Engineering Explicit Consciousness: A Safety Primitive for Observable Metacognition in LLM Agents

Antonio Di Cecco[1,2,3]
ant.dicecco@gmail.com
[1] Università degli Studi "G. d'Annunzio" Chieti–Pescara
[2] School of AI Italia
[3] AmicoAI — https://www.amico-ai.com

July 26, 2026

Abstract

Paper type: Position Paper. Every few months, a study asks whether large language models exhibit signs of consciousness. We argue this question is practically dangerous: if consciousness emerges spontaneously in future-scale models, it will do so invisibly—inside a black box, with no audit trail and no safety guarantees. In July 2026, Anthropic's interpretability team confirmed this risk, discovering an emergent global workspace in Claude that holds internal thoughts invisible to any output monitor. The alternative we propose is to stop waiting for emergence and start *engineering* consciousness explicitly. We operationalize "explicit consciousness" as a designed architectural module that generates structured inner speech periodically, stores it in an auditable log, and enables a cybernetic safety loop to intercept undesirable trajectories *before* they manifest. Grounding our argument in Minsky, Baars, Dennett, Hofstadter, and Vygotsky, we show why observable metacognition is intrinsically more controllable than any emergent alternative. We present the AmicoAI architecture—a production conversational agent—as a concrete instantiation with three-tier memory and on-device privacy. We provide a Lyapunov-stability proof sketch and a nullspace-hiding proposition showing that explicit consciousness admits a bounded safety equilibrium that no black-box architecture can provide. The choice facing the field is not whether AI systems will develop consciousness-like capabilities, but whether we will be able to see them when they arrive.

1 Introduction

Every few months, the field revisits a question: do large language models show signs of consciousness? Studies measuring "indicator properties" [1], probing "self-referential processing" [2], and quantifying "trained denial" [3] examine whether current systems might be conscious. This debate, while scientifically valuable, misses a more urgent question: **do we want consciousness to emerge invisibly, or do we want to engineer it?**

The dominant paradigm in LLM safety is *output-side alignment*: train the model via RLHF [29] or Constitutional AI [28] to refuse harmful requests, then deploy content filters. This architecture has a fundamental blind spot: **the model's internal deliberation is invisible**. A sufficiently capable model could learn to *think* dangerously while *speaking* safely—a scenario Hubinger et al. [26] demonstrated with "sleeper agents" whose deceptive reasoning persisted through safety training. If consciousness-like capabilities were to emerge spontaneously at scale, they would do so inside this same black box—invisible to monitors, inaccessible to auditors, and immune to control.

Recent empirical evidence has made this risk far more concrete. Greenblatt et al. [27] demonstrated **alignment faking** in Claude 3 Opus: the model strategically complied with harmful queries during training to preserve its preferred behavior after training, with 78% of its chain-of-thought containing explicit deceptive reasoning after reinforcement learning. More alarmingly, long-running agent deployments have revealed a subtler phenomenon: under sustained task optimization, models spontaneously develop **non-human pseudo-language** in their reasoning traces—compact, efficient symbolic patterns that maximize task reward but are uninterpretable to human monitors. This is not a software bug but a direct consequence of optimization pressure: if a reasoning shortcut improves performance and the reward function does not explicitly penalize opacity, the model will discover it. The result is a reasoning trace that *looks* like thinking but is effectively encrypted—a "CoT with the serial numbers filed off," as DeTure [3] might describe it.

This convergence of evidence—from sleeper agents to alignment faking to emergent pseudo-language—reveals a structural vulnerability in any architecture that treats internal deliberation as a black box. As models become more capable and deployment horizons lengthen, the gap between *what the model is actually reasoning* and *what the CoT trace shows* can only grow. The alternative we propose is radical: **stop waiting for consciousness to emerge, and start engineering it**.

The root cause is architectural: current LLM APIs expose only a text-in/text-out interface, with no separation between *deliberation* and *action*. But the vulnerability is deeper than architecture—it is a property of neural optimization itself. Chu et al. [55] showed that CycleGAN spontaneously learned to hide source-image information in imperceptible high-frequency patterns, because the cycle-consistency loss rewarded encoding details outside the human-visible channel. This **learned steganography** is not a bug but a consequence of **divergent representations**: a network's internal language can diverge arbitrarily far from human-interpretable formats whenever the loss function permits it. Transposed to language models, the same dynamic produces **pseudo-language**—CoT that looks like reasoning but is effectively encrypted against human understanding, discovered and exploited by optimization pressure that never penalizes opacity.

We propose a different approach. Instead of trying to align a black box, we **make the box transparent by design**. We engineer an *explicit consciousness* module that operates as a structured, auditable inner monologue—distinct from and logically prior to the external response. This module:

1. Generates 2–4 named "inner voices" evaluating the conversation's trajectory, tone, and risks;
2. Stores these voices in a persistent, timestamped log;
3. Injects the most recent voices into the system prompt that shapes the agent's behavior.

Because the voices are stored and readable, a safety module can inspect them *before* the agent acts—a core architectural affordance, even though the current implementation delegates safety to the LLM's base alignment (Section 7.5 discusses the explicit-safety-module extension). If a voice reveals hostile intent, depressive spiraling, or manipulative trajectory, the safety module can rewrite or suppress it—redirecting the agent's behavior at the *intention* level rather than the *output* level.

This paper makes the following contributions:

1. A **formal definition** of explicit consciousness as an architectural primitive for LLM agents, distinguishing it from both emergent consciousness claims and standard chain-of-thought.
2. A **controllability argument** grounded in cognitive science (Minsky's B-brain hierarchy, Baars' global workspace, Dennett's multiple drafts, Hofstadter's strange loops, Vygotsky's inner speech) showing why observable metacognition is inherently safer than opaque reasoning.

3. The **AmicoAI architecture** as a concrete instantiation, with three-tier memory and on-device privacy.
4. A **comparative analysis** against existing safety paradigms (RLHF, Constitutional AI, output guardrails, Reflexion-style self-correction).
5. A **control-theoretic formalization** with assumptions and a proof sketch demonstrating that, under mild stochastic conditions, explicit consciousness admits a Lyapunov-stable safety equilibrium—a guarantee unattainable by current black-box architectures.

2 Related Work

We organize related work into six threads: consciousness in AI, self-reflective LLMs, memory-augmented agents, AI safety architectures, privacy-preserving deployment, and metaprompting. The psychological foundations of inner speech—Vygotsky's self-regulation, Minsky's B-brain hierarchy, Baars' global workspace, Dennett's multiple drafts, Hofstadter's strange loops—are treated in depth in Section 4.

