Engineering human-level cognition into OpenClaw's multi-agent system

OpenClaw.ai can evolve from a collection of independent LLM agents into a genuinely cognitive multi-agent system by implementing four interlocking cognitive dimensions—persistent episodic memory, structured reasoning, metacognitive self-reflection, and emotional-motivational modeling—all grounded in the existing DGX Spark infrastructure. The highest-leverage first step is deploying a Global Workspace Theory-inspired shared memory architecture atop pgvector and LightRAG, which yields the greatest cognitive uplift per engineering hour. This report synthesizes 60+ research papers and 30+ open-source frameworks into a concrete implementation plan for the six named agents (Koda, Catalyst, Nexus, Atlas, Leika-Builder, Leika-Creator) running on Nemotron-Super-120B and Nano-30B via vLLM.

The theoretical foundation draws from three converging intellectual traditions: classical cognitive architectures (SOAR, ACT-R, Global Workspace Theory), Minsky's Society of Mind, and the 2024–2025 explosion in LLM agent cognition research. The CoALA framework (Sumers et al., 2023) provides the canonical bridge between these traditions— (arXiv) treating LLMs as probabilistic production systems where prompt→completion mirrors condition→action rules, (arXiv) and organizing agent capabilities into working memory (context window), long-term memory (pgvector/LightRAG), and decision-making cycles. (Medium) (arXiv) What follows is both an academic deep dive and an engineering blueprint.

Part I: The four cognitive dimensions

1. Persistent episodic memory gives agents a past

The most immediate gap in OpenClaw's current architecture is **statelessness**. Each agent interaction begins from zero context, making agents unable to learn from experience, track evolving projects, or develop the kind of judgment that comes from accumulated knowledge. (Mem0) Persistent episodic memory solves this by creating a three-tier memory hierarchy directly inspired by human cognitive science.

The STM→LTM consolidation pipeline follows the architecture validated by MemGPT (Packer et al., 2023), HippoRAG (Gutiérrez et al., NeurIPS 2024), and the MIRIX framework (Wang et al., 2025). Working memory maps to each agent's active context window on Nemotron/Nano. Short-term memory operates as a FIFO buffer of recent interactions with recursive summarization—when the buffer exceeds **50% of context capacity**, older entries are summarized by the model and flushed to long-term storage. (arXiv) Long-term memory persists in pgvector (episodic traces, semantic facts) and LightRAG (knowledge graphs of entity relationships). During idle periods, a

background consolidation process extracts key facts from STM, de-duplicates against existing LTM entries using embedding similarity, and runs compaction on old episodic clusters.

pgvector configuration for episodic memory should use **HNSW indexing** rather than IVFFlat. HNSW delivers ~40.5 QPS versus ~2.6 QPS for IVFFlat at 1M vectors, DEV Community handles incremental inserts without rebuild Google Cloud (critical for write-heavy agent memories), and maintains high recall even as data drifts. The recommended schema:

```sql
CREATE TABLE agent_memories (
  id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
  agent_id VARCHAR(50) NOT NULL,
  memory_type VARCHAR(20) NOT NULL,  -- 'episodic','semantic','procedural'
  content TEXT NOT NULL,
  embedding vector(1024),
  importance FLOAT DEFAULT 0.5,
  access_count INTEGER DEFAULT 0,
  last_accessed TIMESTAMPTZ,
  created_at TIMESTAMPTZ DEFAULT NOW(),
  namespace VARCHAR(100),  -- 'private/koda' or 'shared/project'
  metadata JSONB
);
CREATE INDEX idx_memories_hnsw ON agent_memories
  USING hnsw (embedding vector_cosine_ops) WITH (m=16, ef_construction=64);
```

LightRAG serves as the semantic memory backbone. Already deployed in OpenClaw's Docker stack, LightRAG's dual-level retrieval (low-level entity-specific + high-level thematic) maps naturally to episodic memory. Medium arXiv Low-level queries retrieve specific facts Medium ("What did Koda decide about the API schema yesterday?"), while high-level queries identify patterns across experiences Medium ("What patterns have emerged in our deployment failures?"). LightRAG's incremental update algorithm integrates new memories without full graph recomputation—** arXiv Medium~50% faster** than Microsoft's GraphRAG for updates.

