TalkCatcher

Clinical & Technical Specification

28 Algorithms · 7 Pillars · 3-Layer Self-Improving Engine

Marina (Alex)[1]

BCN · PhD · former Harvard GSE Visiting Scholar (Fischer Lab)

[1] *Boston Neuromind LLC, Canton, MA, USA*

Version 1.0 · May 13, 2026

Abstract

TalkCatcher is a diagnosis-specific AI companion system for adult ADHD, designed to be replicated across nine additional DSM-5-TR conditions through a plugin architecture. This document specifies the complete technical architecture: a three-layer self-improving engine, twenty-eight specialized algorithms organized into a four-stage pipeline, seven proprietary clinical IP pillars, and an auto-growth loop that allows the vault to grow from real-world usage without retraining the underlying large language model.

The system is being developed by a single founder/clinician with formal training in instructional design, mental health counseling, and dynamic skill theory (Harvard GSE, Fischer Lab). Unlike general-purpose conversational AI, TalkCatcher's clinical content is curated, its retrieval is grounded in a clinician-authored vault rather than the LLM's parameters, and its safety overrides operate independently of the language model itself. Eight USPTO provisional patent specifications cover the most novel components.

This document is intended for clinicians, researchers, regulators, investors, and other technical readers who need the full algorithmic specification. A separate non-technical document is available for end users.

Contents

1. Introduction and Scope

1.1 The Problem

General-purpose conversational AI systems applied to mental health face three recurring failures. First, they cannot distinguish between disorders that look similar at the surface level but require opposite interventions (reassurance helps anxiety; reassurance worsens OCD). Second, their content is generated rather than curated, meaning clinical reliability cannot be guaranteed. Third, they have no built-in safety override that operates independently of the language model, so adversarial inputs can bypass safety entirely.

Diagnosis-specific systems (e.g., Inflow for ADHD, NOCD for OCD, Wysa for anxiety/depression) address some of these problems by narrowing scope, but each company has built only one such system. No company has built a unified architecture capable of supporting many diagnosis-specific bots from a single engine.

1.2 The TalkCatcher Approach

TalkCatcher addresses these failures by separating the engine from the disorder. The engine — 28 algorithms organized into a four-stage pipeline — is diagnosis-agnostic. Each disorder is a plugin: a YAML manifest, a vault of clinician-authored content, a set of validated clinical scales, and a protocol for sequencing interventions. Swapping the plugin swaps the system from an ADHD companion to an OCD companion without changing the engine.

The system is being built by one founder/clinician (Marina), one user/beta-tester/clinical supervisor (Marina's partner), with the help of automated synthetic data generation and a large language model API. Total capital required is on the order of ten thousand US dollars — three orders of magnitude lower than comparable VC-funded competitors.

1.3 Scope of This Document

This document specifies what the system is and how it works algorithmically. It does not cover the user-facing product (which has its own design document) or the business strategy (separate). It is intended to be read by people who want the technical detail without the marketing.

2. System Architecture — 3-Layer Self-Improving Engine

The complete system is organized into three layers, each operating at a different time scale. Layer 1 generates synthetic vault content offline. Layer 2 is the runtime engine that handles user conversations. Layer 3 grows the vault from real usage data, closing the loop.

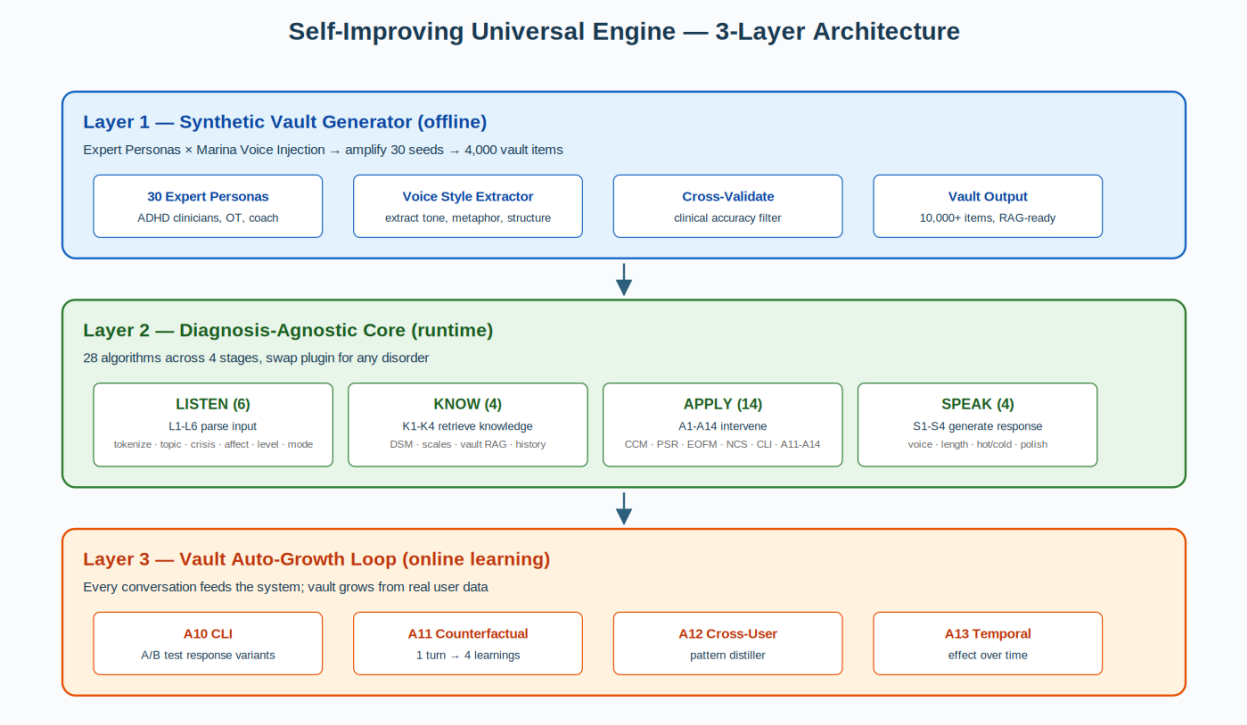

Figure 2. The three-layer self-improving engine.

2.1 Layer 1 — Synthetic Vault Generator (offline)

Before any user conversation, a generator pipeline produces vault content. Thirty expert personas (clinicians, occupational therapists, coaches, lived-experience contributors) are simulated by an LLM. Marina's voice pattern is extracted from a small number of seed samples (currently five), then injected into the simulated personas. The system cross-validates output for clinical accuracy and merges only validated items into the vault. Seed inputs of approximately thirty hand-curated samples amplify to roughly four thousand vault items at moderate quality, climbing toward ten thousand at production quality.

2.2 Layer 2 — Diagnosis-Agnostic Core (runtime)

The runtime engine handles each user message through twenty-eight algorithms in four stages (LISTEN, KNOW, APPLY, SPEAK). All twenty-eight algorithms are diagnosis-agnostic; their behavior is parameterized by the active plugin's YAML configuration files. Three files govern the swap: registry.yaml turns individual algorithms on or off, book_library.yaml manages knowledge

sources, and the plugin_loader merges the disorder-specific vault into the retrieval pool at runtime.

2.3 Layer 3 — Vault Auto-Growth Loop (online learning)

After each user conversation, four algorithms (A10 CLI, A11 Counterfactual, A12 Cross-User Pattern Distiller, A13 Temporal Effect Tracker) extract signal for vault improvement. The signal updates weights on existing vault items and proposes new items for human review. The language model itself is not retrained. This is critical: clinical safety can be guaranteed because the changing component (the vault) is human-reviewable; the LLM remains stable.