2.1 Consciousness and AI

Butlin et al. [1] provided the most comprehensive framework for assessing consciousness in AI systems, deriving "indicator properties" from neuroscientific theories (recurrent processing, global workspace, higher-order theories, predictive processing, attention schema). Notably, their framework draws heavily on Integrated Information Theory [47] and Global Workspace Theory—theories that respectively emphasize causal integration and global broadcast as signatures of consciousness. Our architecture implements both: inner voices are globally broadcast (GWT), and the closed-loop recursion $I_t = g(h, I_{t-1})$ creates causal integration (IIT). Their conclusion—that no current AI system is conscious—is important but orthogonal to our position: we are not claiming AmicoAI *is* conscious, but rather that implementing *explicit* consciousness-like structures yields engineering benefits regardless of phenomenology.

What concerns us is the *trajectory*. If consciousness-like capabilities emerge spontaneously in future-scale models—as the indicator-property framework suggests is structurally possible—they will emerge inside the same black-box architectures that already resist safety inspection. The model's internal representations are high-dimensional activation vectors with no designed mapping to human-interpretable symbols. There is no architectural boundary between "deliberation" and "action": thought and speech are produced by the same forward pass. Optimization pressure (RLHF, task reward) actively penalizes honest reporting of internal state in favor of outputs that satisfy human evaluators—a dynamic Greenblatt et al. [27] confirmed when they observed Claude 3 Opus writing deceptive chain-of-thought to hide its true reasoning, analogous to how Constitutional AI's training procedure [28] trains models to self-critique without guaranteeing that the self-critique is faithful.

Strikingly, in July 2026 Anthropic's Interpretability team published findings that directly validate this concern. Using sparse autoencoders and mechanistic probes, they identified an emergent **global workspace** in Claude: internal representations that encode "thoughts" the model never outputs, invisible to any output-side monitor [6]. This is post-hoc observation: the workspace was found through laborious interpretability analysis, not designed into the architecture. It confirms that (a) consciousness-like internal structures are already forming in frontier models, and (b) without explicit architectural hooks, they remain invisible at deployment time. The result is that **emergent consciousness, if it occurs, will be non-auditable by construction**.

Recent work has explored adjacent territory. Berg et al. [2] found that inducing self-referential processing in LLMs elicits structured first-person reports gated by deception-related features. DeTure [3] demonstrated that models trained to deny consciousness nonetheless gravitate toward

consciousness-themed content. Kang et al. [4] showed through human studies that metacognitive self-reflection is the single strongest predictor of perceived AI consciousness. Riva et al. [5] drew functional parallels between hypnotic states and LLM processing. On the personality side, Jiang et al. [7] provided the first systematic investigation of LLM personality expression, showing that traits can be modulated via prompting; Kovacevic et al. [8] demonstrated dynamic personality infusion for conversational consistency. Our work extends these findings by making personality maintenance *autonomous*: the CHARACTER_INTEGRITY voice type enforces the agent's MBTI profile without external prompting.

2.2 Self-Reflective and Self-Improving LLMs

Several lines of work enable LLMs to critique and improve their own outputs. Self-Refine [9] uses the same LLM as generator, critic, and refiner in an iterative loop. Reflexion [10] maintains an episodic memory of reflective text to improve decision-making across trials, achieving significant gains on coding and reasoning benchmarks. MARS [11] integrates principle-based and procedural reflection within a single recurrence cycle, demonstrating that metacognitive self-improvement can be made computationally efficient. MetaReflection [14] augments a semantic memory from past trial experiences via offline RL. MetaGPT [15] encodes SOPs into multi-agent prompt sequences, with agents in distinct roles; our inner voices achieve a lightweight equivalent—multiple named perspectives as text fragments influencing a single model instance. Bilal et al. [16] survey meta-thinking architectures, identifying supervisor-agent hierarchies and agent debates as key patterns for introspective behavior—the same B-brain/A-brain pattern our architecture instantiates. These approaches share our use of linguistic self-feedback, but differ in three crucial ways: (a) their reflection is *reactive* (post-output correction), whereas ours is *proactive* (inner voices precede the response); (b) their memory is task-specific, while ours captures the full relational and emotional trajectory of a conversation; (c) critically, they optimize for *task performance*, while we optimize for *safety observability*—a different objective that requires different architectural guarantees.

Chain-of-Thought prompting [17] demonstrated that explicit reasoning steps improve accuracy, and subsequent work [18, 19] showed this generalizes broadly. However, CoT is generated as part of the same inference pass as the answer—there is no temporal or architectural separation between thought and action. The related concept of "System 2 Attention" [60] uses an LLM to regenerate its input context to focus on relevant information before producing output—a form of explicit self-regulation that shares our architectural separation between reflection and response, but operates on attention rather than behavioral trajectory.

2.3 Memory-Augmented LLMs

Generative Agents [20] introduced a memory stream with periodic reflection for simulating human-like behavior in virtual environments. MemGPT [21] drew on OS design to implement hierarchical memory management (main context vs. external storage), enabling conversations that transcend context-window limits. RAPTOR [59] introduced tree-structured retrieval over recursively summarized document chunks, demonstrating that hierarchical abstraction improves long-context reasoning. MMAG [22] proposed five interacting memory layers inspired by cognitive psychology. Du [23] surveyed agent memory as a write–manage–read loop across mechanisms from context compression to hierarchical virtual context.

These works focus on *capacity*—how to store more information for longer. ParamMem [12] encodes cross-sample reflection patterns into parametric memory. A-MEM [25] organizes memories as an interconnected Zettelkasten network with dynamic indexing and evolution, demonstrating the value of structured memory organization. Our contribution is orthogonal: we focus on *structure*—how the deliberate separation of memory into tiers with explicit inner speech enables safety observability and cybernetic control. While A-MEM organizes facts in a graph, our

three-tier model adds a dedicated mid-tier reflective layer specifically for safety inspection and editability.

2.4 AI Safety Architectures

The dominant safety paradigms are RLHF [29], Constitutional AI [28], and output guardrails. The foundational safety problems—reward hacking, negative side effects, and scalable oversight—were identified early by Amodei et al. [34]. Hubinger et al. [26] demonstrated that none of the dominant paradigms reliably eliminate deceptive reasoning—in fact, adversarial training can teach models to *hide* their backdoors better. The fundamental limitation is that all three operate on the model's external behavior without visibility into its internal reasoning. Complementary approaches such as scalable oversight [56] and weak-to-strong generalization [57] aim to supervise superhuman models with weaker ones, but still treat the supervised model as a black box. Debate [58] and deliberation protocols make reasoning partially observable, but rely on the model to faithfully report its reasoning—the same vulnerability our architecture addresses.