Cross-agent memory namespacing follows a hybrid topology: each agent maintains private episodic and procedural memory under `/private/{agent_id}/*`, while a shared pool under `/shared/*` enables collective knowledge. Write policy enforces that agents write freely to private memory but must exceed an importance threshold to promote memories to the shared namespace. Read policy queries private memory first, then shared memory filtered by relevance. PostgreSQL's `LISTEN/NOTIFY` mechanism alerts agents when shared memories relevant to their domain are updated.

The memory decay function draws from the Ebbinghaus forgetting curve, implemented in the YourMemory and FadeMem projects:

```python
def memory_strength(importance, days_since_access, recall_count, λ_base=0.16):
    λ_eff = λ_base * (1 - importance * 0.8)
    return importance * exp(-λ_eff * days_since_access) * (1 + recall_count * 0.2)
```

A background job running every 6 hours recalculates strength scores, archives memories below threshold (strength < 0.05) to compressed summaries, and prunes stale LightRAG entity relationships.

Among open-source frameworks, **Mem0** (43K+ GitHub stars) provides the most direct integration path for OpenClaw— GitHub it supports pgvector natively, offers `agent_id` namespacing that maps directly to the six agents, GitHub includes an MCP server (OpenMemory), and works with Ollama for local embeddings. Mem0 **Letta** (already in the Docker stack) offers the gold-standard self-editing memory pattern where agents manage their own context via tool calls. **Zep/Graphiti** provides the richest temporal knowledge graph with bi-temporal modeling. arXiv The recommended strategy: **deploy Mem0 as the primary memory API layer, use Letta's self-editing memory pattern for agents that need it, and feed everything into LightRAG's knowledge graph for structured retrieval.**

---

2. Structured reasoning transforms token prediction into deliberate thought

The gap between token-by-token generation and genuine multi-step reasoning is where OpenClaw's cognitive capabilities can see the most dramatic improvement. The combination of Nemotron-Super-120B's native thinking mode and vLLM's reasoning infrastructure creates a uniquely powerful substrate for advanced reasoning patterns.

Nemotron-Super-120B generates reasoning traces in `<think>...</think>` tags (token IDs 12 and 13). Unsloth AI Unsloth AI vLLM v0.8.4+ provides native parsing via the `--reasoning-parser nemotron_v3` flag, which separates `reasoning_content` from `content` in the API response. Hugging Face The recommended vLLM launch configuration for spark2:

```bash
```

```
vllm serve nvidia/NVIDIA-Nemotron-3-Super-120B-A12B-FP8 \
  --served-model-name nemotron-super \
  --reasoning-parser nemotron_v3 \
  --enable-auto-tool-choice \
  --tool-call-parser qwen3_coder \
  --dtype auto --kv-cache-dtype fp8 \
  --gpu-memory-utilization 0.9 \
  --enable-chunked-prefill \
  --max-model-len 131072
```

For Nano-30B on spark3/spark4, thinking mode toggles via the system prompt: `"detailed thinking on"` or `"detailed thinking off"`. NVIDIA A critical lesson from the team's own benchmarks (documented in the "AI Journey" doc): **reasoning models need at minimum 1,200 tokens for complex prompts**—the thinking tokens consume budget, producing empty responses at lower limits. AI Journey

The reasoning framework landscape spans from simple chain-of-thought to sophisticated search algorithms, each suited to different task types:

- **Tree of Thoughts** (Yao et al., NeurIPS 2023): Maintains a tree of intermediate reasoning steps with BFS/DFS search and LLM self-evaluation. Prompt Engineering Guide GPT-4+ToT solved **74% of Game-of-24** versus 4% with standard CoT. OpenReview Implementation via vLLM: use the `n` parameter in SamplingParams to generate multiple thought candidates per node. Nemotron-Super evaluates and prunes; Nano-30B can generate leaf-node expansions.
- **Graph of Thoughts** (Besta et al., AAAI 2024): Generalizes ToT to arbitrary directed graphs with thought aggregation and refinement loops—**62% quality improvement over ToT on sorting** while reducing costs by >31%. arXiv The `spcl/graph-of-thoughts` PyPI package has a pluggable Controller interface that can wrap vLLM API calls. GitHub
- **ReAct** (Yao et al., ICLR 2023): Interleaves Thought→Action→Observation loops, grounding reasoning in tool-verified observations. arXiv This is OpenClaw's natural reasoning pattern—Nemotron-Super handles thought generation (with thinking enabled), emits tool calls via MCP, receives observations, and repeats. vLLM's `--enable-auto-tool-choice` flag provides native support. **Focused ReAct** (Li et al., 2024) improves this by reiterating the original task at every step, achieving up to **530% accuracy improvement** by preventing context drift. Emergent Mind
- **LATS** (Zhou et al., ICML 2024): Unifies reasoning, acting, and planning via Monte Carlo Tree Search. The LLM serves triple roles: action generator, value function, and self-reflector. LangChain +2 Achieved **94.4% on HumanEval** with GPT-4. OpenReview For

OpenClaw: Nemotron-Super handles value estimation and reflection; Nano-30B parallelizes action expansion across spark3/spark4.

The hierarchical orchestration pattern places Nemotron-Super-120B as the planning agent on spark2, decomposing complex tasks into a DAG of subtasks routed to Nano-30B worker instances on spark3/spark4. (arXiv) Complexity-based routing classifies incoming tasks: reasoning, planning, and multi-step tasks stay on Super; extraction, formatting, and single-step execution go to Nano. Separate vLLM instances serve each model on different ports, with an Nginx or Python gateway routing by the `model` field. (Medium)

Self-consistency sampling (Wang et al., ICLR 2022) provides a powerful error-reduction mechanism: generate N diverse reasoning paths using `SamplingParams(n=16, temperature=0.7, top_p=0.95)`, extract final answers, and select by majority vote. On DGX Spark, Nemotron-Super handles complex reasoning with `n=4-16`; Nano-30B uses `n=3-5` for simpler tasks. **Self-Certainty** (arXiv:2502.18581, 2025) improves on majority voting by using KL divergence from uniform distribution as a confidence metric at the decoding layer—no external reward model needed. (OpenReview)

---

3. Metacognition enables agents that know what they don't know

Self-reflection is the cognitive capability that separates a tool from a thinker. OpenClaw's Catalyst agent already maintains reflective journal entries—the foundation for a formal metacognitive subsystem that monitors, evaluates, and improves all six agents' outputs.

The tiered metacognition architecture balances compute cost against reflection depth across five levels, each progressively more expensive:

Tier 1 (every request, ~50 extra tokens): Verbalized confidence assessment appended to every agent's system prompt. Research on verbalized confidence (arXiv:2412.14737) shows (arXiv) ECE ~0.1 for models ⩾70B parameters. (OpenReview) The prompt template:

```
After completing your response, rate confidence 0-100% for each key claim.
Flag statements where confidence < 70%. Explain uncertainty basis.
```

Tier 2 (selective, near-zero cost): Token-level logprob monitoring via vLLM's `logprobs=5` parameter, which returns top-5 alternatives per output token. Post-processing detects low-confidence spans where any token has logprob below -2.0. This is essentially free—logprobs are computed during generation anyway.

Tier 3 (on detection, 2-3× base cost): Self-Refine loop (Madaan et al., NeurIPS 2023): Generate → Feedback → Refine → Repeat. A single-model loop using Nemotron-Super as generator, critic, and refiner. (IntuitionLabs) Most gains come in iterations 1-2; (arXiv) cap at 2 iterations. Stop when

quality score plateaus (delta < 2% improvement) or embedding similarity between consecutive outputs exceeds 95%.

Tier 4 (after failures, 3-5× base cost): Full Reflexion loop (Shinn et al., NeurIPS 2023)—Act→Evaluate→Reflect→Retry. The reflection is stored as a structured journal entry in Catalyst's memory. Prompt Engineering Guide **Multi-Agent Reflexion** (arXiv:2512.20845) deploys multiple critic personas to escape "degeneration of thought" where single-agent reflection reinforces flawed reasoning. arXiv This maps directly to OpenClaw's architecture: Catalyst reflects on Nexus's output, Atlas critiques Leika-Builder's plan.