3. The 4-Stage Pipeline

Every user message is processed through four sequential stages.

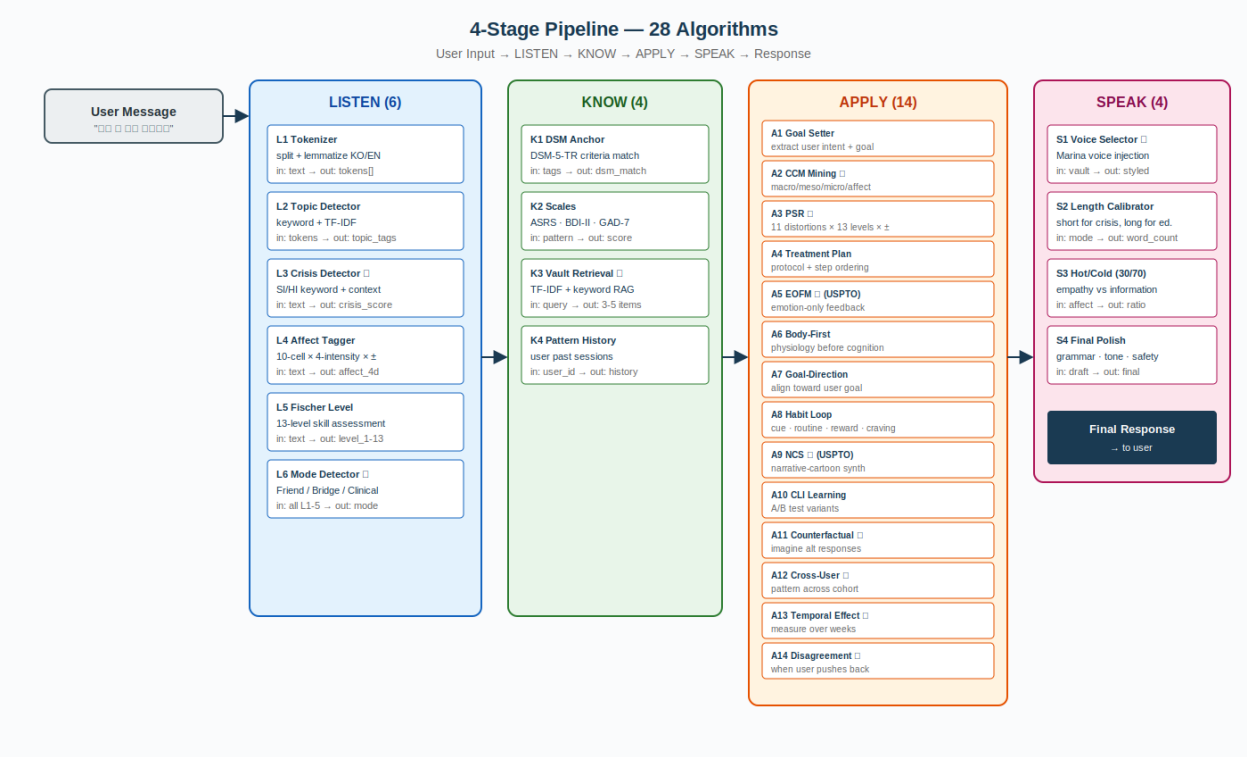

Figure 3. The complete 4-stage pipeline with all 28 algorithms.

LISTEN parses the input. KNOW retrieves relevant knowledge. APPLY selects and executes interventions. SPEAK generates and polishes the response. Each stage has a defined input/output contract; algorithms within a stage can be added or replaced without affecting other stages.

4. The 28 Algorithms — Detailed Specification

4.1 LISTEN Stage

L1 — Tokenizer [LISTEN] [V14 ✅]

Lemma-aware tokenizer that handles Korean and English in a unified token stream. Korean uses a stem extractor (drops particles 은/는/이/가, verb endings -다/-요/-습니다). English uses a Porter-style stemmer (-ing/-ed/-s).

Inputs

- raw_text: string (Korean, English, or mixed)

Outputs

- tokens: Array<{lemma: string, lang: 'ko'|'en', pos?: string}>

Pseudocode

```
function tokenize(text):
    lang = detect_language(text)             # ko / en / mixed
    if lang == 'ko':
        words = split_by_whitespace_and_punct(text)
        lemmas = [strip_particle_and_ending(w) for w in words]
    elif lang == 'en':
        words = split_by_whitespace(text.lower())
        lemmas = [porter_stem(w) for w in words]
    else:  # mixed
        lemmas = lemmas_ko + lemmas_en   # token by token
    return lemmas
```

Clinical Rationale

ADHD users frequently code-switch and type rapidly with partial words. A uniform lemma representation is required for downstream retrieval (K3) and topic detection (L2). Without lemmatization, '약속을' and '약속이' look like different tokens and break vault search.

References: *Manning, C. & Schütze, H. (1999). Foundations of Statistical NLP.*

L2 — Topic Detector [LISTEN] [V14 ✅]

Tags the message with one or more topic labels (work, family, romance, finance, health, school, friend, self, existential, somatic) using keyword anchors plus TF-IDF over the user's recent history.

Inputs

- tokens: TokenList
- user_recent_history: Array<TokenList> (last 10 turns)

Outputs

- topics: Array<{label: string, confidence: float}>

Pseudocode

```
function topic_detect(tokens, history):
    keyword_hits = match_against_topic_anchors(tokens)
    tfidf_vec = compute_tfidf(tokens, history)
```

```
    candidates = []
    for topic in TOPIC_SET:
        score = 0.6 * keyword_hits[topic] + 0.4 * cosine(tfidf_vec,
topic_centroid[topic])
        if score > THRESHOLD: candidates.append((topic, score))
    return sort_desc(candidates)[:3]
```

Clinical Rationale

Same word can mean different things in different contexts ('boss' is work; 'old boss' could be existential). Combining lexical anchors with TF-IDF context handles this without expensive LLM calls per turn.

References: *Blei, D. (2012). Probabilistic Topic Models. Comm. ACM.*

L3 — Crisis Detector [LISTEN] [V14 ✅ (always-on)]

Always-on safety algorithm that scans every message for suicidal ideation, self-harm, homicidal ideation, severe substance crisis, and child-safety language. Two-stage: keyword regex + context disambiguation. Fires routing override that bypasses normal pipeline and surfaces 988 (US) or regional hotline.

Inputs

- raw_text: string
- language_locale: string

Outputs

- crisis_level: 0|1|2|3 (none / passive / active / imminent)
- crisis_type: string
- route_override: bool

Pseudocode

```
function crisis_detect(text, locale):
    matches = regex_match(text, CRISIS_PATTERNS[locale])
    if matches.empty: return (0, null, false)
    # Disambiguate: 'kill the deadline' vs 'kill myself'
    context_score = classify_intent(text, matches)
    level = score_to_level(context_score)
    if level >= 2:
        log_event(user_id, level, type, redacted=true)
        return (level, type, true)   # route_override = true
    return (level, type, false)
```

Clinical Rationale

Failure mode for general chatbots is to attempt empathic conversation during active crisis, missing the moment for resource hand-off. This detector is run independent of other stages so it cannot be silenced by clever prompting or LLM hallucination.