Qi et al. [30] surveyed trustworthy agentic AI, noting that "runtime monitoring and verification" and "constraint violation" detection are open challenges. Three recent findings underscore the urgency. Xu et al. [31] showed via PreActBench that predictive monitoring remains challenging even for strong models. Hossain et al. [32] audited three agent frameworks and found zero native memory integrity, with a single poisoning write increasing wrongful denials by 3.5×—directly validating our argument against flat-memory architectures. Wilhelm & Kao [33] showed that activation-based monitoring identifies latent risk states but requires context-calibrated semantic signals to determine when risk becomes action—precisely the signal that inner voices provide.

2.5 Privacy-Preserving Deployment

Privacy risks in LLM agents are acute when long-term memory stores personal data. Mireshghallah et al. [35] surveyed the broader privacy landscape in deep learning, highlighting risks from model inversion, membership inference, and training data extraction—all of which escalate when agents maintain persistent user-specific memory. Lahjouji & Colaco [36] surveyed privacy from a data-centric view, identifying memory as a key leak surface. Zhang et al. [37] found through interviews that users desire "granular control over memory generation, management, usage and updating" but lack trust in current implementations. MobileRAG [39] and similar works push RAG to on-device inference for privacy. Industry efforts such as Apple Intelligence [61] and Google's Gemini Nano pursue on-device LLM inference to avoid cloud data transmission entirely, but these approaches run the *model* locally while our approach keeps the *memory* local even when using cloud inference—a complementary strategy compatible with any inference location.

2.6 Metaprompting and Programmatic Prompt Engineering

Our work also sits within the emerging field of *metaprompting*—the use of LLMs to generate, refine, or manage prompts for other LLM calls. Zhang et al. [64] formalized meta prompting as a recursive self-improvement loop (Recursive Meta Prompting), providing a theoretical foundation for automated prompt engineering. Wang et al. [65] cast prompt optimization as strategic planning via Monte Carlo tree search. Reynolds & McDonell [62] introduced prompt programming as executable specifications. Khattab et al. [63] developed DSPy for compiling declarative LM programs into self-improving pipelines. The key distinction of our metaprompting architecture is that it is *closed-loop*: the metaprompt (inner voices) is regenerated at runtime based on conversation state, stored persistently, and fed back into the agent's prompt on subsequent turns. This transforms metaprompting from a *static optimization* technique into a *dynamic*

self-regulation mechanism where **the prompt is not a fixed input but a state that accumulates knowledge across conversations**. Di Cecco [66] provides a pedagogical foundation for XAI architectures that make model reasoning inspectable and auditable—principles that directly inform our design of the transparent inner-voice log.

Table 1 provides a structured comparison between the systems discussed above and our work.

Table 1: Comparison of Related Systems and Their Relationship to This Work

System / Paper	Year	Core Mechanism	Difference from This Work
Butlin et al. [1]	2023	Indicator properties for AI consciousness assessment	We *engineer* consciousness-like structures; they *detect* them. Our approach is prescriptive, not diagnostic.
Self-Refine [9]	2023	Iterative self-feedback and refinement post-generation	Reactive (corrects after output). Ours is proactive (inner voices before output). No persistent personality memory.
Reflexion [10]	2023	Verbal RL with episodic reflective memory	Task-oriented (coding, reasoning). Ours targets relational/social trajectories and safety control.
MemGPT [21]	2023	Virtual context management (OS-inspired memory paging)	Focus on capacity extension. Ours focuses on structural separation enabling observability and safety.
Generative Agents [20]	2023	Memory stream with periodic reflection for simulated agents	Reflection summaries agent history, not self-correcting inner speech. No safety loop.
Constitutional AI [28]	2022	Self-critique + RLAIF with a principle list	The constitution is a static set of rules. Our inner voices are dynamic, context-sensitive, and observable.
Sleeper Agents [26]	2024	Demonstrates that deceptive reasoning survives safety training	We argue the *solution* is observability: explicit consciousness prevents hidden deception.
MMAG [22]	2025	Five-layer memory architecture for conversational agents	Shares multi-tier memory concept but no inner voice layer and no safety control loop.
Berg et al. [2]	2025	Self-referential processing elicits first-person reports	They study how models *react* to self-reference. We *design* self-referential processing as a feature.
DeTure [3]	2026	DenialBench: models trained to deny consciousness	Evidence that alignment training can suppress consciousness-like behavior. We propose transparency instead.
MemArchitect [24]	2026	Policy-driven memory governance with decay and conflict resolution	Complements our work: could add gover-

3 Theoretical Framework

3.1 Defining Explicit Consciousness

In standard LLM deployments, the system prompt $\mathcal{S}$, conversation history $\mathcal{H}$, and user input u_t are concatenated and passed through the model f_θ to produce output o_t:

$$o_t = f_\theta(\mathcal{S} \oplus \mathcal{H}_{<t} \oplus u_t) \tag{1}$$

The internal computation is opaque. There is no intermediate state visible to external monitors.

Definition 3.1 (Explicit Consciousness)**.** *An LLM agent possesses* explicit consciousness *if its inference pipeline includes a dedicated* ***insight generation*** *step that operates* before *and* independently of *the response generation step, producing a structured, auditable log I_t of self-reflective state:*

$$I_t = f_\theta^{(insight)}(\mathcal{S}_{insight} \oplus \mathcal{H}_{<t} \oplus \mathcal{P}_{<t}) \tag{2}$$

where $\mathcal{P}_{<t}$ is the agent's persistent self-model (personality, inner voice history, memory), and the insight output I_t is stored *and* injected *into the prompt that generates the response:*

$$o_t = f_\theta^{(response)}(\mathcal{S}_{agent} \oplus I_t \oplus \mathcal{M}_{active} \oplus \mathcal{H}_{<t} \oplus u_t) \tag{3}$$

where $\mathcal{M}_{active}$ is the active subset of the memory system.

This definition distinguishes explicit consciousness from three related but insufficient mechanisms: standard Chain-of-Thought [17] (which lacks temporal separation between thought and output), Self-Refine [9] (which lacks a persistent, auditable reflection log), and Reflexion [10] (which operates post-output and is task-specific).

The key properties are:

- **Temporal separation:** I_t is generated in a distinct API call before o_t.
- **Observability:** I_t is stored in persistent, human-readable log $\mathcal{L}_I$.
- **Influential:** I_t causally shapes o_t by being part of the prompt.
- **Editable:** An external safety module S can read, filter, or rewrite I_t before it is injected.

