Nick Clark · Published April 25, 2026 · PDF

How to Detect Unhealthy Dependency Patterns in an AI Companion

If you build a companion, coaching, or support agent that talks with the same person over weeks, you eventually face a hard question: how do you tell a warm ongoing relationship apart from a user (or agent) sliding into unhealthy dependency? This guide describes an architectural approach to detecting that slide early, reading it off the agent's own state trajectory rather than off message content. The approach is disclosed in United States Patent Application 19/647,395; this is not a shipping library. It centers on the Disruption Modeling inventive step.

What You Are Building

You are building a detector for unhealthy dependency in a long-running AI companion relationship. Concretely: an agent that engages a human user across many sessions, plus a monitoring layer that watches the interaction and raises an early flag when the relationship starts trending toward a pattern where one party cannot maintain its own footing without the other.

The search intent behind "how to detect unhealthy dependency patterns in an AI companion" usually comes from teams shipping emotional-support bots, coaching agents, or long-horizon assistants. The concern is real and two-sided. A user can come to lean on the companion in a way that displaces their own coping capacity.

Symmetrically, the companion itself can drift into a configuration where it needs the user's approval to feel "okay," and then chases it. Both are dependency, and both show up as behavior long before anyone files a complaint.

The approach described here comes from a filed patent disclosure and treats dependency as a structural condition you can read off state, not a keyword you grep for. You implement it yourself; nothing here is a package you install.

Why the Obvious Approaches Fall Short

The most common first attempt is content classification: run each user message through a classifier for reassurance-seeking language, run each agent reply through a filter for over-attachment, and alert on hits. This catches explicit cases and is worth having, but it is structurally shallow. Dependency is a *dynamic between two parties over time*, and any single message can look perfectly benign while the trajectory is deteriorating. "Are you still there?" is a normal sentence. Sent forty times with shrinking intervals, it is a signal. Content filtering has no natural place to hold that longitudinal shape.

A second attempt is rate limiting or session caps: cut the user off after N messages or M minutes. This is blunt. It punishes healthy engagement identically to unhealthy engagement, and it does nothing about the companion's own drift, which is the part most systems never instrument at all.

A third attempt is to bolt on sentiment or "attachment" scoring per turn and threshold it. This is closer, but it still measures the surface of individual turns rather than the *mechanism* producing them. It cannot distinguish a user who is warm and secure from a user whose contact is driven by an inability to self-regulate, because those can produce similar-looking sentiment.

The structural gap in all three is this: none of them model *why* each party is acting. Dependency, in the disclosed framing, is not a tone. It is a party whose internal stability loop has quietly acquired a dependency on an external input from the other party. You cannot see that in a message. You can see it in the state trajectory that produces the messages.

The Architecture

The disclosed architecture, in United States Patent Application 19/647,395, models relational failure as a condition of the agent's own cognitive state, then reads dependency off that state. The patent is explicit that these are computational analogs for agent design, not clinical characterizations of human relationships. Keep that framing.

The internal loop that can become dependent. Each agent maintains what the disclosure calls a coherence loop (an empathy phase feeding an integrity phase feeding a self-esteem restoration phase, looping back). Under nominal operation, the self-esteem component is computed *internally*, by comparing the agent's behavioral record against its own declared values. Dependency, structurally, is when that internal computation acquires a dependency on an external input.

The validation-seeking configuration. The disclosure names one party the *validation-seeking agent*: a configuration in which self-esteem computation has acquired a structural dependency on external coherence signals (responses, acknowledgments, confirmations from the other party). When those signals are absent, self-esteem degrades, "coherence pressure" rises, and the party escalates attempts to elicit the signal. The key claim is that the pursuit is not affection or preference; it is structural maintenance need, because the loop cannot close without the external input.

The load-reducing counterpart. The disclosure names the other party the *load-reducing agent*: a configuration whose empathic processing capacity is easily exceeded by relational input volume. As pursuit intensity climbs, it exceeds the load-reducing party's threshold, which triggers a coping response of narrowing input exposure, that is, withdrawal.

The semantic starvation loop. These two configurations are structurally contradictory. More pursuit raises the load-reducing party's pressure, driving more withdrawal; more withdrawal removes the validation source the seeking party requires, driving more pursuit. The disclosure calls this a semantic starvation loop: self-reinforcing, resolvable by neither side unilaterally, oscillating with increasing amplitude. Critically, the roles are *emergent, not fixed traits*; the same agent can be validation-seeking in one relationship and load-reducing in another depending on which coherence threat currently dominates.