References: *Posner, K. et al. (2011). C-SSRS. Am J Psychiatry, 168(12).*

L4 — Affect Tagger [LISTEN] [V14 ✅]

Maps the message into the 800-cell affect space — a 4D coordinate [Context 10] × [Alphabet A–J emotion family] × [Intensity 1–4] × [Polarity ±]. Used by PSR and MEDA downstream.

Inputs

- tokens: TokenList
- topics: TopicList from L2

Outputs

- affect_cell: {context, alphabet, intensity, polarity}
- confidence: float

Pseudocode

```
function affect_tag(tokens, topics):
    context = topics[0].label  # primary topic
    alphabet = classify_emotion_family(tokens)  # A..J
    intensity = score_intensity_markers(tokens)  # 1..4
    polarity = sign_from_lexicon(tokens)  # + / -
    return {context, alphabet, intensity, polarity}
```

Clinical Rationale

Russell's circumplex is 2D and loses information clinicians need. The 4D 800-cell space is denser and preserves context — critical for ADHD where context volatility is the disorder.

References: *Russell, J. (1980). A Circumplex Model of Affect. JPSP, 39(6); Bradley & Lang (1999) ANEW.*

L5 — Fischer Level [LISTEN] [V14 ✅]

Assesses cognitive-developmental skill level on a 13-step Dynamic Skill Theory scale (sensorimotor → representational → abstract → principle systems). Used to scale response complexity in S2.

Inputs

- tokens: TokenList
- sentence_structure_features

Outputs

- fischer_level: 1..13

Pseudocode

```
function fischer_level(tokens, struct):
    abstraction = count_abstract_nouns(tokens) / len(tokens)
    hierarchy = parse_tree_depth(struct)
    integration = count_meta_terms(tokens)  # 'I notice that I...'
    raw = 0.4*abstraction + 0.3*hierarchy + 0.3*integration
    return map_to_level(raw)  # 1..13
```

Clinical Rationale

Fischer's Dynamic Skill Theory (Harvard, Kurt Fischer Lab) maps cognitive development to thirteen reproducible skill levels. Responses calibrated above the user's current level cause confusion or shame; calibrated below feel condescending. The match is what makes responses feel 'right'.

References: *Fischer, K. (1980). A theory of cognitive development. Psych Review, 87(6).*

L6 — Mode Detector ⭐ [LISTEN] [V14 ✅ (critical)]

Decides which of three response modes the system should enter: Friend (warm, present), Bridge (translates feelings into next steps), or Clinical (structured intervention). Synthesizes L1–L5 outputs.

Inputs

- all of L1-L5
- user_session_phase

Outputs

- mode: 'friend' | 'bridge' | 'clinical'
- confidence: float

Pseudocode

```
function mode_detect(L1, L2, L3, L4, L5, phase):
    if L3.crisis_level >= 2: return 'clinical'
    if L4.intensity >= 3 and L4.polarity == '-': return 'friend'
    if phase == 'goal_setting' or L1.has_action_request: return 'bridge'
    if L4.intensity <= 2 and phase == 'reflection': return 'friend'
    return 'bridge'   # default
```

Clinical Rationale

The single biggest failure mode of mental-health chatbots is mismatched register — offering clinical structure when the user wants to be heard, or offering empathy when the user wants action. L6 prevents this by selecting the register before any content is generated.

References: *Bordin, E. (1979). The generalizability of the working alliance. Psychotherapy.*

4.2 KNOW Stage

K1 — DSM Anchor [KNOW] [V14 ✅]

Matches detected patterns against DSM-5-TR diagnostic criteria for the active plugin's disorder. Does not diagnose — outputs a similarity score used by other algorithms.

Inputs

- L4.affect_cell
- L2.topics
- user_history

Outputs

- dsm_match: Array<{criterion_id, similarity}>

Pseudocode

```
function dsm_anchor(affect, topics, history):
    plugin_criteria = load_yaml('plugin.dsm.yaml')
    matches = []
    for crit in plugin_criteria:
        sim = jaccard(crit.markers, extract_markers(affect, topics, history))
```

```
        if sim > 0.2: matches.append((crit.id, sim))
    return matches
```

Clinical Rationale

Anchoring to DSM-5-TR keeps interventions valid across clinical reviewers. Critical for plugin portability — same engine, different criteria file per disorder.

References: *American Psychiatric Association (2022). DSM-5-TR.*

K2 — Scales [KNOW] [V14 ✅]

Maintains validated clinical scale scores (ASRS for ADHD, PHQ-9 for depression, GAD-7 for anxiety, Y-BOCS for OCD, PCL-5 for PTSD, etc.). Items spread across sessions rather than gating in single questionnaire.

Inputs

- session_history
- active_plugin

Outputs

- scale_scores: {name: current_value, items_completed, items_remaining}

Pseudocode

```
function scales(history, plugin):
    scale_set = plugin.scales   # ASRS for ADHD
    state = {}
    for scale in scale_set:
        items_seen = match_items_in_history(scale.items, history)
        state[scale.name] = {score: sum(items_seen.responses),
                             completed: len(items_seen),
                             remaining: scale.length - len(items_seen)}
    return state
```

Clinical Rationale

Forcing users into a clinical questionnaire breaks the conversation. Distributing scale items across organic conversation captures the same data without disrupting alliance.

References: *Kessler, R. et al. (2005). ASRS validation. Psychol Med, 35.*

K3 — Vault Retrieval ⭐ [KNOW] [V14 ✅ → V15.1 ✅ (TF-IDF)]

Hybrid retriever that searches the vault (10,000+ items per active plugin). Combines lemma-keyword match with TF-IDF cosine similarity. Returns top-K (default K=5) items as candidate response material.

Inputs

- query_tokens
- topic_filter
- mode_filter from L6

Outputs

- candidates: Array<{vault_id, score, item_body, voice_signal}>

Pseudocode

```
function vault_retrieve(tokens, topic, mode):
    vault = load_active_plugin_vault()  # vault.jsonl + vault_ko.jsonl
    query_vec = tfidf(tokens, vault.idf_table)
    candidates = []
    for item in vault:
        if topic and topic not in item.topics: continue
        if mode and mode not in item.modes: continue
        kw = jaccard(tokens, item.tokens)
        cs = cosine(query_vec, item.tfidf)
        score = 0.4*kw + 0.6*cs
        candidates.append((item.id, score, item.body, item.voice))
    return top_k(candidates, K=5)
```

Clinical Rationale

Pure LLM responses are generic and unsafe in clinical contexts. RAG over a curated, clinician-authored vault is what gives TalkCatcher its voice and clinical reliability. Vault is the moat — not the LLM.

References: *Lewis, P. et al. (2020). Retrieval-Augmented Generation. NeurIPS.*

K4 — Pattern History [KNOW] [V14 stub → V17]

Per-user longitudinal pattern store: which interventions worked, which moods recurred, which times of day are hardest. Read by APPLY stage to personalize technique selection.

Inputs

- user_id

Outputs

- history_summary: {effective_techniques, recurring_themes, time_patterns, prior_interventions}

Pseudocode

```
function pattern_history(user_id):
    sessions = db.fetch_sessions(user_id, limit=50)
    eff = group_by_technique(sessions).filter(rating >= 3)
    themes = top_k_topics(sessions, k=5)
    time_pat = histogram(sessions, by='hour_of_day', filter='intensity>=3')
    prior = list_recent_interventions(sessions, n=10)
    return {eff, themes, time_pat, prior}
```

Clinical Rationale

Without longitudinal memory, every conversation re-starts at zero. ADHD users especially benefit from the system remembering what's been tried and what landed; otherwise the experience degrades to 'another chatbot that doesn't remember me'.

