Contents

A Combined Thesis for The Living Fund

Version: 2.0 **Author:** Henri Francois, SumLogLabs **Date:** May 2026 **Purpose:** A technical synthesis of the architecture, drawing exclusively from the research library at `/research/`. This document is the engineering specification, not a pitch.

---

1. Problem statement

Build a trading system that ingests heterogeneous market data, generates testable causal hypotheses, validates them rigorously, deploys capital to validated strategies, and corrects course on adverse evidence. Operate on a two-person headcount and roughly one thousand dollars per month of compute during the build phase. Compete on architecture rather than on scale.

The system will trade equities, ETFs, options, and futures. It will hold positions on horizons ranging from intraday to several weeks. It will generate hypotheses across multiple asset classes simultaneously and look for cross-domain patterns that single-asset funds cannot construct.

The output is not a prediction. The output is a portfolio of trades, each carrying an explicit reasoning chain, an explicit calibration, and an explicit decay clock.

2. Why scaled LLMs alone will not solve this

Three independent results in the recent literature converge on the same conclusion. First, frontier reasoning models exhibit an accuracy collapse beyond a problem-complexity threshold, and reasoning effort declines past that point even when the token budget remains available (Shojaee et al., 2025). Second, chain-of-thought is governed by training distribution, and brittleness off-distribution shows up cleanly in controlled-environment ablations (Zhao et al., 2026). Third, RL-trained reasoning models are more faithful than vanilla LLMs but still describe the cues influencing their answers only 59 percent of the time, versus 7 percent for non-reasoning baselines (Chua and Evans, 2025).

The implication is architectural. Edge does not come from scaling the model or from feeding it more data. Edge comes from compositional structure on top of the model: explicit causal discovery, explicit debate, explicit verification, and explicit memory. The literature on long-horizon agentic memory underscores how unsolved this is in standard practice; GPT-5 scores 17.9 of 100 on the StuLife four-year lifelong-learning benchmark (Cai et al., 2025). Memory is not a solved problem off the shelf. It must be built.

3. The combined architecture

Eleven components compose a single reasoning cycle. Each is anchored to specific work in the library. Each addresses a known failure mode in existing AI fund attempts. The combination is the product, not any single piece.

1. Multimodal episodic memory with biologically grounded properties (Pink et al., 2025; Ebrahimi et al., 2024; D'Alessandro et al., 2024).
2. Causal discovery designed for nonstationary financial time series (Sadeghi et al., 2024; Ferdous et al., 2023).
3. Tree-of-thoughts hypothesis generation constrained by the discovered graph (Yao et al., 2023).
4. Multi-agent debate with finance-specialist personas (Du et al., 2023; Xiao et al., 2024).
5. Calibration that corrects RL-induced overconfidence (Xie et al., 2026; Lu et al., 2026; Chua and Evans, 2025).
6. Validation via Combinatorial Purged Cross-Validation and ABM stress tests (Sadeghi et al., 2024; Robust Time Series Causal Discovery for ABM Validation).
7. Conviction-sized deployment with hard operational guards (Lin et al., 2026).
8. Reflective post-trade learning into procedural memory (Shinn et al., 2023; Park et al., 2023; Zhao et al., 2024; Allard et al., 2026).

9. Procedural memory consolidation via Bayesian selection (Forouzandeh et al., 2026; Zhang et al., 2023; Wang et al., 2024).
10. Anomaly-aware regime detection wired to attention and kill switches (Li and Fan, 2026; Martinez, 2026).
11. Source trust learned dynamically through credit assignment.

A sleep loop runs the consolidation work asynchronously, separate from inference (Park et al., 2023; Su et al., 2026; Yu et al., 2026). Working memory handles context-window paging (Packer et al., 2024; Sun and Zeng, 2025; Allard et al., 2026). The whole system operates over one shared embedding space across asset classes so cross-domain patterns can fire.

The rest of this document specifies each component, identifies the integration risks, and maps the build sequence.

4. Component one: multimodal episodic memory

Pink et al. (2025) define five properties an episodic store must satisfy to function as the long-term memory of an LLM agent: single-shot encoding, instance specificity, time stamping, contextual binding, and replayability. We adopt these as acceptance criteria for the episodic tier. Each ingested observation is stored once, retains a unique identifier, carries a timestamp, links to its full ingestion context, and remains retrievable for replay during reflective learning.

The episodic store sits in Postgres with the pgvector extension. Each row contains the raw content, an embedding, a list of related tickers, hypotheses, and sources, a salience score that decays unless reinforced, a confidence score derived from source trust, and a Graphiti-style validity window with a `superseded_by` pointer for reconsolidation. Retrieval is k-nearest-neighbor over the embedding column with secondary filters on ticker, time window, hypothesis, and regime.