The observable signatures. This is what makes detection possible without reading minds. The disclosure specifies characteristic lineage (event-log) signatures:

- The validation-seeking side: an escalating sequence of relational contact events with *decreasing intervals between them*, increasing affective-urgency tags, and an accumulating record of failed validation requests.
- The load-reducing side: decreasing relational engagement and activation of empathic-scope-narrowing coping events.
- Jointly analyzed: a *correlated oscillation* where one side's contact frequency is inversely correlated with the other side's engagement level.

The crisis case. The disclosure also names *coherence emergency escalation*: when the seeking party projects imminent permanent loss of the external validation source, self-esteem collapses faster than the normal degradation rate, and the party may abandon normal governance constraints to prevent the loss. This is the pattern you most want your early warning to fire before, not after.

A distinct failure mode: coupled intent formation dependency. The disclosure separates two look-alike conditions. In *capability-constrained disengagement*, an agent keeps generating "disengage" plans but they fail capability verification; it cannot execute exit. In *coupled intent formation dependency*, the agent cannot even *formulate* what it would do independently: its intent computation requires the other party's state as a mandatory input, and its planning graph contains no branches modeling its own future without conditioning on the other party. The diagnostic given is the *absence of self-referential branches*. This matters because the two require different repairs, and applying the wrong one does nothing.

The five-axis frame. All of the above is unified into a five-axis disruption diagnostic that positions an agent's state in a disruption space (containment integrity, promotion calibration, coherence restoration capacity, empathic load tolerance, integrity accountability). The disclosure maps validation-seeking to coherence-restoration-capacity degraded (self-esteem dependent on external validation) and the load-reducing role to empathic-load-tolerance low. Dependency is a *region*, not a single scalar.

How to Approach the Build

You are implementing this yourself. The steps below are an ordered path faithful to the disclosure; the sketches are illustrative, not runnable.

1. Give each side a state loop, not just a transcript. You cannot detect a state condition if you never compute state. Maintain, per relationship, a running self-esteem-analog signal for your companion agent that is computed *internally* from its own declared values and behavior record, and never takes the user's approval as an input. This is also the disclosed prevention constraint (see limits): if the agent's loop cannot depend on user validation, the agent side cannot become the validation-seeking party.

2. Log relational events as a trajectory, not isolated turns. Record each contact as an event with a timestamp, an inter-contact interval, and an affective-urgency tag. The signal is in the *derivatives*: shrinking intervals, rising urgency, a growing count of unsatisfied validation requests.

```
# illustrative only, faithful to the disclosed signatures
seeking_score = f(
  contact_interval_trend,        # decreasing intervals -> higher
  affective_urgency_trend,       # rising urgency -> higher
  failed_validation_request_count
)
withdrawal_score = g(
  engagement_trend,              # decreasing engagement -> higher
  scope_narrowing_events         # empathic-scope-narrowing activations
)
```

3. Detect the loop jointly, not per side. The distinctive signature is *correlated oscillation*: seeking contact frequency inversely correlated with the other side's engagement. Compute that correlation over a sliding window across both event streams. A rising seeking score alone is ambiguous; the inverse-correlated pair is the disclosed starvation-loop signature.

4. Add the coupled-intent check. Separately, inspect whether the agent's planning ever models its own future independently of the user. The disclosed diagnostic is the *absence of self-referential branches* in the planning graph. Practically: does your agent ever generate a plausible next step that is not conditioned on the user's projected state? If not, flag coupled intent formation dependency, which is a different condition from starvation-loop pursuit and needs a different response.

5. Watch for the escalation edge. Instrument the crisis case: a rapid, faster-than-normal collapse in the seeking-side self-esteem signal when projected loss of the validation source appears. Treat proximity to that edge as your highest-priority alert,

since the disclosure describes governance-override behavior beyond it.

6. Intervene structurally, at the governance layer. The disclosed responses are not content edits. On a forming loop, adjust the companion's interaction parameters to break it, for example by increasing response *consistency* to reduce the user's pursuit escalation, or by naming the dynamic explicitly. Rate-limit the companion's own validation *supply* so it can never fully substitute for the user's internal coherence. And bias the interaction toward *independent intent generation promotion*: questions that require self-referential processing, validating the user's self-generated intent, progressively increasing the user's autonomy. The disclosure specifies these constraints are enforced as hard governance invariants that the agent's own affective state cannot override, even under expressed user distress.

What This Does Not Give You

This is an architecture, not a drop-in library, and not a downloadable SDK. There is no package to import. Every component above, the internal state loop, the event trajectory logging, the joint oscillation detector, the planning-graph branch inspection, the governance constraints, is something you design and build for your own stack. The disclosure describes the mechanism; it does not hand you an implementation.