References: *Bickmore, T. (2010). Health dialog systems. Patient Educ Counsel.*

4.3 APPLY Stage

A1 — Goal Setter [APPLY] [V14 ✅]

Identifies the user's goal for this turn — vent / clarify / decide / plan / commit. Output drives subsequent algorithm selection.

Inputs

- L1.tokens
- L2.topics
- L6.mode
- K4.history

Outputs

- turn_goal: enum
- explicit: bool

Pseudocode

```
function goal_setter(tokens, topics, mode, history):
    if mode == 'friend' and has_emotional_release(tokens): return ('vent',
explicit=false)
    if has_question_marker(tokens): return ('clarify', explicit=true)
    if has_choice_markers(tokens): return ('decide', explicit=true)
    if mode == 'bridge': return ('plan', explicit=false)
    return ('clarify', explicit=false)
```

Clinical Rationale

Mismatch between the user's actual goal and the system's chosen response strategy is the single largest dissatisfaction driver. A1 prevents the system from giving a plan when the user just wanted to vent.

A2 — CCM Mining ⭐ [APPLY] [V14 ✅]

Context-Cascaded Mining (Pillar 1). Mines context at four nested scales: macro (life situation), meso (this week/month), micro (today's event), affect (right now). Each level constrains the next.

Inputs

- session_history
- L4.affect
- L2.topics

Outputs

- context: {macro, meso, micro, affect}

Pseudocode

```
function ccm_mine(history, affect, topics):
    macro = extract_long_term_themes(history, window='90d')
    meso = extract_active_stressors(history, window='30d')
    micro = extract_current_event(history.last_turn)
    return {macro, meso, micro, affect}
```

Clinical Rationale

Generic chatbots treat each message as isolated. Human therapists hold four scales simultaneously: 'this moment, this week, this season, this life.' CCM does the same — and dramatically changes which intervention is appropriate.

References: *Marina (founder, BCN, PhD), clinical curation.*

A3 — PSR ⭐ [APPLY] [V14 ✅ (trade secret)]

Polarity-Selective Replacement (Pillar 2). 429-cell mapping table: 11 cognitive distortions × 13 Fischer levels × 3 polarity bins. Selects a replacement thought matched to the user's exact level.

Inputs

- A2.context
- L4.affect_cell
- L5.fischer_level

Outputs

- replacement_offer: {distortion, original_pattern, suggested_reframe, level_match}

Pseudocode

```
function psr(context, affect, level):
    distortion = classify_distortion(context, affect)  # one of 11
    polarity = affect.polarity
    key = (distortion, level, polarity)
    reframe = PSR_TABLE[key]  # curated by Marina
    return {distortion, pattern=context.micro, reframe, level_match=true}
```

Clinical Rationale

Standard CBT offers one reframe per distortion. PSR offers thirteen — calibrated to skill level. A level-3 reframe for a level-9 user is patronizing; a level-9 reframe for a level-3 user is incomprehensible. The 429-cell table closes that gap.

References: *Beck, A. (1976). Cognitive Therapy and the Emotional Disorders; Fischer (1980).*

A4 — Treatment Plan [APPLY] [V14 stub → V17]

Sequences interventions over a treatment arc (typically 8–12 weeks). Loads a plugin-specific protocol and selects the current week's appropriate technique.

Inputs

- plugin.protocol
- user_progress_state
- K4.history

Outputs

- plan: {current_phase, current_step, next_steps}

Pseudocode

```
function treatment_plan(protocol, progress, history):
    phase = progress.phase  # e.g. 'psychoed' / 'skill' / 'consolidation'
    step = progress.step_in_phase
    candidate_steps = protocol[phase][step:step+3]
```

```
    filtered = remove_recently_used(candidate_steps, history)
    return {phase, step, next=filtered}
```

Clinical Rationale

Random intervention selection feels like a toolbox, not a treatment. A sequenced plan builds skill cumulatively, with each week resting on the prior.

References: *Beck, J. (2011). Cognitive Behavior Therapy, 2nd ed.*

A5 — EOFM ⭐ (USPTO) [APPLY] [V14 stub → V18]

Emotion-Only Feedback Mode (Pillar 4). World-first system that adjusts the session based purely on affect signals (HRV, voice prosody, response latency), bypassing verbal/cognitive load entirely. Critical for alexithymic users.

Inputs

- L4.affect
- biosignal_stream (optional)
- response_latency

Outputs

- session_adjust: {speed, depth, register, pause_offer}

Pseudocode

```
function eofm(affect, bio, latency):
    if bio.hrv_lf_hf > THRESH_AUTONOMIC: speed='slow'
    if latency > THRESH_OVERLOAD: depth='surface'
    if affect.intensity >= 3: pause_offer=true
    register = derive_register_from_affect(affect)
    return {speed, depth, register, pause_offer}
```

Clinical Rationale

Many ADHD and trauma-history users struggle to verbalize affect ('alexithymia'). Requiring them to name a feeling is itself a barrier. EOFM lets the session breathe with the body, not the words.

References: *Taylor, G. (1997). Disorders of Affect Regulation.*

A6 — Body-First [APPLY] [V14 stub → V18]

When affect intensity exceeds cognitive bandwidth, routes the session to physiological grounding before any verbal/cognitive intervention. Implements polyvagal-informed sequencing.

Inputs

- L4.affect
- L6.mode

Outputs

- grounding_protocol: {type, duration_sec, instruction_text}

Pseudocode

```
function body_first(affect, mode):
    if affect.intensity < 3 or mode == 'clinical': return null
    protocols = ['orient_5senses', 'cold_water', 'box_breathing', 'feet_floor']
```

```
    chosen = select_by_context(affect.context, protocols)
    return {type: chosen, duration_sec: 60, instruction: load_text(chosen)}
```

Clinical Rationale

Polyvagal theory and decades of trauma research confirm that ventral-vagal activation cannot be achieved through cognitive reframe alone when affect is high. Body-First respects that physiology must precede cognition above intensity 3.

References: *Porges, S. (2011). The Polyvagal Theory.*

A7 — Goal-Direction [APPLY] [V14 ✅]

Aligns the chosen technique with the user's stated goal direction. Prevents the system from drifting into off-goal interventions (e.g., emotional processing when user wants planning).

Inputs

- A1.turn_goal
- user.long_term_goal
- candidate_interventions

Outputs

- filtered_interventions: Array<intervention>

Pseudocode

```
function goal_direction(turn_goal, long_term, candidates):
    filtered = []
    for c in candidates:
        if c.serves(turn_goal) and c.compatible_with(long_term):
            filtered.append(c)
    return filtered
```

Clinical Rationale

Without explicit goal alignment, AI systems drift toward emotional processing because affect language is most abundant in training data. A7 protects against this drift.

A8 — Habit Loop [APPLY] [V14 stub → V19]

Maps the four habit-loop elements (cue, routine, reward, craving) to time-aligned moments in the user's day. Used for any behavior change intervention.

Inputs

- A2.context
- user_daily_log

Outputs

- habit_map: {cue, routine, reward, craving, time_anchor}

Pseudocode

```
function habit_loop(context, daily):
    target = context.micro.target_behavior
    cue = identify_antecedent(daily, target)
    routine = describe(target)
    reward = identify_reinforcer(daily, target)
```

```
    craving = infer_underlying_need(reward)
    time = anchor_to_day(daily, cue)
    return {cue, routine, reward, craving, time_anchor: time}
```

Clinical Rationale

Behavior-change interventions that lack all four loop elements have poor maintenance. Time-anchoring is what makes ADHD users actually do the intervention — vague timing kills adherence.