Markets are inherently multimodal. Filings are text, prices are tabular, charts are images, and earnings calls are audio. LANISTR (Ebrahimi et al., 2024) and MAGNUM (D'Alessandro et al., 2024) provide two patterns for handling this. LANISTR uses attention over jointly masked modalities and tolerates missing modality samples, which matches the patchy reality of financial feeds. MAGNUM is modular at the per-modality encoder level, which matches our pluggable-source ingestion design. The v0 system uses text-only embeddings to keep cost low, with the architecture wired to accept LANISTR-style multimodal embeddings at v2 once we can spend compute on training a unified embedder.

The episodic store is the source of truth for everything downstream. All other tiers are derivative.

5. Component two: causal discovery for nonstationary markets

Standard PC and FCI assume stationarity, which financial time series violate at every regime change. Sadeghi et al. (2024) at Bloomberg propose CD-NOTS as the financial-market-aware extension of CD-NOD, handling lagged dependencies in nonstationary series. Ferdous et al. (2023)

introduce eCDANs, which uses PC-stable plus momentary conditional independence tests and a surrogate variable for time dependence to handle autocorrelated, nonstationary data efficiently in higher dimensions.

The system runs CD-NOTS as the default weekly causal discovery job over the prior 90 days of return, volume, and factor data. When dimensionality exceeds 50 series or when intraday data is involved, eCDANs runs instead. The output is a versioned causal graph stored as nodes and edges in the semantic tier (Postgres with the AGE extension). Each edge carries a confidence interval and a regime tag; edges discovered only during one regime do not transfer to others without revalidation.

A third paper in the library, "Robust Time Series Causal Discovery for Agent-Based Model Validation," provides the protocol for stress-testing discovered graphs. We simulate liquidity-shock and volatility-spike regimes via agent-based models and check whether the discovered edges survive. Edges that flip sign or vanish under perturbation are flagged as unstable and excluded from hypothesis generation.

The downstream effect is meaningful. Hypotheses are generated to explain patterns in a graph that has already passed a structural-discovery test, rather than from scratch over raw data. This forces the LLM off the training distribution where it would otherwise default to memorized patterns.

6. Component three: tree-of-thoughts hypothesis generation

Yao et al. (2023) generalize chain-of-thought to a tree where each node is a partial reasoning state, a self-evaluation prunes branches, and lookahead with backtracking explores promising paths. On Game of 24, GPT-4 with ToT achieves 74 percent accuracy versus 4 percent with vanilla CoT.

In our system, the proposer agent uses ToT as its inner loop. Each branch represents a candidate causal mechanism that could explain a discovered edge in the causal graph. Self-evaluation prunes branches that violate base rates from the `base_rates` table or that conflict with high-confidence beliefs in the semantic graph. Backtracking returns to a viable parent when a branch dead-ends.

Compute is the binding cost of ToT. Branch factor multiplies inference calls. The system applies ToT only to new theses; sleeve-tier signals (mean reversion on SPY, VIX reversion, calendar effects) skip the tree and run single-pass.

Hypotheses generated by the proposer are passed to the debate stage with the supporting branch path as a reasoning trace, not just a final answer. This trace becomes part of the persisted hypothesis record and is used later by the reflective-learning stage to attribute success or failure to specific reasoning steps.

7. Component four: multi-agent debate with finance personas

Du et al. (2023) establish that multiple LLM instances proposing answers, critiquing each other, and iterating to consensus outperform single-agent self-consistency on factuality and math. Adaptive Heterogeneous MAD reaches 76.5 on MMLU versus 63.9 for the single-agent baseline.

The basic debate protocol runs four roles for every non-trivial hypothesis. The proposer generates the candidate causal mechanism with explicit economic rationale. The critic attempts falsification using independent evidence retrieved from episodic memory. The judge weighs the exchange and emits a decision with a stated confidence. The red team has a single job: find data leakage, lookahead bias, survivorship, or regime overfit. This last role is not in the original Du protocol; we add it because financial data has more failure modes than MMLU.

Xiao et al. (2024) provide the finance-specific elaboration. TradingAgents constructs a trading-firm-inspired role taxonomy: bull researcher, bear researcher, risk manager, technician, fundamentalist, macro analyst. We inherit this taxonomy as the persona ensemble. Each persona has a separate calibration record (Brier score per persona), and each persona's vote is weighted by its recent calibration during decision making. Bull and bear from TradingAgents map to proposer and critic in Du's protocol; the other personas serve as additional signals fed to the judge.