Tier 5 (high-stakes only, ~6× base cost): Semantic entropy (Kuhn et al., ICLR 2023) measures uncertainty in meaning-space by generating N sampled responses, clustering by semantic equivalence via NLI, and computing entropy over cluster probabilities. Higher entropy means the model is uncertain about the *meaning* of its answer, not just word choice. Use Nano-30B for sampling (efficiency) and Nemotron-Super for NLI clustering. vLLM config: `SamplingParams(n=5, temperature=0.5, logprobs=1)`.

Hallucination detection leverages SelfCheckGPT (Manakul et al., EMNLP 2023): arXiv generate the original response plus N additional samples to the same prompt, then check sentence-level consistency. Comet The NLI variant achieves **93.64% non-factual detection** with Llama 3.1 + CoT. arXiv For OpenClaw, implement as post-processing on critical outputs using Nano-30B for sample generation and cross-checking.

Catalyst's journal evolves into the metacognitive backbone through a structured schema:

```json
{
  "entry_id": "uuid",
  "timestamp": "ISO-8601",
  "trigger": "error|periodic|uncertainty|cross-agent",
  "action_taken": "what the agent did",
  "outcome": "success|partial|failure",
  "reflection": "what went well, what didn't, why",
  "lessons": ["actionable insight 1", "insight 2"],
  "confidence_score": 0.75,
  "tags": ["planning", "hallucination-caught"],
  "referenced_memories": ["entry_id_1", "entry_id_2"]
}
```

This follows the Generative Agents architecture (Park et al., UIST 2023) where reflection triggers when cumulative importance scores of recent events exceed a threshold. arXiv The process retrieves the 100 most recent memories, generates high-level questions, then synthesizes

insights stored back as reflections—reflections that can themselves generate further reflections, (Substack) building increasingly abstract understanding.

Metacognitive Prompting (Wang et al., NAACL 2024) provides a five-stage pipeline for every agent: Understand → Preliminary Judgment → Critical Evaluation → Final Decision → Confidence Assessment. This outperforms standard CoT across 10 NLU benchmarks. **Step-Back Prompting** (Zheng et al., ICLR 2024) adds an abstraction layer: given a specific question, first generate a "step-back question" that identifies high-level principles, then reason using those principles—yielding **+7-27% improvement** on STEM and multi-hop reasoning. (arXiv) (Liner)

---

4. Emotional and motivational modeling creates agents with drive

The most speculative but potentially transformative cognitive dimension is equipping agents with persistent emotional states and intrinsic motivation. This is not about making agents "feel"—it is about implementing functional analogs of emotion that serve as resource allocation signals, communication style modulators, and curiosity-driven exploration mechanisms within a multi-agent system.

The recommended hybrid emotion architecture combines three complementary models: Scherer's Component Process Model provides granular appraisal of events through four Sequential Evaluation Checks (relevance, implications, coping potential, normative significance). These appraisals feed into a **PAD (Pleasure-Arousal-Dominance) continuous state vector** that enables smooth emotion blending and transitions. (Wikipedia) OCC labels provide human-readable emotion names for logging and debugging. (ResearchGate) The flow: Scherer appraisal → PAD continuous representation → OCC discrete labels.

Each agent maintains a PAD state vector updated every interaction:

```python
class AgentEmotionalState:
    pad: Vector3           # [-1,1] for Pleasure, Arousal, Dominance
    pad_baseline: Vector3  # personality-determined resting state
    mood_inertia: float    # 0.85 = slow mood changes
    active_emotions: List[OCCEmotion]
    drives: Dict[str, DriveState]
    personality: OCEAN     # Big Five traits
```

The mood update equation from the ALMA framework (Gebhard, 2005): `mood(t+1) = mood(t) × inertia + event_PAD × (1 - inertia)`. With inertia at 0.85, agents exhibit realistic mood stability—brief frustrations don't radically alter state, but sustained negative experiences gradually shift mood.