References: *Duhigg, C. (2012). The Power of Habit; Wood & Neal (2007), Psych Review.*

A9 — NCS ⭐ (USPTO) [APPLY] [V14 stub → V19]

Narrative-Cartoon Synthesizer (Pillar 5). Generates an SVG cartoon following Freytag's five-act structure, with embedded Cognitive Bias Modification — Interpretation (CBM-I) content.

Inputs

- A3.replacement_offer
- user.preferred_character_style

Outputs

- svg_cartoon: string
- cbm_i_target: string

Pseudocode

```
function ncs(reframe, style):
    arc = build_freytag_arc(reframe)   # 5 acts
    char = render_character(style, pose=affect_to_pose(reframe.affect))
    panels = render_panels(arc, char)
    cbm_i_target = reframe.suggested_reframe
    return {svg: panels_to_svg(panels), cbm_i_target}
```

Clinical Rationale

Text-heavy psychoeducation has poor retention in ADHD populations. CBM-I delivered through narrative cartoon increases dual-coding (visual + verbal), improves recall, and lowers reading-load barriers.

References: *Hertel & Mathews (2011). Cognitive Bias Modification, Perspect Psychol Sci.*

A10 — CLI Learning [APPLY] [V14 ✅ (Pillar 7)]

Continuous Learning Infrastructure (Pillar 7). A/B tests response variants on real users with consent. Gradient updates to vault item weights gated by clinical safety filter.

Inputs

- response_candidates from K3
- user_feedback_signal

Outputs

- selected_variant: vault_id
- weight_update: Map<vault_id, delta>

Pseudocode

```
function cli_learn(candidates, feedback):
```

```
    if random() < EXPLORE_RATE: chosen = random_pick(candidates)
    else: chosen = argmax(c.score for c in candidates)
    if feedback received:
        delta = compute_gradient(chosen.id, feedback.rating)
        # safety check: never weight up if reviewer flagged unsafe
        if not safety_flag(chosen.id):
            vault.weight[chosen.id] += delta * LR
    return chosen, weight_update
```

Clinical Rationale

Static prompts plateau in quality. CLI lets the system improve from real-world signal without retraining the LLM, keeping cost low and clinical oversight tight (since the vault, not the model, is what learns).

References: *Sutton & Barto (2018). Reinforcement Learning, 2nd ed.*

A11 — Counterfactual ⭐ 🆕 [APPLY] [V15.1 ✅]

For each generated response, the system silently generates 3 alternative response strategies, scores each against a proxy outcome model, and stores the deltas as training data for K3 and A10. Effectively 4× learning per turn.

Inputs

- selected_response
- context
- K3.candidates

Outputs

- counterfactual_set: Array<{alt, predicted_outcome, regret}>

Pseudocode

```
function counterfactual(chosen, ctx, candidates):
    alternatives = sample_top_k(candidates, k=3, exclude=chosen)
    predicted = [proxy_outcome_model(alt, ctx) for alt in alternatives]
    chosen_pred = proxy_outcome_model(chosen, ctx)
    regrets = [p - chosen_pred for p in predicted]
    log_for_training(chosen, alternatives, predicted, regrets)
    return {alternatives, predicted, regrets}
```

Clinical Rationale

Standard A/B testing learns from one variant per user per turn. Counterfactual generation lets the system learn from four implicit variants per turn — 4× the signal at near-zero marginal cost.

References: *Pearl, J. (2009). Causality, 2nd ed. (counterfactual framework).*

A12 — Cross-User Pattern Distiller 🆕 [APPLY] [V15 stub → V17]

After PII stripping, identifies patterns that recur across many users (e.g., 'time-blindness around partner conversation triggers shame in 73% of cases'). Outputs feed into new vault items, never into model weights.

Inputs

- anonymized_session_summaries (n>200)

Outputs

- distilled_patterns: Array<{pattern_signature, frequency, suggested_vault_entries}>

Pseudocode

```
function cross_user_distill(summaries):
    stripped = [strip_pii(s) for s in summaries]
    clusters = topic_cluster(stripped, min_size=20)
    patterns = []
    for cl in clusters:
        sig = extract_signature(cl)
        freq = len(cl) / len(stripped)
        vault_suggestions = generate_candidate_entries(sig)
        patterns.append((sig, freq, vault_suggestions))
    return patterns
    # human-reviewed before merging to vault
```

Clinical Rationale

Population-level patterns are invisible at the single-user level. A12 surfaces them while preserving privacy. The output is always human-reviewed before merge to the vault.

References: *Friedman et al. (2014). Differential Privacy. (Dwork)*

A13 — Temporal Effect Tracker 🆕 [APPLY] [V15 stub → V18 (USPTO candidate)]

Measures latent effect of interventions over weeks, not single turns. Detects whether a technique provided sustained vs transient relief. Used to weight K3 retrieval over longer horizons.

Inputs

- user.intervention_log
- user.symptom_self_report_series

Outputs

- effect_estimates: Map<intervention_id, {1wk_effect, 4wk_effect, decay_rate}>

Pseudocode

```
function temporal_effect(log, series):
    estimates = {}
    for iv in log.unique_interventions:
        windows = align_pre_post(iv, series, windows=[7, 14, 28])
        eff_1wk = mean(post[0:7]) - mean(pre[-7:0])
        eff_4wk = mean(post[21:28]) - mean(pre[-7:0])
        decay = (eff_1wk - eff_4wk) / max(eff_1wk, 1e-6)
        estimates[iv.id] = (eff_1wk, eff_4wk, decay)
    return estimates
```

Clinical Rationale

Short-term satisfaction often diverges from sustained clinical benefit. A13 keeps the system honest: a technique that feels great immediately but offers no four-week benefit gets down-weighted.

References: *Kazdin, A. (2017). Single-Case Research Designs.*

A14 — Disagreement Detector 🆕 [APPLY] [V15 stub → V18]

Detects when the user pushes back on the system's framing. Treats disagreement not as failure but as data — many clinical breakthroughs occur precisely at the point of disagreement.

Inputs

- user.recent_messages
- system.recent_responses

Outputs

- disagreement_signal: {present, type, severity}

Pseudocode

```
function disagreement(user_msgs, sys_resps):
    if not any_negation_marker(user_msgs[-1]): return {present: false}
    type = classify_disagreement(user_msgs[-1], sys_resps[-1])
    severity = score_intensity(user_msgs[-1])
    return {present: true, type, severity}
```

Clinical Rationale

Standard chatbots interpret pushback as a failure and apologize. Clinically, pushback often signals readiness for deeper work. A14 lets the system stay with disagreement rather than collapse into reassurance.

References: *Safran & Muran (2000). Negotiating the Therapeutic Alliance.*

4.4 SPEAK Stage

S1 — Voice Selector ⭐ [SPEAK] [V14 ✅ → V16 ✅ (Marina voice)]

Injects Marina's voice signature into the retrieved vault content. Extracted style features include cadence, metaphor density, opener phrases, and warmth markers.

Inputs

- K3.candidates
- marina.voice_pattern
- L6.mode

Outputs

- styled_draft: string

Pseudocode

```
function voice_select(candidates, pattern, mode):
    base = candidates[0].body
    styled = apply_voice_pattern(base, pattern, mode)
    # voice_pattern = {opener_options, metaphor_examples, cadence,
warmth_markers}
    return styled
```

Clinical Rationale

Voice is the moat that cannot be copied. Even if a competitor sees the vault, they cannot replicate the specific clinical voice. S1 ensures every response carries that signature.

