activity.
Healthcare AI clinical decision support	Multiple FDA-cleared SaMDs	Image-based or single-signal diagnostics. None measure live conversational behavioral signal cross-modally.
Hiring / HR video AI	HireVue, myInterview, Modern Hire	Post-hoc video analysis for hiring decisions. Cloud-only. HireVue was forced to drop facial analysis in 2021 after FTC scrutiny; still does speech-content + voice-pace. Cadence's non-diagnostic posture pre-dates that pressure; not a hiring product.
Single-vendor voice biomarkers	Ellipsis Health (FDA breakthrough designation), Aural Analytics (ALS / Parkinson's), Sonde Health (voice biomarkers), Winterlight Labs (speech dementia screening — part of Cambridge Cognition)	Voice-only screening for one named condition. Cadence is condition-agnostic, ≥3 modalities, clinician-facing — not consumer-diagnostic.
Passive behavioral inference	Mindstrong (defunct 2022, keystroke patterns), CompanionMx (smartphone call patterns), Microsoft Viva Insights (calendar metadata)	Device-interaction proxies — no audio/video. Cadence measures the actual conversational behavior, not its activity-log shadow.
Meeting-platform-agnostic transcription	Otter, Read.ai, Fireflies	Same Bridge-style "operates above any platform" distribution pattern (browser extension + native shim). Transcription only — no behavioral analytics, no cross-modal fusion, no on-device perception. Cadence is the behavioral-analytics layer they don't have.

Known active patents in the landscape (recommended USPTO search targets)

Holder	Representative patent / family	Relevance
HireVue	US10803751B2 and related family	"Video interview behavioral analysis." Highest-density patent landmines in the live-video + behavior space.
Affectiva (acquired by Smart Eye AB, 2021)	Extensive emotion-AI patent portfolio	FACS-and-emotion-from-video coverage; Cadence's differentiator is the combination with acoustic + speech timing + citation grounding, not the AU detection itself.
Cogito Corp	Several issued patents on real-time voice-tone analytics	Real-time voice prosody for contact-center applications. Single-modality.
Sonde Health	Voice-biomarker patents	The most-aggressive voice-IP holder in the space. Voice-only.
MIT Media Lab / Affective Computing Group	Picard et al. foundational patents	Mostly expired but verification recommended.
Realeyes / Emoshape / Eyeris	Emotion-from-video patents	Adjacent to Affectiva category.

A targeted prior-art search around the audit-arc subsystem (§6.5) is recommended on the terms documented in §6.5's "Terminology for prior-art search" paragraph: *distributed voting consensus system, Byzantine-resistant federated trust, self-auditing knowledge base, RAG corpus evolution, shadow-config quarantine, reproducibility stamping for grounded LLM systems*.

What Cadence does that none of the above does

The novelty claims below are ranked Tier 1 → Tier 3 by patent-defensibility. Tier 1 claims are the strongest standalone IP; Tier 2 are meaningful but more defensive; Tier 3 are best-claimed as part of the combination.

Tier 1 — strongest standalone IP

- 1. Real-time cross-modal convergence detection on consumer hardware** — the patent's central claim. Existing tools do one modality, or multiple modalities post-hoc on a recording, or one modality live. Cadence does ≥3 modalities live on the user's own Mac with no media upload. The triple-conjunction (≥3 modalities + live + on-device) is what's novel; each lever individually has prior art.
- 2. Citation-grounded reasoning with structural enforcement** — every observation cites a paper from a bundled library; the validator drops keys not in the library. The LLM **literally cannot** fabricate sources by construction. No other system in the category enforces this at the architecture layer (most rely on "the model says it cited X," which is unverifiable).
- 3. Perception is on-device by architectural default; cloud only sees derived 32-scalar feature windows** — the privacy-architecture claim. The network protocol forbids raw audio, video, or transcript content from leaving the device. It's not "we chose not to upload" — it's "the system has no code path to."
- 4. Cadence Bridge — platform-agnostic distribution** (§3.13). One code path covers Zoom Web, Google Meet, Teams Web, WebEx, BigBlueButton + any of those wrapped in any LMS iframe. Competitors pick one platform and ship a vendor-specific connector.

Tier 2 — meaningful but more defensive

- 5. HIPAA-BAA on every cloud hop** — signed BAA with AWS (infrastructure) and signed BAA with OpenAI (with zero-data-retention enforced on the per-session synthesizer + the cadence-chat tool dispatch). Two-vendor BAA coverage at this layer is uncommon.
- 6. Repeatability gating, not suppression** — Indicators with ICC < 0.75 (Fleiss threshold per Stegmann 2020) surface with a "low repeatability" chip, not hidden. Liss/Berisha 2024

"exploratory, not silenced" framing. Most healthcare AI ships indicators without exposing test-retest reliability at all.