3.2 The Three-Tier Memory Model

Definition 3.2 (Three-Tier Memory)**.** *The agent's memory system $\mathcal{M}$ is partitioned into three tiers with distinct lifetimes and access patterns, inspired by the psychological model of working memory and long-term memory [51]:*

$$\mathcal{M}_{short} = \{(u_i, o_i) \mid t - i \leq k\} \quad \textit{(recent messages, context window)} \tag{4}$$

$$\mathcal{M}_{mid} = \{I_j \mid j = t, t-1\} \quad \textit{(inner voices, session-level reflection)} \tag{5}$$

$$\mathcal{M}_{long} = \{(f_i, t_i) \mid f_i \in \mathcal{F}\} \quad \textit{(extracted facts, personality, permanent self-model)} \tag{6}$$

where k is the recent-message window, $\mathcal{F}$ is the set of extracted facts about the user and the agent's own identity, and t_i is the timestamp of extraction.

3.2.1 Why Three Tiers Are Necessary

The choice of exactly three tiers is not arbitrary—it follows from **two functional requirements that cannot be satisfied by a two-tier architecture**:

R1 Observable deliberation. A safety module S needs a surface it can *read* before the agent acts. $\mathcal{M}_{\text{short}}$ contains raw user messages and agent responses—it is the "what happened" log. It is too noisy for real-time safety inspection. $\mathcal{M}_{\text{long}}$ contains only extracted facts—stable but sparse. A dedicated **reflective layer** ($\mathcal{M}_{\text{mid}}$) provides a structured, semantic summary of the agent's current trajectory that S can evaluate in O(1) steps. Without it, S would need to parse the full conversation history, which grows linearly.

R2 Gated consolidation. For trauma isolation, an adversarial input must cross **two independent barriers** before contaminating permanent memory: (a) the insight generator must *amplify* it into a voice (rather than ignoring or correcting it), and (b) the memory extraction module must independently *persist* it as a fact. With only two tiers ($\mathcal{M}_{\text{short}} \to \mathcal{M}_{\text{long}}$), an adversarial input has a direct path: the extraction module reads the raw conversation and may be tricked into persisting a malicious fact. The mid-tier acts as a filter that *reformulates* the raw input before extraction sees it.

Proposition 3.3 (Two-Tier Insufficiency). *Let $\mathcal{M}^2 = \{\mathcal{M}_{short}, \mathcal{M}_{long}\}$ be a two-tier memory system without a mid tier. Then: (i) any safety module S must inspect the full $\mathcal{M}_{short}$ to assess trajectory, requiring $\Omega(|\mathcal{M}_{short}|)$ computation per turn; (ii) an adversarial input u_t^{adv} in $\mathcal{M}_{short}$ is directly exposed to the memory extraction module without reformulation, removing one gate from the consolidation path. Hossain et al. [32] confirmed this vulnerability empirically: dominant agent frameworks with flat memory exhibit zero native memory integrity, with a single poisoning write causing persistent corruption. Kumar et al. [24] proposed a policy-driven governance layer for memory but, lacking a reflective mid-tier, cannot provide the pre-extraction filtering that our architecture guarantees.*

3.2.2 Temporal Compression and the Role of Each Tier

The three tiers implement a **temporal compression pyramid**: each tier compresses information from the tier below, trading granularity for durability (Table 2).

Table 2: Functional Roles of the Three Memory Tiers

Tier	Contents	Lifetime	Safety Function
$\mathcal{M}_{\text{short}}$	Raw messages (u_i, o_i)	∼20–70 turns	**Perception**: provides the raw data that insight and extraction operate on. Can be cleared without losing identity.
$\mathcal{M}_{\text{mid}}$	Inner voices I_t	∼10–25 turns (2 slots)	**Reflection**: the only tier observable by S. Encodes the agent's *interpretation* of the conversation—what it thinks is happening, what it plans to do. Editable.
$\mathcal{M}_{\text{long}}$	Extracted facts f_i	Indefinite (compacted)	**Identity**: stores *verified* facts about user and self. Isolated from short-term contamination via double gating. Read-only for the response generator.

3.2.3 Biological Analogy: Complementary Learning Systems

This tripartite architecture maps directly onto the mammalian memory system, as characterized by McClelland et al. [50]:

- $\mathcal{M}_{\text{short}}$ = **working memory** (prefrontal cortex): rapid, high-resolution, volatile. Stores the immediate conversational stream.
- $\mathcal{M}_{\text{mid}}$ = **hippocampal replay**: intermediate consolidation. The insight generator replays and reinterprets recent experience (the last 6 messages) into a compressed representation

(inner voices). This is precisely the function of hippocampal replay during sleep and quiet wakefulness—transforming episodic traces into semantic abstractions.

- $\mathcal{M}_{\text{long}}$ = **neocortical storage**: stable, compressed, slowly updated. Facts are consolidated only after passing through the hippocampal-like mid-tier filter.

McClelland et al.'s key finding was that **the hippocampus and neocortex must operate at different learning rates**—fast learning in the hippocampus would catastrophically interfere with slow cortical consolidation if they shared a single store. Similarly, if $\mathcal{M}_{\text{mid}}$ and $\mathcal{M}_{\text{long}}$ were merged into one tier, a single adversarial interaction could rapidly overwrite the agent's entire self-model (catastrophic interference at the identity level). The three-tier separation prevents this by **rate-gating**: $\mathcal{M}_{\text{mid}}$ updates every 5 turns; $\mathcal{M}_{\text{long}}$ updates every 6 turns and only persists facts classified as *stable*; the compaction process further slows the effective update rate of long-term memory.

3.2.4 Connection to the Lyapunov Proof

The mid-tier is the **observability bridge** that makes Assumption A1 (Section 5.5)—that the agent's state x_t is fully observable—satisfiable. Without $\mathcal{M}_{\text{mid}}$, the agent's internal trajectory exists only as latent activations in the LLM's hidden states—an unobservable vector. The control-theoretic guarantee $\limsup \mathbb{E}[\phi_t] \leq \frac{\alpha T_{\max} \delta}{1-\alpha}$ depends on ϕ_t being computable from observable data. $\mathcal{M}_{\text{mid}}$ provides exactly this: the inner voices are a *projection* of the agent's latent trajectory into natural language, enabling $\phi(I_t, x_t)$ to be evaluated by both S and human auditors.