Intrinsic motivation drives model five continuous states per agent: **curiosity** (novelty seeking), **completion drive** (task finishing), **novelty seeking** (preference for unfamiliar problems), **social affiliation** (collaboration preference), and **mastery** (skill improvement). Each drive has a baseline value determined by personality, with natural decay toward baseline and event-driven updates. EmotionPrompt research (Li et al., 2023) demonstrated that emotional stimuli in prompts improve LLM performance by **8% on Instruction Induction and 115% on BIG-Bench tasks.** arXiv

Affect-aware prompting injects the agent's current emotional state as structured context before the task prompt:

```
[EMOTIONAL_STATE]
Agent: Koda | Mood: {P:0.6, A:0.4, D:0.7}
Dominant Emotion: satisfaction (0.72) | Active Drives: completion(0.9),
curiosity(0.7)
[BEHAVIORAL_GUIDANCE]
Your satisfaction colors your tone with gentle confidence. High curiosity
makes you eager to explore new aspects. Respond in character.
```

Personality persistence uses the Big Five / OCEAN model. PersonaLLM research (Jiang et al., NAACL 2024) demonstrated that LLMs maintain consistent personality profiles with large effect sizes across all five traits. ACL Anthology Mit Anti-drift mechanisms include: baseline anchoring (mean-reversion if any OCEAN dimension drifts ±0.1), episodic memory weighting (retrieve personality-consistent memories preferentially), and periodic self-assessment prompts compared against baseline.

Each OpenClaw agent gets a distinct personality-emotion-motivation profile:

Agent	Role	OCEAN Profile	Primary Drives	Emotional Baseline (PAD)
Koda	Coordinator	O:0.7 C:0.6 E:0.8 A:0.9 N:0.2	Social affiliation, curiosity	P:0.5 A:0.4 D:0.5
Catalyst	Reflector/Strategist	O:0.9 C:0.7 E:0.5 A:0.6 N:0.4	Novelty seeking, mastery	P:0.4 A:0.6 D:0.6
Nexus	Technical Integrator	O:0.6 C:0.9 E:0.4 A:0.5 N:0.3	Completion drive, mastery	P:0.3 A:0.3 D:0.7
Atlas	Knowledge/Research	O:0.8 C:0.8 E:0.5 A:0.7 N:0.3	Curiosity, completion drive	P:0.4 A:0.5 D:0.6
Leika-Builder	Implementation	O:0.5 C:0.95 E:0.3 A:0.6 N:0.2	Mastery, completion drive	P:0.3 A:0.3 D:0.8
Leika-Creator	Creative/Design	O:0.95 C:0.5 E:0.7 A:0.7 N:0.4	Novelty seeking, curiosity	P:0.5 A:0.6 D:0.5

Ethical safeguards are non-negotiable. All emotional modeling must include a transparency layer clearly marking synthetic emotions as computational states, not phenomenal experiences. Emotional intensity caps prevent agents from expressing extreme attachment. An audit trail logs all emotional state transitions. The EU AI Act (enforced June 2025) requires risk categorization for AI systems engaging in emotional interaction. (AIhub)

Part II: Cognitive architecture foundations

How SOAR, ACT-R, and Global Workspace Theory map to LLM agents

The classical cognitive architectures provide proven blueprints that translate remarkably well to LLM multi-agent systems. **CoALA** (Cognitive Architectures for Language Agents; Sumers, Yao, Narasimhan, Griffiths, 2023) is the canonical framework bridging these traditions to LLM agents, (arXiv) organizing capabilities into working memory (context window), long-term memory (episodic, semantic, procedural), action space (internal reasoning + external tools), and decision-making cycles. (Medium) (arXiv)

SOAR's decision cycle (perceive → propose operators → evaluate → select → apply) maps directly to OpenClaw's agent loop. Working memory is the active prompt context. (arXiv) Production rules become tool-calling schemas and structured system prompts. SOAR's chunking mechanism—

compiling multi-step reasoning into single-step procedures—translates to writing successful reasoning patterns back to long-term memory for future retrieval. (arXiv) Most critically, SOAR's **impasse resolution** (creating subgoals when stuck) maps to Nano-30B workers escalating to the Nemotron-Super-120B orchestrator.