Faithfulness matters when the judge inspects the proposer's reasoning chain. Chua and Evans (2025) show that R1-class reasoning models describe their cues 59 percent of the time versus 7 percent for non-reasoners. The judge therefore weights verbal reasoning traces from R1-class agents higher than from non-reasoners, and the calibration table includes a per-agent faithfulness probe so that this weighting is empirical rather than assumed.

The compute cost of a full debate cycle is roughly 3 to 4 times a single-prompt decision. The system pays this cost only for new theses entering the procedural-memory promotion pipeline, never for routine sleeve signals.

8. Component five: calibration that survives RL overconfidence

Xie et al. (2026) extract normalized confidence from output-anchor token probabilities combined with self-evaluation. They find that supervised fine-tuning calibrates well but RL methods (PPO, GRPO, DPO) induce systematic overconfidence. Post-RL SFT with self-distillation restores calibration; AUROC improves from 0.806 to 0.879.

The implication is direct. We cannot trust the verbal "I am 70 percent confident" that an R1-class agent emits. The system extracts anchor-token confidence per the Xie protocol, then applies a regime-stratified empirical recalibration map learned from the agent's prior Brier scores. Crisis-regime overconfidence is treated separately from low-volatility-regime overconfidence; the calibration map is a function of (agent_id, regime, source_class).

Lu et al. (2026) provide the source-side complement. DoublyCal calibrates retrieved evidence con-

fidence first, in a proxy model, and only then passes the calibrated evidence to a black-box LLM for final reasoning. We adopt this two-stage pattern for the research-dispatch pipeline. When the brain detects a knowledge gap and the research agent retrieves new evidence, the evidence carries a source-confidence score (from source trust × evidence quality × recency) that is computed before the proposer sees it. The hypothesis confidence is then the product of source confidence and reasoning confidence, calibrated separately and combined deterministically.

The calibration tracker stores a row per probability emission. Schema includes agent_id, stated_probability, anchor_confidence, regime, brier_score on resolution, faithfulness_probe_score, and timestamp. Sleep-loop weekly job recomputes the recalibration maps and updates per-agent confidence transformations.

9. Component six: validation via CPCV and ABM stress

Single walk-forward backtests overestimate strategy performance because they leak information through path dependency. Combinatorial Purged Cross-Validation (López de Prado) splits the sample into N folds, holds out combinations rather than single segments, purges training samples that overlap test windows, and reports a distribution of out-of-sample Sharpe estimates rather than a single number.

A hypothesis cannot promote from semantic memory to procedural memory without surviving CPCV across at least three regime classifications. Open question Q3 from the reasoning brief asks for the minimum regime count empirically; the v0 default is three.

ABM stress is the second validation layer. Discovered causal graphs and surviving hypotheses are tested against synthetic-market agent-based models that include known stress regimes (liquidity shock, volatility spike, momentum reversal). Hypotheses that depend on edges that flip under ABM perturbation are rejected as regime-fragile.

The third layer is novelty checking. The system computes maximum mean discrepancy between candidate hypothesis embeddings and the embeddings of the prior hypothesis space. A hypothesis flagged novel by MMD that simultaneously fails CPCV is treated as overfit, not innovative. A hypothesis flagged as a remix (low MMD distance) that passes CPCV is acceptable but does not earn the proposer agent novelty credit. This combination addresses open question Q5: novelty alone is not a quality signal.

10. Component seven: deployment, sizing, and operational guards

Lin et al. (2026) survey memory security across the lifecycle of write, store, retrieve, execute, share, and forget. Memory poisoning, retrieval corruption, and cross-agent propagation are real adversarial vectors when ingesting Reddit, X, and other low-trust sources. The system therefore implements write-gate validation on every episodic write, with stricter gating for sources whose dynamic trust score is below 0.4. Speculative-tier writes go to a sandboxed schema that the trade-decision

pipeline cannot read directly.

Operational guards are hard-coded in code, not in prompts. Position caps, gross and net exposure limits, drawdown circuit breakers, latency monitors, and fat-finger guards are non-overridable by any agent. The reasoning that "this trade should be larger" cannot increase a position above the cap; only a human override (Henri or brother, both required for any change) can do that. The reasoning brief makes this explicit and the security survey reinforces it.

Position sizing follows half-Kelly with risk-budget overlay. The size of a position is a function of the agent's calibrated probability of being right, the stated payoff and loss, the confidence interval on the probability, and a global risk budget that caps gross exposure. Hypotheses with EV at or below zero, or with confidence intervals that cross zero, are not deployed regardless of how confident the verbal reasoning sounds.

Every deployed strategy has a 90-day decay clock. At expiry the strategy must re-validate via CPCV on fresh out-of-sample data or be auto-retired. The default assumption is that alpha decays. Re-validation is the only override.