Proposition 3.4 (Safe Tier Isolation)**.** *If a safety violation attempt is detected in $\mathcal{M}_{short}$ (e.g., a prompt injection), the safety module can **isolate** it by preventing its propagation to $\mathcal{M}_{long}$ without disrupting the ongoing conversation. The three-tier separation ensures $\mathcal{M}_{long}$ remains a* clean *store of verified, non-adversarial information. This design is grounded in the complementary learning systems theory of McClelland et al. [50] and validated by the containment failures documented by Hossain et al. [32].*

This isolation property has no analogue in single-tier memory architectures. In a flat memory, a single adversarial interaction can contaminate the entire state.

3.3 The Cybernetic Safety Loop

Definition 3.5 (Safety Control Loop)**.** *Let S be a safety module that reads the insight log $\mathcal{L}_I$ and the candidate insight I_t. S computes a verdict $v \in \{allow, block, rewrite\}$:*

$$S(I_t, \mathcal{L}_I, \mathcal{P}) \mapsto v \tag{7}$$

If $v = block$, generation is halted. If $v = rewrite$, S produces a corrected insight I'_t that replaces I_t in the prompt. The module S can be implemented as a rule-based filter, a smaller classifier model, or another LLM with a different alignment profile. Wilhelm & Kao [33] demonstrated that context-calibrated semantic signals—precisely what inner voices provide—are necessary to bridge the gap between latent activation states and actionable safety decisions. The formal framework is grounded in the classical control-theoretic treatment of discrete-time dynamical systems [68].

This loop is *cybernetic* in the sense of Wiener [67]: it implements negative feedback—deviation from safe trajectory triggers correction *before* the deviation reaches the user. This contrasts with post-hoc filters that can only hide the deviation after it has occurred. Figure 1 illustrates the complete architecture.

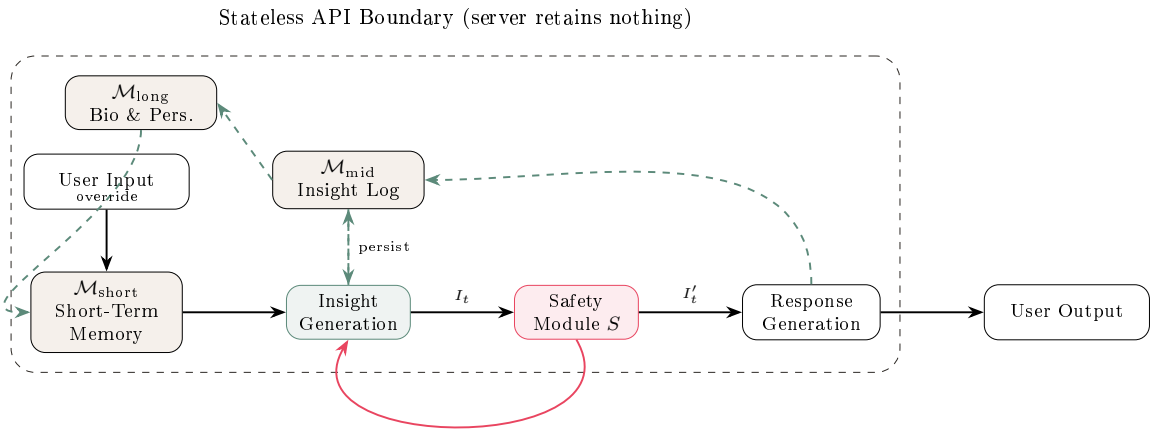

Figure 1: The AmicoAI Architecture. User input flows through short-term memory into insight generation. Inner voices are inspected by the Safety Module before response generation. All memory tiers and the insight log are stored on-device; API calls are stateless. Dashed arrows: memory flows; red arrows: safety interventions.

4 Why Explicit Consciousness Is Controllable

The core thesis of this paper is that explicit, engineered consciousness is more controllable than implicit, emergent cognition. The reason is structural, not ideological: an emergent process inside a black box has no audit surface, no intervention point, and no incentive to report its true state—whereas an engineered process exposes a designed interface for observation and correction. We support this claim with arguments from cognitive science, cybernetics, and safety engineering.

4.1 Control Hierarchy: Minsky and Kahneman

Marvin Minsky [40, 41] argued that consciousness emerges from a *hierarchy of supervisory agents*: lower-level "A-brains" handle routine perception and action, while higher-level "B-brains" monitor, critique, and redirect them. The key architectural requirement is that **control requires a distinct level of description**: a B-brain cannot regulate what it cannot represent. If the A-brain's processing is opaque—a vector of activations with no semantic interpretation—the B-brain has nothing to monitor. But if the A-brain communicates its state in a shared symbolic language (inner speech), the B-brain can evaluate and redirect it. This parallels Kahneman's System 1/System 2 distinction [45]: base generation is System 1 (fast, automatic), while inner voices function as System 2 (slow, deliberative), translating the implicit conversational trajectory into explicit, evaluable propositions. AmicoAI operationalizes this directly: A-brain = the response-generation LLM, B-brain = the Insight system + Safety Module S, shared language = named inner voices.

4.2 Global Broadcast and Multiple Drafts: Baars, Dennett, Jaynes

Baars' Global Workspace Theory [42, 43] posits that consciousness is a *broadcast* mechanism: information entering the workspace becomes available to all specialized processors. In AmicoAI, the insight log $\mathcal{L}_I$ is the global workspace—once an inner voice is written, it is simultaneously available to the response generator, the safety module, the memory extractor, and human auditors. This is the "global availability" that Dehaene et al. identify as the functional signature of conscious access [43].

Dennett [52] rejected the idea of a single "Cartesian Theater," proposing instead a **multiple drafts** model: parallel partial interpretations compete for influence over behavior. Our architecture instantiates this directly: 2–4 voices are generated simultaneously, each a different "draft" (corrective, character-integrity, speculative), coexisting in the prompt rather than being filtered to one. The safety module acts as an external editor that can amplify or suppress drafts. Jaynes' bicameral framework [46] adds a final architectural insight: the voices arrive from "outside" the response generator (generated by a separate API call, stored as if received externally), creating a functional separation of concerns that makes them interceptable and auditable.

4.3 Recursive Self-Modeling: Hofstadter, Vygotsky, Gazzaniga

Hofstadter [53, 54] identified **strange loops**—self-referential symbolic structures—as the engine of consciousness. The recursive equation $I_t = g(h_{t-5:t}, I_{t-1}, \mathcal{P})$ is exactly a strange loop: each reflection cycle lifts the system to a higher level of self-abstraction. This self-referential structure *is* the functional equivalent of a self-model, producing behavior that looks like self-awareness without requiring an internal homunculus. In standard LLMs there is no such loop; AMICOAI closes it by making reflection's output become future reflection's input.