7. **Self-auditing grounding corpus** — see §6.5 + </patent/README.md>. The whole audit-arc subsystem (Auditor → Advocate → Judge → Skeptic + gossip-over-S3 fleet consensus + per-finding [seed_overlay_version](#) provenance) is a structurally distinct invention. Multiple distinct claim candidates within it; Phase 4 is the highest-priority CIP work.

Tier 3 — best claimed as part of the combination

8. **Calibration / uncertainty / input-quality scores per indicator AND per session.** Maps to provisional §8. Three surfaces shipped recently fully implement this:
- [Indicator.confidenceTier](#) — derived high / moderate / low confidence rating combining 95% credible-interval width with worst-case repeatability ICC, rendered as a color-coded chip on every indicator card.
 - [Indicator.recommendations: \[Recommendation\]](#) — structured non-diagnostic follow-up actions ([repeatAssessment](#) , [offerResource](#) , [suggestAccommodation](#) , [flagForReview](#)) that the synthesizer emits when an indicator warrants one. Maps to provisional §7 "recommendations for follow-up actions."
 - [CadenceCore.SessionQualityScore](#) — system-computed input quality score per session combining duration, voiced fraction, face coverage, transcript word count, and HNR into a 0..1 numeric handle with a tier label. Rendered as a percentage headline on the session quality strip.

Why the combination is patentable even when no single component is

A common attorney attack on architectural claims is "this is just OpenFace + Whisper + GPT, no inventive step." The combination remains defensible because:

1. **The on-device/cloud split is architectural, not implementation detail.** Perception **cannot** upload media; the protocol forbids it. That's a runtime constraint, not a polite choice.
2. **Citation enforcement is structural, not prompt-level.** A fixed library is provided to the model and a validator drops anything outside it. Standard RAG asks the model nicely; Cadence enforces.
3. **Repeatability is gating, not decoration.** ICC values flow through to the UI as actionable tier chips. Most healthcare AI hides their test-retest numbers; Cadence makes them load-bearing.
4. **The combination has been demonstrated, not just claimed.** Implementation status tables in </patent/README.md> enumerate the shipped components. The system is operating, not theoretical.

The Tier 1 + Tier 2 combination as a whole — real-time cross-modal on-device perception + structurally enforced citation grounding + platform-agnostic distribution + repeatability gating + the audit-arc subsystem — has no single-vendor analog in the prior art identified above.

What's deliberately not novel (and shouldn't be claimed)

Trying to claim novelty on these specific measurements would lose on prior art:

Established measurement algorithms:

- YIN pitch detection (de Cheveigné & Kawahara 2002) — 22 years old, cited thousands of times
- FACS action units (Ekman & Friesen 1978) — 47 years old, the affective-computing standard
- I-DT fixation detection — decades old
- MATTR lexical diversity measure — Covington-McFall 2010
- LLMs generating clinical narrative — pre-empted by every healthcare AI patent filed 2022-2025

Adjacent academic work that frames the multimodal-fusion and self-critique problem space:

- Schuller et al. *Computational Paralinguistics* — pioneered the "multimodal frontier" framing for paralinguistic signal analysis. Already cited in [IndicatorKind](#) documentation for the [multimodalConvergence](#) indicator.
- Eyben 2016 (GeMAPS) — de-facto vocabulary for acoustic feature extraction.
- Cummins 2018 — speech-depression survey, standard reference for voice-based mental-health screening.

- **Self-RAG (Asai et al. 2023)** — model decides when to retrieve, can critique retrieved chunks. Operates on the *retrieval decision*, not on the underlying corpus. No federation, no quarantine queue, no per-finding provenance.
- **Constitutional AI (Anthropic)** — model critiques its own outputs against principles. Operates on *outputs*, not on a distributed grounding corpus. No federation, no peer consensus, no per-account capping.
- **Federated learning (Google)** — distributed model updates without centralizing training data. Distributes *training updates*; Cadence distributes *audit verdicts on a static corpus*. Different signal axis.