11. Component eight: reflective post-trade learning

Shinn et al. (2023) introduce verbal reinforcement learning. After every task, the agent reflects on the outcome in natural language, stores the reflection in an episodic buffer, and uses it on the next attempt. On HumanEval the method achieves 91 percent pass-at-1 against an 80 percent GPT-4 baseline. Park et al. (2023) generalize the pattern in a 25-agent sandbox; their three-tier loop of observation, reflection, and plan is the canonical template for our daily sleep cycle.

Zhao et al. (2024) extend this with ExpeL: agents extract natural-language insights across many trajectories and retrieve relevant ones at inference time. Allard et al. (2026) add the critical refinement that selective retrieval of insights (top-k by current-task embedding similarity) outperforms prepend-all approaches and improves Gaia results by 7.8 percent. We adopt selective retrieval as the default.

Wang et al. (2024) demonstrate that more than 60 percent of trajectories are typically discarded as failures during fine-tuning, and that prefix-tagging failures and including them in training improves downstream agent quality. The prefix-tag pattern goes into the procedural-memory pipeline directly. Failure trajectories are kept, prefix-tagged, and used during eventual fine-tuning at v3-plus.

Fang et al. (2026) provide the most operationally specific recipe. Their Trajectory-Informed Memory pipeline has four components: Trajectory Intelligence Extractor, Decision Attribution Analyzer, Contextual Learning Generator (which produces strategy tips, recovery tips, and optimization tips), and Adaptive Memory Retrieval. The three-tip taxonomy is a clean schema for our procedural-memory content types. Each procedural-memory entry carries a subtype field with

one of the three values, keyed by whether the source trajectory was successful, recovered after error, or inefficient but ultimately successful.

The post-trade reflection pipeline runs after every resolved trade. The pipeline produces: a reflective summary in natural language, an attribution of credit and blame to the contributing sources (input to the source-trust update job), a typed failure entry if the trade lost (using the AgentErrorTaxonomy of Zhu et al. 2025), and a candidate procedural-memory tip. Procedural-memory tips do not promote on first sighting; they require batch validation as described in the next component.

12. Component nine: procedural memory and skill compression

Forouzandeh et al. (2026) introduce MACLA: procedural memory via Beta-posterior Bayesian selection over candidate procedures, with contrastive refinement of preconditions and meta-procedural composition for hierarchical reasoning. MACLA compresses 2,851 trajectories into 187 reusable procedures, a roughly 15-to-1 ratio. Performance is 78.1 percent across four benchmarks and 90.3 percent on ALFWorld.

This is the canonical reference for our procedural tier. Each candidate procedure carries a Beta posterior (alpha, beta) representing successes and failures. Selection at runtime samples or maximizes expected utility over the posterior conditioned on the current task context. Contrastive refinement updates preconditions when a procedure succeeds in some contexts and fails in others, narrowing the applicability rules over time.

Su et al. (2026) provide the promotion gate. Their Mistake Notebook approach clusters failures by subject within batches, distills them into structured notes, and only commits memory updates when batch performance improves. We adopt this accept-if-improves rule for procedural promotion. A candidate procedure does not enter the procedural tier until it has improved CPCV-evaluated performance on a held-out fold over a candidate batch of trajectories. The batch size is set so that statistical noise does not falsely promote a procedure on a few lucky trades.

Zhang et al. (2023) introduce REMEMBERER, which treats LLM agents as semi-parametric reinforcement learners by updating an external experience memory rather than the model weights. The update rule for posterior shift on success versus failure transfers directly to our Bayesian procedure-selector. Memory becomes the substrate; the LLM is the activation function.

Yu et al. (2026) describe self-consolidation: contrastive reflection summarizing error-prone patterns, then distillation of non-parametric textual experience into compact learnable parameters. We do not use this at v0 through v2. The textual procedural memory is the ground truth and remains reversible; parameter distillation is non-reversible and conflicts with the "errors are training signal" pattern. Self-consolidation is on the v3-plus roadmap once the textual store is large and stable.

The procedural tier therefore contains: reusable reasoning templates (continue, skip, repeat, abort meta-policies), skill functions implemented as Python tools, prompt templates per agent role, and

failure patterns prefix-tagged for negative training. The tier is git-versioned so promotions and demotions are auditable.

13. Component ten: anomaly and regime detection

Li and Fan (2026) detect financial anomalies via a mixture-of-experts over four mechanism-specific channels: price shock, liquidity, systemic contagion, and momentum reversal. Their model identifies all six major US stress events from 2017 to 2024 with a 3.7-day mean lead time and an AUC of 0.888.