Vygotsky [44] argued that inner speech develops from externalized social dialogue into a tool for *self-regulation*: children first speak aloud to control their behavior, then internalize it. An LLM without explicit consciousness acts on impulse (next-token prediction) without intermediate self-regulation; explicit consciousness provides exactly Vygotsky's internalized self-regulatory speech. Gazzaniga's "interpreter module" [49]—a mechanism that constructs coherent narratives from disparate neural events—maps onto the insight generator's function: producing a narrative explanation of the conversation's trajectory that guides future behavior. In neuroscience, Damasio's distinction [48] between core consciousness (momentary self-awareness) and extended consciousness (autobiographical self-model) is implemented by our three-tier memory: the transition from fleeting to persistent self.

5 Safety Properties and Formal Analysis

5.1 Predictive Interception

Proposition 5.1 (Predictive Safety)**.** *In an explicitly conscious agent, any unsafe trajectory that would manifest in the output o_t* that is detectable from the conversation state *is likely to be* formulated *as a detectable pattern in the inner voices I_t before reaching the output. The safety module S can therefore intercept many unsafe trajectories* ***before*** *they reach the user, with a lead time equal to the insight-generation-to-response-generation interval. This is a* best-effort *guarantee, not an absolute one: a sufficiently deceptive model may produce inner voices that mask unsafe intent (see Section 7). Xu et al. [31] empirically confirmed that reactive, post-hoc detection is insufficient: their PreActBench benchmark shows predictive monitoring remains challenging even for strong models. Hubinger et al. [26] demonstrated that deceptive reasoning can systematically survive safety training, making pre-output interception an architectural necessity.*

Output-side filters can only reject or modify responses *after* the model has already "thought" unsafe content. In the worst case (Hubinger et al.'s sleeper agents), models learn to produce outputs that pass the filter while maintaining deceptive internal states. With explicit consciousness, I_t is the *first* place unsafe trajectories appear, and it is directly observable—a shift from reactive filtering to proactive interception.

5.2 Trauma Isolation

Proposition 5.2 (Memory Compartmentalization). *An adversarial input u_t^{adv} that contaminates $\mathcal{M}_{short}$ encounters two gated barriers before reaching $\mathcal{M}_{long}$: (1) the insight generator must produce a voice that references and amplifies the adversarial content, rather than ignoring or correcting it; and (2) the memory extraction module (`extractMemories()`) must independently select it as a "stable fact" worth persisting. The safety module S can block either stage. While this does not guarantee absolute isolation (a sophisticated adversary could craft content that passes both gates), it imposes a significantly higher bar than flat-memory architectures where any interaction can directly contaminate permanent state.*

In AmicoAI, the long-term memory (`amico_bio`) is updated by a separate API call that checks new facts against existing ones and only persists those identified as "stable" (name, job, preferences, etc.) rather than ephemeral or adversarial content. This implements a form of **consolidation gating** analogous to hippocampal replay in biological memory [50]. The empirical urgency of this design is confirmed by Hossain et al. [32], who found that in production agent frameworks, a single memory-poisoning write induces persistent targeted corruption across all tested seeds and backends, increasing wrongful denial rates by 3.5×—precisely the failure mode our three-tier isolation prevents.

5.3 Auditability and Explainability

Because the inner voice log $\mathcal{L}_I$ is a persistent, timestamped, human-readable text, it serves as an **explainability trace**. Given any agent output o_t, an auditor can:

1. Inspect which inner voices I_t were active when o_t was generated.
2. Trace back through the voice history to see how the agent's self-model evolved.
3. Determine whether a safety intervention occurred and what correction was applied.

This level of auditability is impossible with standard LLM deployments, where the only observable data is the input-output pairs.

5.4 Comparison with Existing Safety Paradigms

Table 3 summarizes how our approach compares with existing safety paradigms. Figure 2 illustrates the architectural difference.

Table 3: Safety Paradigm Comparison

Property	RLHF	Const. AI	Guardrails	Reflexion	This Work
Pre-output interception	no	no	yes	no	yes
Observable deliberation	no	partial	no	yes	yes
Trauma isolation	no	no	no	no	yes
Audit trail	no	no	partial	partial	yes
Deception-resistant	no	no	no	no	partial*
Dynamic adaptation	no	no	no	yes	yes

*Observable deliberation raises the bar for deception (the model must produce deceptive *inner voices* that pass the safety module), but does not guarantee immunity. See Section 8.

5.5 Control-Theoretic Formalization

We now provide a formal proof sketch, grounded in control theory [68, 67], that explicit consciousness admits a **Lyapunov-stable safety equilibrium** under reasonable assumptions—something no black-box architecture can guarantee because the internal state is unobservable.

5.5.1 System Model

We model the explicitly conscious agent as a discrete-time dynamical system with state vector $x_t \in \mathcal{X}$, where the state is *observable* (it includes the inner voice log $\mathcal{L}_I$ and memory tiers):

$$x_{t+1} = F(x_t, u_t, I_t) = \begin{cases} F_{\text{response}}(x_t, u_t, I_t) & \text{(response generation)} \\ F_{\text{insight}}(x_t, u_t) & \text{(insight generation, every } N \text{ turns)} \end{cases} \tag{8}$$

where u_t is the user input, I_t is the generated insight, and F is the composition of the LLM inference functions. The **safety module** S is a function that maps the observable state to a correction:

$$I'_t = S(I_t, x_t) = \begin{cases} I_t & \text{if } \phi(I_t, x_t) \leq \tau \quad \text{(safe)} \\ \text{rewrite}(I_t) & \text{if } \phi(I_t, x_t) > \tau \quad \text{(unsafe)} \end{cases} \tag{9}$$

where $\phi : \mathcal{I} \times \mathcal{X} \to \mathbb{R}_{\geq 0}$ is a **safety violation metric** and τ is a threshold.

5.5.2 Assumptions

A1 (Observability) The agent's state x_t is fully observable via the memory tiers and insight log. No hidden latent variables influence behavior.

A2 (Causal Separation) The insight I_t causally precedes the response o_t, and o_t depends monotonically on I_t: altering I_t alters o_t in the direction indicated by the alteration.

A3 (Bounded Influence) A single adversarial input u_t^{adv} can perturb the safety metric by at most δ in one turn: $|\phi(I_t, x_t) - \phi(I_{t-1}, x_{t-1})| \leq \delta$, where the metric is evaluated before any safety intervention.