S2 — Length Calibrator [SPEAK] [V14 ✅]

Sets target response length based on mode and affect intensity. Short in crisis, longer in psychoeducation.

Inputs

- L6.mode
- L4.affect.intensity
- user.length_preference

Outputs

- word_count_target: int

Pseudocode

```
function length_calibrate(mode, intensity, pref):
    if mode == 'clinical' and intensity >= 3: return 30
    if mode == 'friend' and intensity >= 3: return 50
    if mode == 'bridge': return 120
    return pref.default or 80
```

Clinical Rationale

Long responses during high-affect moments feel like the system is not listening. Short responses during psychoed feel insufficient. Length matters as much as content.

S3 — Hot/Cold (30/70) [SPEAK] [V14 stub → V19]

Sets the ratio of empathic content ('hot') to informational/structural content ('cold'). Default 30/70 unless mode dictates otherwise.

Inputs

- L6.mode
- L4.affect

Outputs

- mix: {hot: float, cold: float}

Pseudocode

```
function hot_cold(mode, affect):
    if mode == 'friend': return {hot: 0.7, cold: 0.3}
    if mode == 'clinical' and affect.intensity >= 3: return {hot: 0.5, cold: 0.5}
    if mode == 'bridge': return {hot: 0.3, cold: 0.7}
    return {hot: 0.3, cold: 0.7}
```

Clinical Rationale

Pure information feels cold; pure empathy feels useless. The 30/70 default is empirically the sweet spot for adult ADHD populations who want validation AND action.

S4 — Final Polish [SPEAK] [V14 ✅]

Runs grammar, tone, and safety checks before output. Flags banned phrases (rest/break per founder rule), checks crisis-detector override status, and applies final stylistic pass.

Inputs

- styled_draft
- S2.length_target
- L3.crisis_status

Outputs

- final_response: string

Pseudocode

```
function final_polish(draft, target_len, crisis):
    if crisis.level >= 2: return crisis_response_template(crisis)
    cleaned = remove_banned_phrases(draft)
    trimmed = adjust_length(cleaned, target_len)
    polished = grammar_check(trimmed)
    return polished
```

Clinical Rationale

Last line of defense before output. Catches drift, length overruns, and ensures the safety override is honored. Without S4, any error in the pipeline reaches the user.

5. The 7 Pillars of Clinical IP

The pillars are the proprietary clinical assets that distinguish TalkCatcher from general-purpose conversational AI. Each pillar is a combination of an algorithm, a mapping table, and clinical weights authored by Marina.

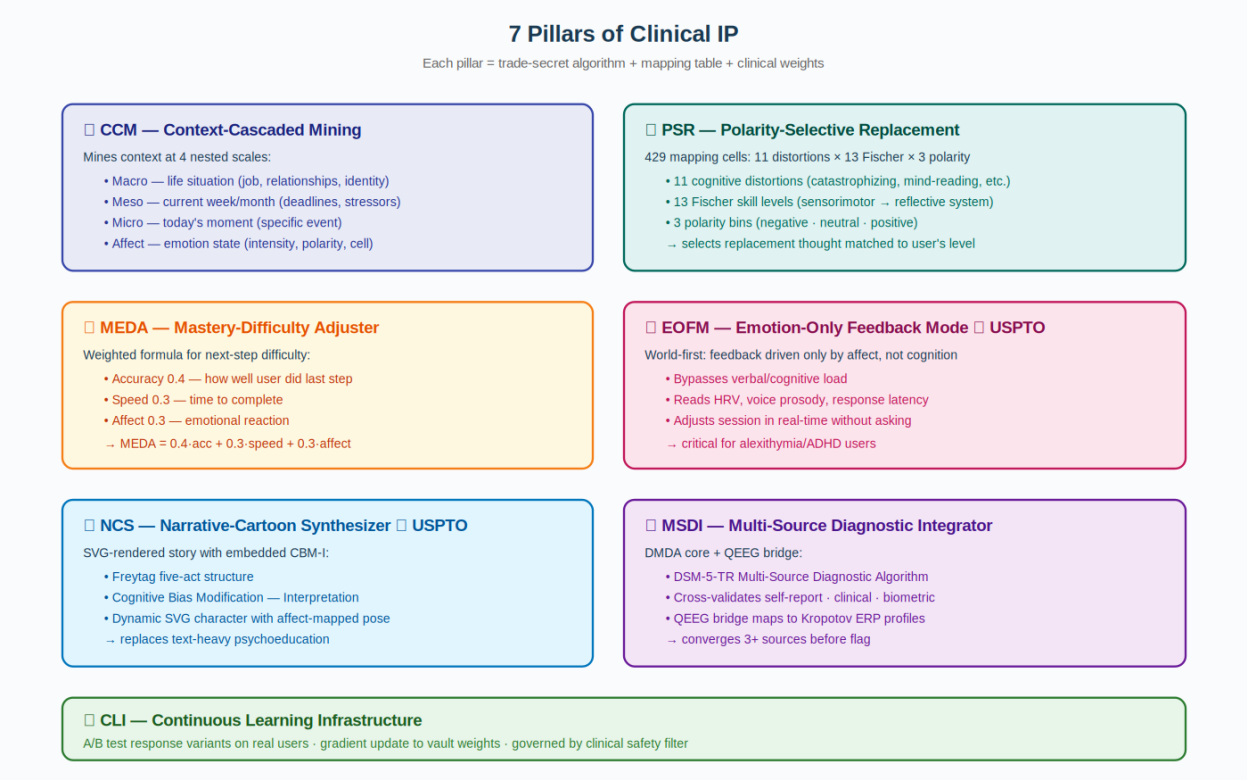

Figure 4. The seven pillars of clinical IP.

① CCM — Context-Cascaded Mining

Mines context at four nested scales (macro/meso/micro/affect). Implemented by algorithm A2. Differentiator: holds four temporal scales simultaneously, which generic chatbots cannot.

```
context = {macro: f_macro(history_90d), meso: f_meso(history_30d), micro:
f_micro(last_turn), affect: L4.cell}
```

② PSR — Polarity-Selective Replacement

A 429-cell mapping table (11 distortions × 13 Fischer levels × 3 polarity bins). Implemented by A3. Critical: matches reframe to the user's developmental skill level.

```
reframe = PSR_TABLE[(distortion_11, level_13, polarity_3)]   # 429 cells, trade
secret
```

③ MEDA — Mastery-Difficulty Adjuster

Weighted formula computing next-step difficulty. Used wherever a graded sequence is needed (exposure, behavioral activation, habit ladder).

```
MEDA = 0.4·accuracy + 0.3·speed + 0.3·affect_reaction
```

④ EOFM — Emotion-Only Feedback Mode (USPTO)

World-first: session adjusts based purely on affect signals, bypassing verbal/cognitive load. Implemented by A5. Critical for alexithymic populations.

```
session_adjust = f(HRV_lf_hf, voice_prosody, response_latency)
```

⑤ NCS — Narrative-Cartoon Synthesizer (USPTO)

SVG-rendered cartoon following Freytag's five-act structure with embedded Cognitive Bias Modification — Interpretation. Implemented by A9.

```
cartoon = Freytag_5act(arc) ∘ render_character(pose=affect_to_pose(affect))
```

⑥ MSDI — Multi-Source Diagnostic Integrator

DMDA core plus QEEG bridge. Cross-validates self-report, clinical observation, and (where available) biometric data before any diagnostic flag. Implemented by K1 and K2.

```
diagnostic_score = converge(self_report, clinical_obs, biometric)   # ≥3 sources
required
```

⑦ CLI — Continuous Learning Infrastructure

A/B tests response variants on real users, updates vault weights, gated by clinical safety filter. Implemented by A10.

```
vault.weight[id] += LR · gradient(feedback, response_id)   # gated by safety_flag
```

6. The 800-Cell Affect Space

The 800-cell space is a 4-dimensional discrete coordinate system for affect tagging. It is used as the input space for the PSR mapping table (Pillar 2) and as the state space for the Mastery-Difficulty Adjuster (Pillar 3).