Established distributed-consensus literature (for the Phase 4 fleet consensus, §6.5):

- Byzantine Fault Tolerance / Paxos / Raft — solve "who is leader" / "is this transaction committed." Cadence's consensus is on scientific-research verdicts, not on a transaction log. The pure-function aggregator + S3-as-gossip-channel + no leader is the structural difference.
- Blockchain / proof-of-work / proof-of-stake — Cadence proves trust through *upstream identity* (pair-code redemption), not downstream computation. Per-account capping replaces stake-weighting structurally.
- CRDT (Conflict-free Replicated Data Types) — converge on data merges. Cadence converges on *judgments about scientific literature* with a 30-day vetoable timer between consensus-apply and effect.
- DAO governance / Snapshot.org — token-weighted on-chain voting. Cadence doesn't tokenize; the "vote" is an LLM verdict and the "voter" is a partner account scoped by an existing trust relationship.

The patent claims should center on the **architectural combination** (cross-modal convergence + citation grounding + on-device + platform-agnostic) and the specific **method of triggering cross-modal observations from a distinct-family convergence gate** (§3.12), plus the **self-auditing grounding corpus** with its Phase 4 gossip-over-S3 fleet-consensus extension (§6.5). The measurements themselves and the adjacent academic patterns above are the substrate, not the invention.

Appendix A — The 32 scalar features

Per [FeatureWindow](#) (30-second windows by default). All emitted to SQLite; the cloud reasoner sees these and only these. Source: [Sources/CadenceCore/Models.swift](#) .

AcousticFeatures (12 fields)

#	Field	What it measures	Unit	Citation
01	<code>f0Mean</code>	Mean fundamental frequency across voiced frames	Hz	Fusaroli 2017
02	<code>f0StdDev</code>	F0 variability — monotone speech signal	Hz	Fusaroli 2017
03	<code>f0Range</code>	Max – min F0 over the window	Hz	Diehl & Paul 2013
04	<code>rmsEnergy</code>	RMS audio amplitude (loudness floor)	float ~0–1	Eyben 2016 (GeMAPS)
05	<code>jitter</code>	Cycle-to-cycle pitch period perturbation	fraction	Bone 2014
06	<code>shimmer</code>	Cycle-to-cycle amplitude perturbation	fraction	Bone 2014
07	<code>hnr</code>	Harmonic-to-noise ratio	dB	Bone 2014
08	<code>speakingRate</code>	Voiced-segment onsets per second (syllable proxy)	onsets/sec	Diehl & Paul 2013
09	<code>voicedFraction</code>	Fraction of frames above voiced-energy threshold	0–1	Eyben 2016
10	<code>pauseCount</code>	VAD-detected pauses per window	Int	Parish-Morris 2016
11	<code>pauseMeanDuration</code>	Mean within-utterance pause duration	seconds	Parish-Morris 2016
12	<code>turnTakingLatencyMs</code>	Cross-stream VAD gap from remote-tap end to local-mic onset	ms	(methodological — derived)

VisualFeatures (12 fields)

#	Field	What it measures	Unit	Citation
13	<code>faceDetected</code>	Did Vision lock onto a face this window?	Bool	(gates below)
14	<code>gazeTowardCameraFraction</code>	Frames with frontal yaw/pitch	0–1	Klin 2002
15	<code>headYawStdDev</code>	Yaw variability	radians	Martin 2018
16	<code>headPitchStdDev</code>	Pitch variability	radians	Martin 2018
17	<code>headMovementRMS</code>	RMS of combined yaw/pitch deltas	radians/frame	Martin 2018
18	<code>auActivations</code>	FACS-13 AU intensities (dict of 13 AUs)	per-AU 0–1	Trevisan 2018 (contested per Milton 2012)
19	<code>expressionEntropy</code>	Shannon entropy over AU activations	bits	Trevisan 2018
20	<code>meanFixationDurationMs</code>	I-DT-detected fixation length	ms	Klin 2002, Sasson 2008
21	<code>saccadeRatePerSecond</code>	Velocity-threshold saccade count	/sec	Tseng 2013, Hutton 1998
22	<code>gazeEntropy</code>	Shannon entropy over 4x4 grid	bits (max 4)	Tseng 2013
23	<code>blinkRatePerMinute</code>	Runs of low-aperture frames	/min	(multiple, methodological)
24	<code>meanEyeOpenness</code>	Mean vertical eye spread, normalized	0–1	(supporting)