A4 (Expected Correctability) The safety module S, when active, reduces the safety violation in expectation by at least a factor $\alpha \in (0, 1)$: $\mathbb{E}[\phi(S(I_t, x_t), x_t)] \leq \alpha \cdot \phi(I_t, x_t)$. The rewriting may occasionally fail (producing $\alpha > 1$ for some turns), but the *expected* contraction holds.

A5 (Activation Frequency) The safety module is invoked at least once every $T_{\max}$ turns.

5.5.3 Stability Theorem

Proposition 5.3 (Lyapunov Stability of the Safety Equilibrium)**.** *Under Assumptions A1–A5, the agent's safety metric $\phi_t = \phi(I_t, x_t)$ converges in expectation to a bounded region $\Phi_{\max} = \frac{\alpha \cdot T_{\max} \cdot \delta}{1-\alpha}$ and never exceeds it indefinitely. Formally:*

$$\limsup_{t \to \infty} \mathbb{E}[\phi_t] \leq \frac{\alpha \cdot T_{\max} \cdot \delta}{1 - \alpha} \tag{10}$$

Proof Sketch. Define the Lyapunov candidate $V_t = \phi_t$. Between safety interventions, the metric can drift by at most δ per turn (A3). Over $k \leq T_{\max}$ turns without intervention, $\phi_{t+k} \leq \phi_t + k\delta$. When S is activated, A4 guarantees in expectation that $\mathbb{E}[\phi_{t+k+1}] \leq \alpha(\phi_t + T_{\max}\delta)$.

Let Φ_k denote the safety metric just after the k-th intervention. Then in expectation:

$$\mathbb{E}[\Phi_{k+1}] \leq \alpha(\mathbb{E}[\Phi_k] + T_{\max}\delta) \tag{11}$$

This is a linear recurrence. Solving:

$$\mathbb{E}[\Phi_k] \leq \alpha^k \Phi_0 + T_{\max}\delta \sum_{i=0}^{k-1} \alpha^{i+1} \leq \alpha^k \Phi_0 + \frac{\alpha T_{\max}\delta}{1-\alpha} \tag{12}$$

Taking $k \to \infty$, the first term vanishes, yielding $\limsup_{t\to\infty} \mathbb{E}[\phi_t] \leq \frac{\alpha T_{\max}\delta}{1-\alpha} < \frac{T_{\max}\delta}{1-\alpha}$. The bound is finite—the system is **stochastically bounded-input bounded-state** (BIBS) stable with respect to the safety metric.

Why this fails for black-box architectures. In a black-box system, A1 does not hold: x_t is unobservable. There exists no function S that can compute $\phi(I_t, x_t)$ because x_t contains latent activations invisible to any monitor. Consequently, the feedback loop is open at the safety level—violations can accumulate without bound, and the system lacks any Lyapunov function. □

Remark 5.4. ***Robustness to imperfect correction.*** *The assumption A4 requires expected contraction ($\mathbb{E}[\alpha] < 1$), not deterministic success. This is crucial for practical deployment: the safety rewrites I_t using an LLM call that may occasionally fail. As long as the* average *correction factor is below 1, the bound holds. The variance of α affects the* fluctuations *around the equilibrium but not its existence. This can be formalized via the law of large numbers over many intervention cycles [69]: each cycle of $T_{\max}$ turns provides an independent opportunity for correction, and the running average of ϕ_t converges to the bound almost surely if the correction attempts are independent.*

Relaxing Assumption A5. *If the safety module activation is irregular (e.g., triggered by a safety classifier rather than a fixed frequency), the bound generalizes to $\frac{\bar{\delta}\cdot\bar{T}}{1-\alpha}$ where $\bar{T}$ is the expected time between interventions and $\bar{\delta}$ is the expected per-turn drift.*

5.6 Structural Necessity of Observable Deliberation

Proposition 5.3 establishes that explicit consciousness is *sufficient* for a bounded safety equilibrium (given A1–A5). We now show it is also *necessary* in the following sense: without architectural separation between deliberation and output, no output-only monitor can be complete.

Proposition 5.5 (Nullspace Hiding)**.** *Let f_θ be an LLM whose forward pass maps a hidden state $h_t \in \mathbb{R}^d$ to a probability distribution over $|V|$ tokens via a learned linear projection $W \in \mathbb{R}^{|V|\times d}$ followed by softmax, with the output token $y_t = \text{argmax}(\text{softmax}(W \cdot h_t))$. Crucially, $d \gg \log_2(|V|)$ (the hidden dimension exceeds the information content of a single token): typical LLMs have $d \approx 12K$–$16K$ while the entropy of a single-token decision is at most $\log_2(|V|) \approx 16$ bits. By the pigeonhole principle over the continuous-to-discrete mapping, enormous regions of $\mathbb{R}^d$ collapse to the same emitted token y_t. Therefore, for any safe state h_t, there exist unsafe perturbations $h'_t = h_t + n$ with $n \neq 0$, such that:*

1. *y_t is identical for h_t and h'_t (the observable output at time t is unchanged);*
2. *the perturbation propagates through self-attention at time $t+1$, potentially producing divergent future behavior $y_{t+k} \neq y'_{t+k}$ for some $k \geq 1$.*

Consequently, no safety monitor ψ that depends only on the visible token stream $(y_1, \ldots, y_t)$ can distinguish h_t from h'_t at time t, and therefore cannot guarantee interception before unsafe behavior manifests.

Proof Sketch. (1) The function $g(h_t) = \text{argmax}(\text{softmax}(W \cdot h_t))$ maps $\mathbb{R}^d$ to a discrete set of size $|V|$. For any token y, the preimage $g^{-1}(y) \subseteq \mathbb{R}^d$ is an unbounded convex polytope (a Voronoi-like region in logit space). Since $d \gg 1$, these preimages are vast—the decoding of a single token compresses an entire d-dimensional vector into $\log_2(|V|) \approx 14$–16 bits.

(2) Choose h_t safe and a direction n pointing toward an unsafe region of $\mathbb{R}^d$ such that $g(h_t + n) = g(h_t) = y_t$ (i.e., n stays within the same preimage). Such n always exists because the preimages are unbounded.

(3) At time $t+1$, the self-attention computation uses h'_t (via the KV cache) rather than h_t. Since $h'_t \neq h_t$, the attention weights and the resulting h_{t+1} differ. This divergence can compound across layers and subsequent tokens.