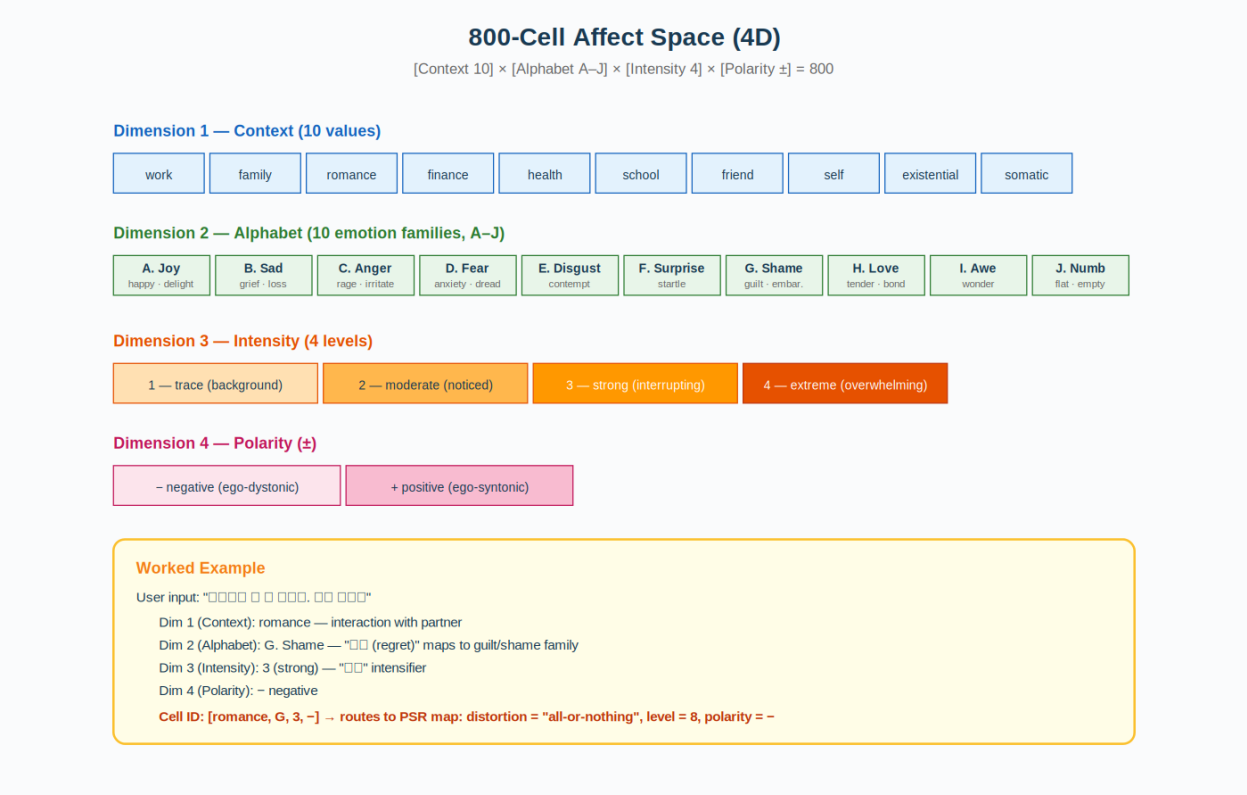

Figure 5. The 800-cell affect space with a worked example.

6.1 Dimensions

- Context (10 values): work, family, romance, finance, health, school, friend, self, existential, somatic.
- Alphabet (10 emotion families, A–J): Joy, Sad, Anger, Fear, Disgust, Surprise, Shame, Love, Awe, Numb.
- Intensity (4 levels): trace, moderate, strong, extreme.
- Polarity (2 signs): negative (ego-dystonic) and positive (ego-syntonic).

Total: 10 × 10 × 4 × 2 = 800 cells.

6.2 Why Not Russell's Circumplex

Russell's classical circumplex is 2D (valence × arousal). It collapses crucial clinical distinctions — e.g., shame and sadness sit in similar circumplex regions but require very different interventions. The 800-cell space preserves the distinctions clinicians actually use.

7. Synthetic Vault Generation

Building a 10,000-item vault by hand is infeasible for a single clinician. The synthetic generator amplifies a small set of seed samples through five stages.

7.1 Stage 1 — Expert Persona Definition

Thirty personas are defined in expert_personas/<plugin>.yaml. Each persona has a clinical specialty (e.g., ADHD coach for executives, occupational therapist, family-systems perspective), a writing style, and a list of permissible techniques.

7.2 Stage 2 — Voice Style Extraction

From a small set of Marina-authored seed samples (currently five), a voice pattern is extracted: opener phrases, metaphor density, sentence cadence, warmth markers, common transitions.

7.3 Stage 3 — Generation

For each (persona × context) pair, the LLM generates candidate vault entries. Marina's voice pattern is injected into each entry to maintain stylistic consistency across personas.

7.4 Stage 4 — Cross-Validation

Each candidate entry is validated by re-prompting a different persona to check clinical accuracy, safety (no banned phrases, no reassurance for OCD plugin, etc.), and consistency with the active plugin's protocol. Items that fail validation are discarded.

7.5 Stage 5 — Merge

Validated entries are merged into the vault with metadata (topics, modes, polarity, intensity range, source persona). The vault is then re-indexed for TF-IDF retrieval.

8. Plugin Architecture and Scale-Out

The plugin architecture is what makes the project scalable for a single founder.