LinguisticFeatures (8 fields) — opt-in via SpeechAnalyzer

#	Field	What it measures	Unit	Citation
25	<code>wordCount</code>	Words transcribed this window	Int	(infrastructural)
26	<code>typeTokenRatio</code>	Unique/total words (raw)	0–1	Losh & Capps 2003
27	<code>mattr</code>	Moving-average TTR, 50-word window	0–1	Losh & Capps 2003
28	<code>disfluencyCount</code>	"Um/uh/like/y'know" markers	Int	(Gorman et al.)
29	<code>repetitionScore</code>	Repeated-trigram fraction	0–1	Tang 2024
30	<code>meanUtteranceLength</code>	Words per utterance (500 ms pause break)	words	(developmental standard)
31	<code>transcript</code>	Raw transcript text	String	(infrastructural)
32	<code>wordTimings</code>	Per-word timings + speaker tags	<code>[TranscribedWord]?</code>	(infrastructural)

Note on count. "32 fields" counts struct properties. The `auActivations` field is a dictionary containing the FACS-13 — AU1, AU2, AU4, AU5, AU6, AU9, AU10, AU12, AU15, AU17, AU20, AU25, AU26 — so expanded, the per-window total is 44 numbers, but the wire format is 32 fields.

Appendix B — The 8 specialist agents

Defined server-side in

`backend/api/client/services/openai/orchestrators/cadence/specialists.js` ; per-specialist allowed-indicators + max-output-tokens + model overrides are editable per-deploy via the `system_settings.rules.cadence_agents` configuration block. Wave-parallel execution; Synthesizer closes. The Mac client sees each specialist arrive as a streaming `agent_complete` SSE frame from `OpenAIReasoner` and persists it as an `AgentFinding` keyed by `agentName` .

Agent	Role	Input scope	Output
AcousticAgent	Acoustic specialist	<code>acoustic</code> block of FeatureWindows + RuleBasedReasoner prior for acoustic indicators + acoustic-tagged research citations	Per-modality findings: scored acoustic indicators with citation anchors
GazeAgent	Visual / gaze specialist	<code>visual</code> block including FACS-13 + per-modality citations	Gaze + facial findings, paired with the double-empathy critique per Milton 2012 / Crompton 2020 to keep framing non-deficit
LinguisticAgent	Transcript-derived features	<code>linguistic</code> block (only when SpeechAnalyzer is on) + linguistic citations	MATTR, repetition, disfluency findings grounded in quotes from <code>wordTimings</code>
TemporalAgent	When things happen	Cross-stream VAD timing data, turn-taking latency series, talk-time evolution	Time-anchored moments: "at 4:22 the pause clusters tighten..."
BaselineAgent	Longitudinal / personal	The user's <code>CalibrationStore</code> (Rainbow / Grandfather Passage) + prior session history	Per-user deltas — explicitly compares to <i>self</i> , never population norms (those are other agents' job)
CitationAgent	Research grounding	All sub-agent findings + the full bundled research set	Verifies every claim has a citation key from the loaded library; flags ungrounded claims as exploratory
ClinicalFramingAgent	Responsible-output specialist	All sub-agent findings + the CitationAgent's verifications	Rewrites findings into non-diagnostic language; refuses any framing that resembles diagnosis
Synthesizer	Cross-modal late fusion	All 7 sub-agents' findings + full FeatureWindows + research corpus + RuleBasedReasoner prior	Final <code>[Indicator]</code> + <code>[Moment]</code> set. Explicitly instructed: surface <code>multimodalConvergence</code> when it's real; this is the patent's frontier framing.

Wave ordering: { Acoustic, Gaze, Linguistic, Temporal, Baseline } in parallel → CitationAgent + ClinicalFramingAgent in parallel on merged findings → Synthesizer closes.

Appendix C — Patent claim → code → behavior (forensic mapping)

Each row maps a provisional-spec phrase or claim element to the concrete code path that implements it and the demonstrable behavior the engineering produces. Patent attorneys can verify any row by reading the cited code.