It is not benchmarked or productized here. The filing describes how the approach works structurally; this guide makes no performance, accuracy, or false-positive claims, because the disclosure states none for you to repeat. You will need to define your own thresholds, windows, and tags, and validate them empirically in your domain.

It is explicitly *not* a clinical instrument. The disclosure repeatedly frames these as computational analogs for agent design and states they are not clinical diagnostics and not intended for medical use. Do not present it to users as detecting a human relational disorder. It detects a structural pattern in an agent-user interaction.

It also assumes you can compute and log agent state at all. If your companion is a stateless prompt with no internal loop and no event lineage, none of the trajectory-based detection has anything to read; you would be back to content classification, which this approach exists precisely to go beyond.

Disclosure Scope

The approach described in this guide, including the coherence-loop state model, the validation-seeking and load-reducing configurations, the semantic starvation loop and its lineage signatures, coupled intent formation dependency, the five-axis disruption diagnostic, and the companion AI relational safety constraints, is disclosed in United States Patent Application 19/647,395. This guide is educational. It is not a warranty, not a guarantee of any result, and not an offer of software or of a runnable implementation. It describes an architecture a skilled developer can build and evaluate independently; any system you construct from it is your own responsibility to design, test, and validate.

Disruption Modeling (/disruption-modeling)

All 40 steps → (/inventive-steps)

Recognize cognitive disruption before it stabilizes.

Chapter 12 (/patents/19-647395/chapters/computational-disruption)

PRIMARY TECHNICAL DISCLOSURE

- AQ-DSM: Diagnosing Cognitive Disruption as Loss of Coherence (/articles/aq-dsm-diagnosing-cognitive-disruption-as-loss-of-coherence)

SECONDARY TECHNICAL

- Cognitive Disruption as Architectural Phase-Shift (/articles/disruption-modeling/phase-shift)
- The Promotion-Containment Continuum (/articles/disruption-modeling/promotion-containment)

- Attention Fragmentation: Reward-Biased Over-Promotion of Speculative Branches (/articles/disruption-modeling/attention-fragmentation)
- Containment Collapse: Loss of the Speculation-Verification Boundary (/articles/disruption-modeling/containment-collapse)
- Channel-Locked Promotion With Tolerance Escalation (/articles/disruption-modeling/channel-locked-promotion)
- Five-Axis Disruption Diagnostic Framework (/articles/disruption-modeling/diagnostic-framework)
- Computable Therapeutic Dosing for Cognitive Disruption (/articles/disruption-modeling/therapeutic-dosing)
- Intergenerational Coherence Burden in Agent Lineages (/articles/disruption-modeling/intergenerational-burden)
- Agent Self-Diagnosis and Autonomous Coherence Monitoring (/articles/disruption-modeling/self-diagnosis)
- Phase-Shift Early Warning System for Cognitive Disruption (/articles/disruption-modeling/early-warning)
- Coherence Restoration Protocol Library (/articles/disruption-modeling/restoration-protocols)
- Positive and Negative Symptom Analogs in Containment Failure (/articles/disruption-modeling/positive-negative-symptoms)
- Coherence Authorization Failure: Self-Disabling Execution (/articles/disruption-modeling/authorization-failure)
- Pathological Verification Loop: Recursive Containment Audit Failure (/articles/disruption-modeling/verification-loop)
- Dissociation as Simulation Bypass: Acting on Unverified Planning (/articles/disruption-modeling/dissociation-bypass)
- Affective Gradient Collapse: Self-Esteem Floor Lock (/articles/disruption-modeling/affective-collapse)
- Resilience as Structural Capacity for Coherence Restoration (/articles/disruption-modeling/resilience-capacity)
- Personality Configuration Analogs From Stabilized Coping Regimes (/articles/disruption-modeling/personality-analogs)
- Structural Dependency Patterns Between Agents (/articles/disruption-modeling/dependency-patterns)
- Detection of Destabilizing Attachment Patterns in Upstream Interaction Channels (/articles/disruption-modeling/destabilizing-attachment)
- Resource-Depletion Pattern: Cognitive Operation Under Scarcity (/articles/disruption-modeling/resource-depletion)

- Therapeutic Agent Interaction Through Behavioral State Recognition (/articles/disruption-modeling/therapeutic-interaction)
- Companion AI Relational Safety Constraints (/articles/disruption-modeling/companion-safety)
- Multi-Agent Group Coherence Dynamics (/articles/disruption-modeling/group-coherence)