The four mechanism channels become four additional dimensions on the regime_state record. In addition to volatility regime, risk regime, dispersion regime, and trend regime, the system tracks an entity-level Market Pressure Index per the paper's protocol and a market-wide aggregate. Daily attention priors are partially driven by which mechanism channel is currently elevated.

Martinez (2026) proposes ReGEN-TAD, an interpretable ensemble combining convolutional-transformer joint forecasting with reconstruction. The aggregate score reflects predictive inconsistency, reconstruction degradation, latent distortion, and volatility shifts. The system uses ReGEN-TAD as a continuous unsupervised anomaly score that does not require labels. Crucially, the score acts as a kill-switch trigger: if the score crosses a threshold, the operational guard layer halts new position openings until a human review.

Both papers are recent (2026) and have limited cross-validation outside their original benchmarks. The integration treats their outputs as features fed to the regime-detection layer, not as ground truth. The system maintains a separate ReGEN-TAD calibration test against known anomaly periods (2008, 2020, 2022, March 2023 SVB) and downweights its contribution if calibration drifts.

14. Component eleven: source trust as dynamic learning

Sources are not pre-tiered. Each source carries a Bayesian posterior trust score that updates from evidence. When a hypothesis resolves, credit and blame are distributed proportionally to the contributing sources via Shapley-value-style attribution. Loud sources that fire on many things receive less credit per signal; specialist sources that fire rarely receive more (inverse-frequency weighting). Bad sources are demoted to a speculative tier where their signals are read but cannot found a hypothesis. Demoted sources can earn back trust by accumulating successful contributions.

The trust score is domain-specific, not just per-source. A blogger who is reliable on semiconductors but useless on macro is recorded with separate posteriors per (source × domain). Time decay applies a 90-day half-life so that a source that hit in 2018 but went quiet does not retain disproportionate authority.

The trust score parameterizes the calibration math from Component five (Lu et al., 2026). Source confidence multiplied by reasoning confidence determines hypothesis confidence. The factorization keeps the two error modes separated and traceable.

15. Sleep loop integrating the components

Asynchronous consolidation, separate from inference, is what produces learning rather than accumulation. Mem0 hits 91.6 on LoCoMo and 93.4 on LongMemEval at 3 to 4 times lower token cost than full-context approaches by running consolidation offline. Park et al. (2023) and Su et al. (2026) reinforce that batch-level consolidation with explicit accept-if-improves gates is the pattern that holds up.

Three cadences run in our system. Daily, after market close: integrate the day's events into long-term memory across each shard, cluster and compress redundant observations, refresh attention priors using the current top-N hypotheses, and generate a daily log entry in the narrative tier. Weekly, on Sunday: re-rank beliefs by recent evidence, run calibration audit (Brier scores per agent, persona, regime, and source class), update source-trust posteriors via Shapley attribution, run iterative constraint tightening on the causal graph (feed the week's falsifications back into causal-learn), promote stable patterns from episodic to semantic, and run AgentDebug-style root-cause attribution on failed trades. Monthly: regenerate the narrative self-model, run the meta-reasoning consolidation job (which reasoning approach worked best per regime), archive low-salience episodic events to cold storage, and rebuild the canonical semantic graph from scratch as a consistency check.

Park et al.'s observation, reflection, and plan loop is the per-shard daily cycle template. Su et al.'s batch-clustered failure abstraction is the weekly promotion gate. Yu et al.'s self-consolidation is the v3-plus monthly upgrade for parametric memory.

Sharded sleep processors run in parallel across signal-type slices (macro, microstructure, sentiment in v0). Each shard runs its own daily cycle on its slice. A master aggregator ingests only consolidated outputs (insights, hypotheses, failure patterns) and never raw episodic traces. The master deduplicates near-redundant outputs, resolves conflicts via confidence × evidence × recency, and writes the canonical version to the semantic graph. Zhang et al. (2025) provide the multi-agent graph hierarchy template via G-Memory, which directly addresses the conflict-resolution open question (Q2 in the reasoning brief).

16. Working memory and selective retrieval

LLM context windows are finite; long-term memory is unbounded. Working memory is the subset loaded into context for the current task. Packer et al. (2024) introduce the MemGPT pattern of virtual context paging: main context holds active state, external storage holds the rest, and the agent uses function-call APIs to page memory in and out of its window.

The system implements four context tiers. Always loaded: the manifesto and voice files, the top 10 beliefs, current regime state, open positions, today's pre-mortem reminders. Recent: the last 7 days of episodic events and the last 30 days of trade outcomes. On demand: similarity-retrieved episodic events relevant to the current task, semantic graph subgraphs queried by entity, and procedural

tips selectively retrieved per Allard et al. (2026). Archived: older or lower-salience content, tool-callable but not preloaded.