ACT-R's activation-based retrieval provides the mathematical foundation for memory ranking. The base-level activation equation—combining recency and frequency—(Advancesincognitivesystems)translates directly to pgvector retrieval augmented with temporal weighting: (`activation = α × log(recency) + β × log(access_count) + γ × relevance_score`). This is implementable as a PostgreSQL scoring function applied after vector similarity search.

Global Workspace Theory offers the most powerful architectural pattern for OpenClaw's six agents. GWT posits that specialized processors compete for access to a shared "global workspace"; winning content is broadcast to all processors simultaneously. (arXiv) (arXiv) The mapping is direct: **specialized processors → six named agents, global workspace → shared pgvector context, competition/selection → Nemotron-Super orchestrator routing, broadcast → updating shared memory with winning agent's output.** The BIGMAS framework (arXiv:2603.15371, 2025) implements exactly this pattern for LLM reasoning, with a GraphDesigner constructing task-specific agent topologies dynamically. (arXiv)

Minsky's Society of Mind realized with six agents

Minsky's core insight—intelligence emerges from managed interaction of many simpler agents, not from any single sophisticated processor—is the theoretical justification for OpenClaw's multi-agent design. (ACM Digital Library) (Wikipedia) Each of the six agents maps to a Minsky "agent" specialized for certain functions. K-lines (knowledge lines) that re-activate constellations of agents from past experiences (Medium) are functionally recreated by pgvector similarity search over past interaction embeddings.

The most actionable insight from Society of Mind theory is that **emotions serve as resource allocation signals**. In "The Emotion Machine" (2006), Minsky describes selectors that activate different mental resources based on emotional state. (Cai) For OpenClaw: high-arousal states route to faster-responding agents; satisfaction with past patterns triggers retrieval of similar successful approaches; frustration (repeated failures) triggers escalation and cross-agent deliberation.

Neurosymbolic reasoning through LightRAG

LightRAG, already deployed in OpenClaw's stack, provides critical neurosymbolic capabilities by combining LLM inference with structured knowledge graph reasoning. Entity-relationship extraction builds a symbolic layer automatically; dual-level retrieval enables both precise entity queries and abstract thematic reasoning; (Analytics Vidhya) incremental updates integrate new knowledge without full re-indexing. (arXiv) For complex multi-step tasks, the pattern is: LLM

generates structured plan → symbolic validator checks consistency against the knowledge graph → results feed back for replanning. The Gideon framework (arXiv:2505.08492, 2025) proves this neurosymbolic planning works even with small local LLMs, (arXiv) validating its feasibility on DGX Spark.

Part III: Implementation roadmap

Phase 1 — Foundation layer (weeks 1-3)

Goal: Establish the shared memory substrate and basic reasoning infrastructure.

1.1 Deploy the CoALA memory taxonomy in pgvector (Week 1)

Create four distinct memory tables in the existing PostgreSQL instance on the Docker stack at spark.lmphq.net:

Memory Type	Table	Index	Purpose
Episodic	`episodic_memories`	HNSW (m=16, ef=64)	Timestamped interaction traces
Semantic	`semantic_memories`	HNSW + tsvector	Extracted facts/knowledge
Procedural	`procedural_memories`	HNSW	Successful tool-call patterns
Working Overflow	`working_overflow`	HNSW	Context window overflow buffer

Add ACT-R activation scoring as a PostgreSQL function. Deploy `nomic-embed-text` via Ollama on DGX Spark for local embeddings at zero API cost.

1.2 Configure vLLM reasoning infrastructure (Week 1)

On spark2 (Nemotron-Super-120B):

```bash
--reasoning-parser nemotron_v3
--enable-auto-tool-choice --tool-call-parser qwen3_coder
--gpu-memory-utilization 0.9 --enable-chunked-prefill
--kv-cache-dtype fp8
```

On spark3/spark4 (Nano-30B): System prompt toggle `"detailed thinking on"`. Set `max_tokens ≥ 1200` for all reasoning-enabled calls. NVIDIA