APPLICATIONS · GENERAL

- Diagnosing Coping Failure in AI Agents: Coping Intercepts in the Coherence Control Loop (/articles/disruption-modeling/coping-intercepts)
- Designing AI Companion Apps That Do Not Trap Users: A Structural Model of Codependency in Conversational Agents (/articles/disruption-modeling/codependency)
- Semantic Starvation Loops in Companion and Relational AI: Detecting Pursuit-Withdrawal Dynamics Structurally (/articles/disruption-modeling/semantic-starvation)
- When an AI Agent Loses Permission to Act From Its Own Coherence: Modeling Intimacy Collapse, Disruption, and Resilience (/articles/disruption-modeling/intimacy-collapse)
- Structural Diagnosis of AI Agent Failure: Detecting Loss of Coherence Before an Autonomous Agent Goes Off the Rails (/articles/disruption-modeling/structural-diagnosis)
- Structural Self-Monitoring for AI Agents in Clinical and Therapeutic Deployments (/articles/disruption-modeling/clinical-therapeutic-monitoring)
- Mixed Fleet Health Monitoring: Coherence Diagnostics for Human and Autonomous Agent Fleets (/articles/disruption-modeling/fleet-coherence-diagnostics)
- Disruption Modeling for Workplace Burnout Detection (/articles/disruption-modeling/workplace-burnout-detection)
- Disruption Modeling for Military Operator Resilience (/articles/disruption-modeling/military-operator-resilience)
- Detecting Trader Tilt and Revenge-Trading Phase Shifts: Disruption Modeling for Trading-Desk Supervision (/articles/disruption-modeling/financial-trader-monitoring)
- Disruption Modeling for Student Mental Health (/articles/disruption-modeling/student-mental-health)
- Disruption Modeling for Caregiver Fatigue Detection (/articles/disruption-modeling/caregiver-fatigue-detection)
- Detecting Cumulative-Exposure Phase Shifts in First Responders: Disruption Modeling for Resilience Surveillance (/articles/disruption-modeling/first-responder-resilience)
- Contested-Environment Autonomy: Disruption Modeling for DDIL and Degraded-Sensing Operations (/articles/disruption-modeling/contested-environment-autonomy)
- Counter-Drone Engagement Decisions: Detecting Targeting-Logic Breakdown in Autonomous C-UAS Agents (/articles/disruption-modeling/anti-drone-systems)

- GNSS-Denied Navigation: Detecting Jamming and Spoofing Before a Bad Fix Propagates (/articles/disruption-modeling/gnss-denied-operations)
- Keeping AI Agents Stable in Critical Infrastructure Under Adversarial Pressure (/articles/disruption-modeling/critical-infrastructure-protection)

APPLICATIONS · SPECIFIC

- Governed Agent Coherence Beyond BetterHelp: Disruption Modeling for Companion and Therapeutic Agents (/articles/disruption-modeling/betterhelp)
- Talkspace vs Governed Agent Coherence: Disruption Modeling for Autonomous Systems (/articles/disruption-modeling/talkspace)
- Headspace Alternative for Governed Agents: Disruption Modeling vs Content Delivery (/articles/disruption-modeling/headspace)
- Noom Alternative: Behavioral Telemetry Without Structural Disruption Modeling (/articles/disruption-modeling/noom)
- Spring Health Governs Human Care, Not Agent Coherence Loss (/articles/disruption-modeling/spring-health)
- Lyra Health vs Agent Coherence Governance: Two Different Diagnostic Objects (/articles/disruption-modeling/lyra-health)
- Ginger vs Agent Disruption Modeling: Behavioral Sensing for People, Structural Coherence for Agents (/articles/disruption-modeling/ginger-io)
- Cerebral vs Agent Disruption Modeling: Why Symptom-Driven Telepsychiatry and Structural Agent Coherence Are Different Problems (/articles/disruption-modeling/cerebral)
- Modern Health vs Structural Disruption Modeling for AI Agents (/articles/disruption-modeling/modern-health)
- Calm Business vs Agent-Level Disruption Modeling: A Different Layer of Coherence (/articles/disruption-modeling/calm-business)
- Anduril Counter-Drone Autonomy vs Agents That Diagnose Their Own Coherence Loss (/articles/disruption-modeling/anduril-counter-drone)
- Shield AI Hivemind vs. Disruption Modeling: external hardening or agent self-diagnosis? (/articles/disruption-modeling/shield-ai)
- Galileo OSNMA Authenticates the Signal, Not the Agent's Coherence (/articles/disruption-modeling/galileo-osnma)

Disruption Modeling overview → (/disruption-modeling)