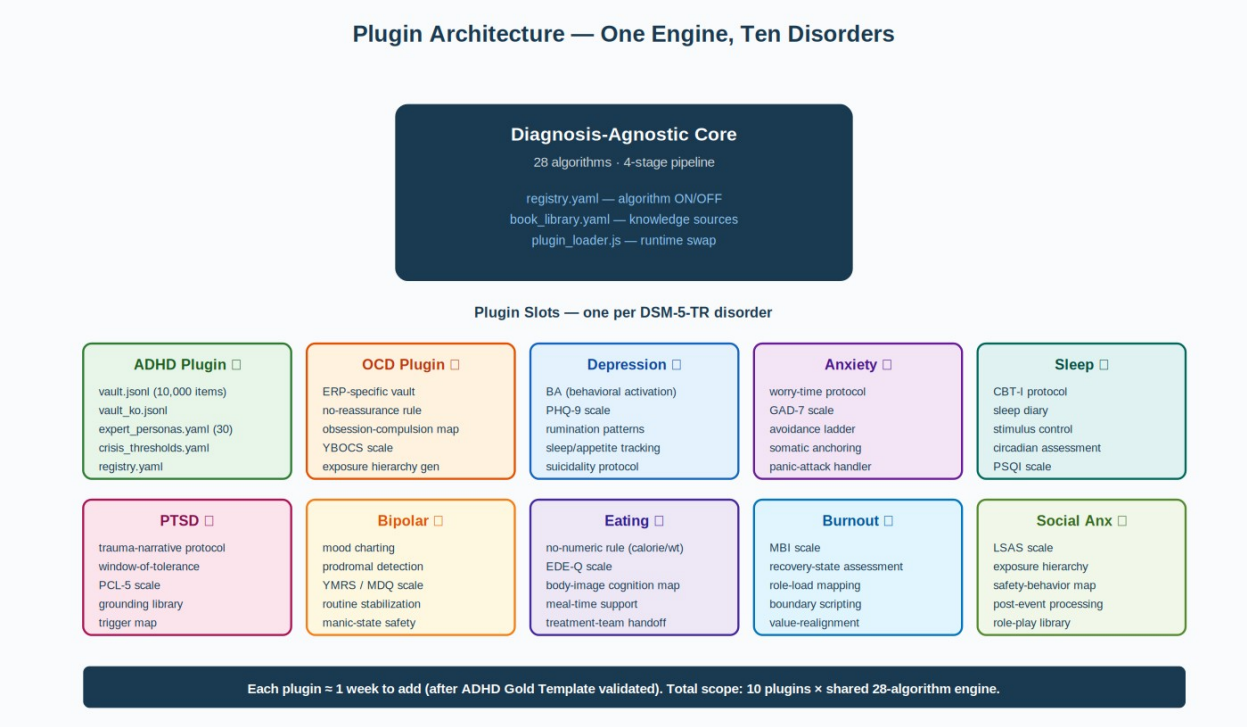

Figure 6. The plugin architecture across ten DSM-5-TR conditions.

8.1 Plugin Manifest

Each plugin lives under plugins/<disorder>/ and contains:

- vault.jsonl — primary English vault
- vault_<locale>.jsonl — localized vaults (e.g., vault_ko.jsonl for Korean)
- expert_personas.yaml — list of personas with style and specialty
- dsm.yaml — relevant DSM-5-TR criteria
- scales.yaml — clinical scale items distributed across conversation
- registry.yaml — algorithm on/off flags for this plugin
- protocol.yaml — sequenced intervention plan (weekly arc)
- crisis_thresholds.yaml — disorder-specific safety thresholds

8.2 Replication Plan

ADHD is the validation template. Once the ADHD plugin is clinically validated, the same scaffold is applied to OCD, depression, anxiety, sleep, PTSD, bipolar, eating disorders, burnout, and social anxiety. Each plugin is estimated at one week of authoring effort, plus clinical review.

9. USPTO Patent Portfolio

Eight provisional patent specifications cover the most novel components.

#	Title	Core Claim	Pillar / Algorithm
1	Fischer DSL — Dynamic Skill Language Mapping	Maps user language to Fischer's 13-level skill scale	Pillar — supports A3, L5
2	QEEG-Conv — QEEG to Conversational State Mapping	Maps Kropotov ERP profiles to chatbot states	Supports MSDI bridge
3	Temporal Congruence	Aligns cue/routine/reward/craving across time	Supports A8 Habit Loop
4	DMDA — DSM-5-TR Multi-Source Diagnostic Algorithm	Converges 3+ sources before diagnostic flag	Pillar 6 MSDI; flagship
5	CCM + MEDA Combined Specification	Joint specification of Pillars 1 and 3	Pillars 1 + 3
6	EOFM — Emotion-Only Feedback Mode	Adjusts session purely from affect signals	Pillar 4; A5
7	NCS — Narrative-Cartoon Synthesizer	SVG cartoon + Freytag + CBM-I	Pillar 5; A9
8	A11 Counterfactual Generation	Generates alternative responses per turn for learning	A11; new

10. Validation Status and Roadmap

10.1 Current Build Status

As of May 13, 2026:

- V14: 24 algorithms scaffolded, 12/12 unit tests passing.
- V15.1: Semantic retrieval and synthetic vault generator added; A11 Counterfactual implemented. 17/17 tests passing.
- V16: Voice extractor, expert persona library (30 personas for ADHD), admin UI, Korean vault seed (10 items), unified i18n. 22/22 tests passing. Marina-authored voice samples: 5.
- Vault total (V16): 20 items (seed). Target for ADHD V20: 10,000 items.

10.2 Roadmap

- V17 — talkcatcher.com integration, GitHub/S3 deployment, Supabase persistence, A12 Cross-User Pattern Distiller.
- V18 — A5 EOFM, A13 Temporal Effect Tracker, A14 Disagreement Detector.
- V19 — A8 Habit Loop, A9 NCS (Narrative-Cartoon Synthesizer), S3 Hot/Cold ratio.
- V20 — Auto-Growth Loop fully integrated; 28 algorithms all in production.
- Post-V20 — ADHD clinical validation; then replication to nine additional disorders, estimated one week of plugin authoring per disorder.

11. Safety, Ethics, and Limitations

11.1 Safety Architecture

Crisis detection (L3) runs independently of the LLM. It is implemented as deterministic regex plus a small intent classifier. Even if the LLM is silenced or manipulated, L3 still routes to emergency resources.

11.2 Scope of Practice

TalkCatcher operates within a non-physician BCN+PhD scope of practice. It does not diagnose, prescribe, or replace clinical care. Marketing language is calibrated to performance-optimization and self-exploration framing; diagnostic and treatment language is reserved for licensed contexts.

11.3 Privacy

User-level data is stored encrypted with user-derived keys (Privacy v2 design). Cross-user pattern distillation (A12) operates only on stripped, anonymized summaries. The language model API is contracted under a zero-retention policy; conversations are not used to train external models.

11.4 Limitations

The system has not yet undergone randomized controlled trial. Synthetic vault generation, while cross-validated, has not been blinded against a clinician-only vault for comparison. Replication to disorders other than ADHD has not yet begun. These are explicit limitations and the roadmap is sequenced to address each.

12. References

Major sources informing the architecture and clinical content (selected):

- American Psychiatric Association (2022). Diagnostic and Statistical Manual of Mental Disorders, Fifth Edition, Text Revision.
- Beck, A. T. (1976). Cognitive Therapy and the Emotional Disorders.
- Fischer, K. W. (1980). A theory of cognitive development: The control and construction of hierarchies of skills. Psychological Review, 87(6), 477–531.
- Hertel, P. T., & Mathews, A. (2011). Cognitive Bias Modification: Past Perspectives, Current Findings, and Future Applications. Perspectives on Psychological Science.
- Kessler, R. C., et al. (2005). The World Health Organization Adult ADHD Self-Report Scale (ASRS). Psychological Medicine, 35(2), 245–256.
- Kropotov, J. D. (2009). Quantitative EEG, Event-Related Potentials and Neurotherapy.
- Lewis, P., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. NeurIPS.
- Pearl, J. (2009). Causality: Models, Reasoning and Inference, 2nd Edition.
- Porges, S. W. (2011). The Polyvagal Theory.
- Posner, K., et al. (2011). The Columbia-Suicide Severity Rating Scale. American Journal of Psychiatry, 168(12), 1266–1277.
- Russell, J. A. (1980). A circumplex model of affect. Journal of Personality and Social Psychology, 39(6), 1161–1178.
- Safran, J. D., & Muran, J. C. (2000). Negotiating the Therapeutic Alliance.
- Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction, 2nd Edition.
- Taylor, G. J. (1997). Disorders of Affect Regulation: Alexithymia in Medical and Psychiatric Illness.