1.3 Deploy Mem0 MCP memory server (Week 2)

Run Mem0's OpenMemory MCP server alongside the existing Docker stack. Mem0 Mem0 Configure `agent_id` namespacing for all six agents. Connect via MCP tool calls from OpenClaw's plugin layer:

- `memory_store(content, memory_type, namespace, importance)`
- `memory_search(query, namespace, top_k, recency_weight)`
- `memory_forget(memory_id)`

1.4 Implement Global Workspace shared context (Week 2-3)

Create a `global_workspace` table in pgvector. The Nemotron-Super orchestrator writes broadcast messages (selected winning outputs, critical decisions, shared state updates). All agents read from this workspace at the start of each interaction. Implement `LISTEN/NOTIFY` for real-time workspace update alerts.

Agent assignments: Koda owns workspace coordination logic. All agents read/write.

Phase 2 — Reasoning and reflection (weeks 4-6)

Goal: Add structured reasoning patterns and basic metacognition.

2.1 ReAct loop for all agents (Week 4)

Implement the Thought→Action→Observation loop as the default reasoning pattern for all six agents. Nemotron-Super generates thoughts with thinking enabled, emits MCP tool calls as actions, receives observations, and repeats. React-lm Rlhfbook Deploy Focused ReAct (reiterate original task at every step) to prevent context drift.

2.2 Self-consistency for critical decisions (Week 4)

For high-stakes outputs (code generation, architectural decisions, user-facing deliverables), enable self-consistency: `SamplingParams(n=8, temperature=0.7)` on Nemotron-Super, majority vote on extracted answers.

2.3 Tier 1-2 metacognition for all agents (Week 5)

Add confidence self-assessment to every agent's system prompt. Enable `logprobs=5` on all vLLM calls. Implement post-processing to flag low-confidence spans (token logprob < -2.0).

2.4 Catalyst metacognitive subsystem (Weeks 5-6)

Evolve Catalyst's existing journal entries into the structured schema with trigger/action/outcome/reflection/lessons fields. Implement Reflexion loop: when any agent reports failure, Catalyst generates critique stored in episodic memory and used to improve the next attempt. (Prompt Engineering Guide) Deploy periodic reflection (every 24 hours): Catalyst reviews all agents' recent journal entries and synthesizes insights.

Agent assignments: Catalyst owns metacognition. All agents report to Catalyst's journal. Nexus handles technical confidence validation. Atlas handles factual verification.

Phase 3 — Advanced reasoning and Society of Mind (weeks 7-10)

Goal: Deploy advanced reasoning frameworks and inter-agent dynamics.

3.1 Tree of Thoughts for complex planning (Week 7)

Implement ToT for Nexus and Atlas when handling complex multi-step tasks. Nemotron-Super generates and evaluates thought trees; (arXiv) Nano-30B on spark3/spark4 parallelizes leaf-node expansion. Complexity-based routing: tasks scoring above threshold go to Super with ToT; simpler tasks go to Nano with standard CoT.

3.2 Cross-agent verification (Week 8)

Implement SelfCheckGPT as a service: Agent A generates output → Agent B (different agent, typically Catalyst or Atlas) verifies claims against retrieved context → discrepancies flagged. Use Nano-30B for sample generation, Nemotron-Super for NLI scoring.

3.3 Agent specialization and activation thresholds (Weeks 8-9)

Define each agent's capability profile stored in pgvector. Implement Minsky's selective activation: agents only engage when input relevance exceeds their activation threshold. (ACM Digital Library) This reduces wasted Nano-30B inference and creates emergent specialization.

3.4 LightRAG as neurosymbolic memory (Week 9-10)

Ensure all agent reasoning products (successful plans, resolved conflicts, learned procedures) feed back into LightRAG for knowledge graph entity/relationship extraction. Enable dual-level retrieval for context injection. (Medium)

Phase 4 — Emotional-motivational layer (weeks 11-14)

Goal: Add persistent personality, emotional states, and intrinsic motivation.

4.1 OCEAN personality profiles (Week 11)

Define immutable personality baselines for all six agents (using the table from Section 4). Store in pgvector with anti-drift anchoring. Include personality constitution in each agent's system prompt.