Provisional claim phrase	Code path	Demonstrable behavior
"Receive behavioral signal data from a host platform"	<code>Sources/CadenceDaemon/DaemonCoordinator.swift::handleDetection</code> + <code>Sources/CadenceCore/Capture/CaptureCoordinator.swift</code>	Detect meeting → r mic/cam/screen/ren
"Extract multimodal behavioral features"	<code>Sources/CadenceCore/Perception/PerceptionPipeline.swift::flush</code>	Emits a <code>FeatureWindow</code> struct documented
"Operates independent of any specific video or LMS platform"	<code>chrome-extension/src/content/injected.ts</code> + <code>chrome-extension/manifest.config.ts</code>	Single content scrip <code>true</code> ; hooks <code>getL</code> code works in Goog Whereby, Jitsi, and Brightspace / Mood
"Cross-modal temporal co-occurrence"	<code>Sources/CadenceCore/Reasoning/LiveInterventionEngine.swift::evaluate</code>	Fires only when ≥3 interaction / linguist same window — dis
"Reasoning anchored to peer-reviewed literature"	<code>Sources/CadenceCore/Library/DocumentLibrary.swift::search</code> + <code>LiveInterventionEngine.retrieveCitationContext</code>	Indicator-tag-index (search). The model keys from the loaded resolvable
"Real-time perception executed locally on the user's device"	<code>Sources/CadenceCore/Perception/AcousticFeatureExtractor.swift</code> + <code>Sources/CadenceCore/Perception/GazeAndFaceService.swift</code>	YIN pitch via <code>Acce</code> <code>VNDetectFaceLar</code> SpeechAnalyzer (m perception
"Reasoning available without network connectivity"	<code>Sources/CadenceCore/Reasoning/FoundationModelsReasoner.swift</code>	<code>SystemLanguageModel</code> <code>LanguageModelLS</code> reasoner; cloud pat
"Verifiable against established norms"	<code>Sources/CadenceCore/Reasoning/RuleBasedReasoner.swift</code> + <code>Sources/CadenceCore/Resources/normative_bands.json</code>	Every session comp LLM gets this as a s disagreement explic
"Reproducible measurement across model generations"	<code>Sources/CadenceCore/Models.swift::Indicator.generation</code> + <code>Sources/CadenceCore/SessionReprocessor.swift</code>	Every indicator carr appends, never ove
"Parallel specialist analysis with reconciled synthesis"	<code>backend/api/client/services/openai/orchestrators/cadence/handlers.js</code> (Wave 1 + Wave 2 + Synthesizer dispatch); Mac-side consumption in <code>Sources/CadenceCore/Reasoning/OpenAIReasoner.swift::reasonWithFindings</code>	Wave-parallel execut Synthesizer adjudic <code>agent_complete</code>
"Compositional anomaly detection across signal types"	<code>Sources/CadenceCore/Perception/PerceptionPipeline.swift::AnomalyKind</code>	6 independent dete gazeAwayEnd, blink toggleable; per-det
"Derived behavioral features as the wire-format unit"	<code>Sources/CadenceCore/Reasoning/OpenAIReasoner.swift::buildRequestPayload</code>	The cloud reasoner <code>FeatureWindow</code> : repeatability hints + frame metadata. No content unless expl

Provisional claim phrase	Code path	Demonstrable behavior
"Aggregation under structural anonymity guarantees"	<code>Sources/CadenceCore/Federation/</code> + <code>Sources/CadenceCore/Fingerprint*</code>	Salted SHA-256 fingerprint, k=5 (suppresses aggregation; salt never leaves the host)
"Active and passive collection modes" (provisional Fork 9)	Active: <code>Sources/CadenceCore/Calibration/CalibrationStore.swift</code> (Rainbow / Grandfather Passage recording — user performs a scripted task for the system). Passive: live <code>CaptureCoordinator</code> + retrospective <code>DryRunImporter</code> / <code>BulkDryRunImporter</code> (system observes naturally-occurring behavior, whether live or time-shifted)	Both modes terminate; <code>ReasoningContext</code> collection time-shifted
"Source-agnostic passive ingestion (live or retrospective)"	<code>Sources/CadenceCore/DryRun/DryRunImporter.swift</code> + <code>BulkDryRunImporter.swift</code>	The perception → reality gap: audio/video AND perception; no divergent "impossible" <code>FeatureWindow</code> ; identical orchestration bounded by <code>bulkCrossSessionSynthesizer</code>
"Longitudinal trajectory across sessions"	<code>Sources/CadenceCore/DryRun/CrossSessionSynthesizer.swift</code>	Deterministic per-lr outlier flagging across over the rollout (gate <code>bulkCrossSessionSynthesizer</code> flippable); reproducibility same prompt
"Confidence / uncertainty / explainability"	<code>Sources/CadenceCore/Models.swift::Indicator</code> (<code>confidenceLow</code> , <code>confidenceHigh</code> , <code>bandZScore</code> , <code>bandCitationKey</code> , <code>drivers</code> , <code>reasoning</code> , <code>generation</code>)	Every indicator carries an anchor it was scored against; human-readable reasoning
"Non-diagnostic indicators inside the host workflow"	<code>Sources/CadenceCore/Reasoning/Prompts.swift::systemPrompt</code> + <code>ClinicalFramingAgent</code>	Hard constraint in the host as an adversarial check: "Not a diagnostic data point"