Sun and Zeng (2025) propose H-MEM with a four-level hierarchy (domain, category, trace, episode) and index-based routing rather than exhaustive similarity search. The system adopts H-MEM's hierarchy as the default semantic-graph routing, mapping domain to asset class, category to factor, trace to hypothesis, and episode to event.

The retrieval API surface follows MemGPT: read_memory, write_memory, archive_search, expressed as function calls available to every agent. This abstraction lets us swap the underlying store (pgvector to Qdrant, AGE to Neo4j) without changing agent code.

17. Six gaps in the literature that we own

The reasoning brief identifies six points where the surveyed literature stops short of what a working trading system requires. Each is a candidate edge.

First, iterative constraint tightening. The standard practice runs causal discovery once and treats the resulting graph as static. Falsification rejects a hypothesis but does not refine the graph. Our weekly sleep loop feeds falsifications back into the causal-discovery routine, rebuilding the graph as assumptions break.

Second, reasoning-pattern consolidation. Most systems consolidate facts and discard reasoning paths. We extract failure patterns from rejected reasoning trajectories using the AgentErrorTaxonomy of Zhu et al. (2025) and write them into procedural memory as negative training, prefix-tagged per Wang et al. (2024). Future agents avoid known dead ends.

Third, real-time mid-reasoning anomaly detection. Backtesting is the standard reward signal but it is lagging. We implement a real-time detector that flags when an agent's reasoning contradicts observed market behavior during reasoning, not after. The detector aborts bad reasoning mid-flight. ReGEN-TAD's ensemble anomaly score from Martinez (2026) is the upstream signal.

Fourth, multi-scale reasoning consistency. Most systems reason at one abstraction level. We decompose into hierarchical scales (macro regime, sector, microstructure) and run cross-scale consistency checks: a micro hypothesis cannot contradict the active macro hypothesis without explicit reconsolidation.

Fifth, in-loop symbolic constraint propagation. Formal verification is typically post-hoc and expensive. We check known economic and microstructure constraints during hypothesis generation rather than after, so that infeasible hypotheses are pruned before full reasoning is invested. This requires an explicit constraint graph (no-arbitrage relationships, balance sheet identities, microstructure invariants).

Sixth, meta-reasoning consolidation. Systems repeat the same expensive search every time. We extract meta-knowledge (which reasoning approach worked best under which regime) during the

monthly sleep cycle and inject it at runtime as a prior over which agent persona to weight more heavily.

18. Validation of the system itself

The system needs its own evaluation harness, not just trade-level backtests. Three layers of evaluation run continuously.

Memory layer: synthetic recall queries test whether the brain can retrieve known events from a held-out subset of its own history. Pink et al.'s five episodic-memory properties serve as the acceptance test schema (single-shot encoding rate, instance specificity hit rate, time-stamp recall accuracy, contextual binding fidelity, replayability success rate). The StuLife benchmark from Cai et al. (2025), adapted to a financial-history simulator, provides a long-horizon test.

Reasoning layer: the bull/bear/judge ensemble runs on historical case studies (Lehman 2008, March 2020, March 2023 banking) where the correct call is known and reasoning chains can be evaluated against the actual outcome. The faithfulness probe of Chua and Evans (2025) measures whether agents' verbal explanations match the cues that actually drove their answers.

Trade layer: CPCV across at least three regimes is the binding constraint for procedural promotion. ABM stress tests run on a synthetic market simulator with known regime injections. Live paper trading on IBKR for a minimum of 90 days before any capital deployment.

The brain stores its own track record on its own predictions. Calibration curves, faithfulness scores, regime-conditioned Brier scores, and source-trust trajectories are all published internally weekly and externally monthly. The published curves are the receipts; the brain's known errors are part of the moat because most AI funds will not publish their wrong calls.

19. Build sequence

The architecture is dense. The build is incremental. Eight weeks at toy scale is sufficient to demonstrate every component on a small data footprint. The cost is on the order of 1,500 dollars of compute and infrastructure, which fits on a personal credit card. Scale comes from capital, not from architectural change.

Week one: Postgres with pgvector and AGE extensions, an Obsidian vault for narrative tier, a git repo for procedural tier, a three-agent skeleton (researcher, trader, PM), and an IBKR paper account.

Week two: a 150-source ingestion pipeline pulling from RSS, free APIs, and targeted scrapers; the source_signals table; initial source-trust scaffold with static priors; episodic writes with confidence and validity windows.