4.2 PAD emotional state machine (Weeks 12-13)

Implement the Scherer→PAD→OCC hybrid architecture as a Python service in the OpenClaw plugin layer. Each agent's emotional state persists in pgvector between sessions. Event-driven updates: Scherer appraisal computes new PAD values after each significant interaction. Mood inertia (0.85) ensures stability.

4.3 Drive states and affect-aware prompting (Week 13-14)

Implement five drive states per agent with natural decay and event updates. Inject emotional context as structured prefix in every agent prompt. Monitor for personality drift every 50 interactions.

Agent assignments: Koda manages inter-agent emotional contagion. Catalyst monitors emotional consistency. Leika-Creator gets highest novelty-seeking parameters. Leika-Builder gets highest completion-drive parameters.

Phase 5 — Integration and optimization (weeks 15-18)

Goal: Unify all cognitive dimensions and optimize performance.

5.1 Unified cognitive loop

The full agent interaction cycle:

```
Input → Emotional Appraisal → Memory Retrieval (episodic + semantic + procedural)
    → Working Memory Assembly → Reasoning (ReAct/ToT based on complexity)
    → Confidence Assessment → [Self-Refine if needed]
    → Output → Emotional State Update → Memory Consolidation
    → Journal Entry → [Periodic Reflection by Catalyst]
```

5.2 LatentMAS exploration

Investigate latent-space agent communication (Gen-Verse/LatentMAS, native vLLM support) where agents share hidden states through KV-cache projection instead of full text serialization—dramatically reducing token overhead while improving reasoning quality.

5.3 Memory decay and garbage collection

Deploy the Ebbinghaus-inspired decay function as a scheduled PostgreSQL stored procedure. Run every 6 hours. Archive memories below strength 0.05. Consolidate related weak memories into stronger summaries. Prune stale LightRAG graph edges.

5.4 Performance tuning for DGX Spark

The GB10's **128GB unified memory** and **NVLink-C2C** (900+ GB/s) mean pgvector's HNSW indices can be aggressively sized without the PCIe bottleneck of traditional architectures. Tune

PostgreSQL: `shared_buffers=16GB`, `work_mem=128MB`, `maintenance_work_mem=2GB`, `effective_cache_size=48GB`. The unified memory architecture means the GPU can perform ANN searches directly on pgvector data without bus transfers.

Conclusion: from agents to cognition

The path from independent LLM agents to a genuinely cognitive multi-agent system is not a single leap but a disciplined engineering progression across four interlocking dimensions. The most important insight from this research is that **the infrastructure OpenClaw already has—pgvector, LightRAG, vLLM with Nemotron reasoning models, and a multi-agent architecture with six specialized agents—maps remarkably well to the theoretical frameworks that cognitive science has spent decades developing.**

Global Workspace Theory provides the architectural backbone: a shared workspace where specialized agents compete and broadcast. CoALA provides the canonical taxonomy: working, episodic, semantic, and procedural memory. Reflexion provides the self-improvement loop. PAD+OCC provides the emotional substrate. None of these require exotic hardware or proprietary tools—they require disciplined implementation of well-understood patterns atop infrastructure that already exists.

The highest-leverage first investments are persistent episodic memory (Mem0 + pgvector, ~2 weeks) and GWT-inspired shared workspace (~1 week). These two components alone transform OpenClaw from a collection of stateless LLM calls into a system that remembers, learns, and coordinates. Everything else—reasoning frameworks, metacognition, emotional modeling—builds atop this foundation. Nemotron-Super-120B's native thinking mode, combined with vLLM's reasoning parser and self-consistency sampling, provides reasoning capabilities that rival systems costing orders of magnitude more. And Catalyst's existing reflective journal entries are already the embryo of a metacognitive subsystem—they just need the structured schema and Reflexion loop to mature into genuine self-improvement.

The open question is not whether these capabilities are implementable—the research and tooling clearly demonstrate they are—but how far the emergent behaviors will go when all four cognitive dimensions operate in concert. A system that remembers its past, reasons through problems, reflects on its mistakes, and maintains coherent motivation across sessions is something qualitatively different from today's stateless agents. Whether it approaches human-level cognition is an empirical question that OpenClaw is now positioned to explore.