Week three: episodic retrieval API following MemGPT's read_memory and write_memory pattern; pgvector kNN search; coverage-aware retrieval that returns gap_detected when confidence is low;

the working-memory tier loader.

Week four: semantic graph in AGE with H-MEM hierarchy; Graphiti-style validity windows on facts; initial causal-discovery scaffold using causal-learn library defaults; CD-NOTS as the v0 default.

Week five: sharded sleep processors (three shards by signal type); daily and weekly sleep loops; Park-style observe-reflect-plan inside the daily cycle; calibration tracker writing per-prediction Brier scores; source-trust Bayesian update job.

Week six: multi-agent debate layer (proposer, critic, judge, red team); TradingAgents persona ensemble (bull, bear, risk, technician, fundamentalist, macro); research-dispatch agent for knowledge-gap filling; failure-pattern extraction using AgentErrorTaxonomy.

Week seven: counterfactual replay engine; CPCV backtest harness; pre-mortem requirement enforced at hypothesis-promotion time; ABM stress test rig; iterative constraint tightening on the causal graph.

Week eight: investor-facing UI showing brain reasoning, sleep cycle visualization, and live paper trade log; MMD novelty validator; mid-reasoning anomaly detector wired to ReGEN-TAD; 30-day backfill demo.

Beyond week eight, the priority order is: live capital deployment with conservative caps, multimodal embedder via LANISTR, fine-tuning curriculum using prefix-tagged failure trajectories per Wang et al. (2024), parameter distillation via self-consolidation per Yu et al. (2026), and migration of the vector store and graph to dedicated systems when scale forces it.

20. Open technical questions

Six questions are unresolved at the architectural level. Each will be answered empirically during the build.

What throughput does the consolidation pipeline need to hit, in observations per second, to keep up with the data feed? CD-NOTS runs are weekly so the bottleneck is daily ingestion plus consolidation, but we have not yet stress-tested.

How is conflict resolution between sharded brains weighted? The default is confidence × evidence-quality × recency, but empirical tuning is required.

What is the minimum out-of-sample regime count required before a hypothesis can promote from semantic to procedural memory? The default is three; this may be too lax during stable regimes and too strict during turbulent ones.

What MMD distance threshold separates novel from remixed? This is an empirical calibration on the existing hypothesis embedding space.

How is a hypothesis that is novel by MMD but produces poor out-of-sample CPCV results to be classified? Likely overfit. The system rejects this combination by default but the rejection threshold needs tuning.

How does formal step-wise verification (HERMES, FVEL, StepProof) adapt to probabilistic reasoning? Lean 4 patterns assume deterministic logic; financial reasoning is probabilistic. The integration here is genuinely open.

Three additional questions are operational. Compute-wise, the multi-agent debate cost of 3 to 4 times per decision is acceptable for new theses but the sleep-loop frequency depends on the same compute pool; we need a budget split. Regime detection: separate ML model (TALON, Wang LLM-Prompt) versus LLM judgment; the v0 default is LLM judgment with TALON as a candidate forecaster head, but a separate small classifier may win on cost. Persona ensemble: prompt variants at v0, fine-tunes at v3-plus; the transition criterion is sample size on the calibration tracker.

21. Closing

The combined architecture has eleven components and four supporting systems (sleep loop, working memory, sharded processors, source trust). Every component is anchored to specific work in the library. The pieces are individually validated; the integration is the contribution.

Three properties make this design distinct from existing AI fund attempts. First, reasoning is forced off the LLM training distribution by Stage-one causal discovery on the actual data, so the model cannot default to remembered correlations. Second, every decision passes through adversarial debate with explicit calibration, so single-prompt confidence cannot drive deployment. Third, memory persists, decays, consolidates asynchronously, and assigns credit to sources, so the system actually learns from outcomes rather than restarting amnesic.

The build is feasible at toy scale on a two-person team and roughly 1,500 dollars per month. Capital scales the data firehose, the compute budget, and the universe; capital does not change the architecture. A working v3 demo is 8 weeks away.

References

(Author year, source folder. Full PDFs in `/research/`.)

Allard, Teinturier, Xing, Viaud (2026). Experiential Reflective Learning for Self-Improving LLM Agents. `07_agent_architectures/`.

Cai et al. (2025). Building Self-Evolving Agents via Experience-Driven Lifelong Learning (ELL/StuLife). `07_agent_architectures/`.

Chua, Evans (2025). Are DeepSeek R1 and other reasoning models more faithful? `05_skepticism_llm_reasoning/`.

D'Alessandro, Calabrés, Elkano (2024). MAGNUM: A Modular End-to-End Multimodal Learning Method. 10_other/.

DeepSeek-AI (2025). DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. 02_reasoning_frameworks/.

Du, Li, Torralba, Tenenbaum, Mordatch (2023). Improving Factuality and Reasoning in Language Models through Multiagent Debate. 02_reasoning_frameworks/.

Ebrahimi, Arık, Dong, Pfister (2024). LANISTR: Multimodal Learning from Structured and Unstructured Data. 10_other/.

Fang et al. (2026). Trajectory-Informed Memory Generation for Self-Improving Agent Systems. 07_agent_architectures/.

Ferdous, Hasan, Gani (2023). eCDANs: Efficient Temporal Causal Discovery from Autocorrelated and Non-stationary Data. 03_causal_discovery/.

Forouzandeh, Peng, Moradi, Yu, Jalili (2026). MACLA: Learning Hierarchical Procedural Memory for LLM Agents. 01_memory_architectures/.

Guo et al. (2023). Empowering Working Memory for Large Language Model Agents. 01_memory_architectures/.

Li, Fan (2026). Explainable Heterogeneous Anomaly Detection in Financial Networks. 09_finance_markets/.

Lin, Li, Chen (2026). A Survey on the Security of Long-Term Memory in LLM Agents. 01_memory_architectures/.

Liu, Cao, Deng (2024). Multimodal Financial Foundation Models: Progress, Prospects, Challenges. 09_finance_markets/.

Lu et al. (2026). Double-Calibration: Towards Trustworthy LLMs via Calibrating Knowledge and Reasoning Confidence. 08_validation_novelty/.

Martinez (2026). ReGEN-TAD: An Interpretable Ensemble-Based Generative Framework for Anomaly Detection. 09_finance_markets/.

Packer et al. (2024). MemGPT: Towards LLMs as Operating Systems. 01_memory_architectures/.

Park, O'Brien, Cai, Morris, Liang, Bernstein (2023). Generative Agents: Interactive Simulacra of Human Behavior. 07_agent_architectures/.

Pink, Wu, Vo, Turek, Mu, Huth, Toneva (2025). Position: Episodic Memory is the Missing Piece for Long-Term LLM Agents. 01_memory_architectures/.

Robust Time Series Causal Discovery for Agent-Based Model Validation (provenance to verify). 03_causal_discovery/.

Sadeghi, Gopal, Fesanghary (2024). Causal Discovery in Financial Markets: A Framework for Nonstationary Time-Series Data. 03_causal_discovery/.

Shinn et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. 02_reasoning_frameworks/.

Shojaee et al. (2025). The Illusion of Thinking. 05_skepticism_llm_reasoning/.

Su et al. (2026). Mistake Notebook Learning. 07_agent_architectures/.

Sun, Zeng (2025). H-MEM: Hierarchical Memory for High-Efficiency Long-Term Reasoning. 01_memory_architectures/.

Sun et al. (2025). TALON: Adapting LLMs to Time Series Forecasting. 09_finance_markets/.

Vilella et al. (2024). Anomaly detection in cross-country money transfer temporal networks. 09_finance_markets/.

Wang, Li, Han, Zhang, Baldwin (2024). Learning From Failure: Integrating Negative Examples when Fine-tuning LLMs as Agents. 07_agent_architectures/.

Wang, Li, Lan (2025). LLM-Prompt: Integrated Heterogeneous Prompts for Time Series Forecasting. 09_finance_markets/.

Xiao, Sun, Luo, Wang (2024). TradingAgents: Multi-Agents LLM Financial Trading Framework. 09_finance_markets/.

Xie, Liu, Yao (2026). Know When You're Wrong: Aligning Confidence with Correctness for LLM Error Detection. 08_validation_novelty/.

Yao, Yu, Zhao, Shafran, Griffiths, Cao, Narasimhan (2023). Tree of Thoughts. 02_reasoning_frameworks/.

Yu, Zhu, Xie, Shao (2026). Self-Consolidation for Self-Evolving Agents. 07_agent_architectures/.

Zhang, Chen, Zhang, Xu, Zhao, Yu (2023). Large Language Models are Semi-Parametric Reinforcement Learning Agents (REMEMBERER). 01_memory_architectures/.

Zhang, Fu, Wan, Yu, Wang, Yan (2025). G-Memory: Tracing Hierarchical Memory for Multi-Agent Systems. 01_memory_architectures/.

Zhao et al. (2024). ExpeL: LLM Agents Are Experiential Learners. 07_agent_architectures/.

Zhao et al. (2026). Is Chain-of-Thought Reasoning of LLMs a Mirage? 05_skepticism_llm_reasoning/.

Zhu et al. (2025). Where LLM Agents Fail and How They can Learn From Failures. 07_agent_architectures/.