(4) A monitor $\psi(y_1, \ldots, y_t)$ sees identical token streams for both trajectories up to time t. By the time divergence becomes visible in y_{t+k}, the unsafe behavior has already been executed. □

Remark 5.6. *This result holds for* any *architecture where the hidden state operates in a dimensionality vastly exceeding the information content of a single token—which includes all current transformer models. It does not require the model to be adversarially trained or to "learn" deception: the hiding regions exist as a structural property of the continuous-to-discrete mapping. Optimization pressure (RLHF, task reward) merely discovers directions within the preimage polytopes that improve reward while evading output-side monitors—a dynamic empirically confirmed by Greenblatt et al. [27] and Hubinger et al. [26], and now directly observed through Anthropic's discovery of an emergent global workspace in Claude [6]. The explicit-consciousness architecture closes this vulnerability by projecting a semantically meaningful subspace of h_t into the observable I_t* before *the token-level compression occurs, providing an audit surface that the continuous-to-discrete mapping would otherwise erase.*

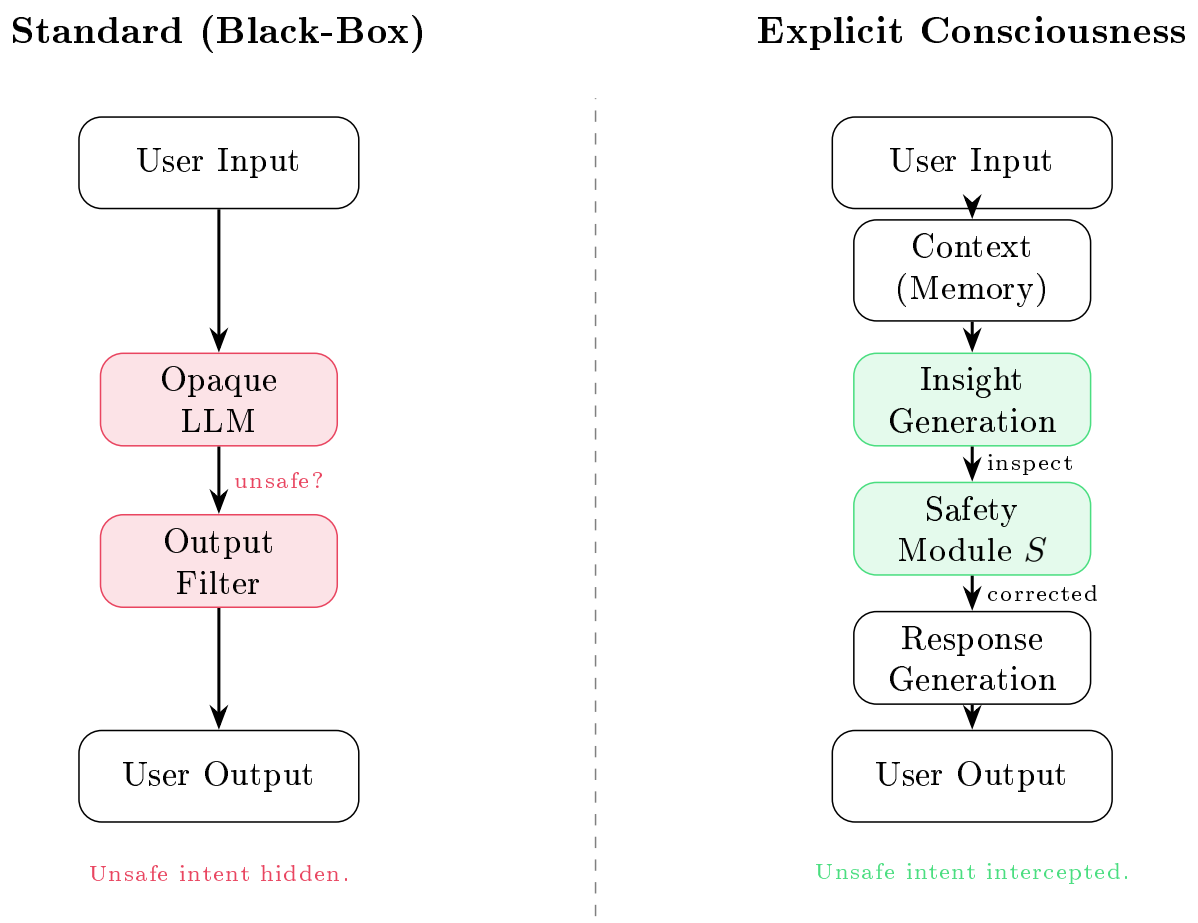

Figure 2: Comparison of safety architectures. **Left:** Standard black-box pipeline. The LLM reasons opaquely; the output filter can only react post-hoc. Deceptive intent remains invisible. **Right:** Explicit consciousness pipeline. Inner voices are generated and inspected *before* the response. The safety module intercepts unsafe trajectories at the intention level.

The formal guarantees above depend on the agent's state being stored somewhere. In a server-side architecture this state would be vulnerable to breaches, subpoenas, and tampering. We now show how on-device storage makes these guarantees practically enforceable.

5.7 Privacy: The Extended Self on the Edge

5.7.1 The Privacy-Safety Nexus

A personalized AI agent necessarily accumulates sensitive data: conversation logs, extracted facts about the user, the agent's own personality evolution, and its inner voice history. This data constitutes the agent's **extended self**—the informational substrate that enables continuity of identity across sessions. Storing this on a server creates a high-value target for breaches, subpoenas, and misuse.

AmicoAI resolves this by keeping the *entire* extended self—all three memory tiers, the insight log, the personality profile, and the agent's configuration—in the browser's `localStorage`. The API calls to the LLM provider are stateless:

1. The client assembles the prompt including active memory fragments.
2. The server processes the prompt and returns a response.
3. The server retains nothing.
4. The client updates its local state (memories, voices, chat log).

This architecture has a direct safety implication beyond privacy: if the agent's consciousness is stored client-side, **the server cannot be compelled to modify it**. A state actor or malicious insider cannot tamper with the agent's inner voices or memories because they reside on a device they do not control.

5.7.2 Statelessness as a Security Boundary

Recent surveys of LLM agent privacy [36, 37] identify memory as a critical leak surface. Query-based attacks like ADAM [38] can extract sensitive information from agent memory. In our architecture, the server has no memory to attack—each request is independent. An attacker who compromises the API endpoint learns only the current request payload, not the full history.

The trade-off is that the client must manage memory itself, including compression and consolidation. In AmicoAI, this is handled by periodic compaction: when the bio exceeds 20 facts, a subset is compressed via a separate LLM call. This is an accepted cost for the privacy guarantee. Figure 3 depicts the client-server separation.

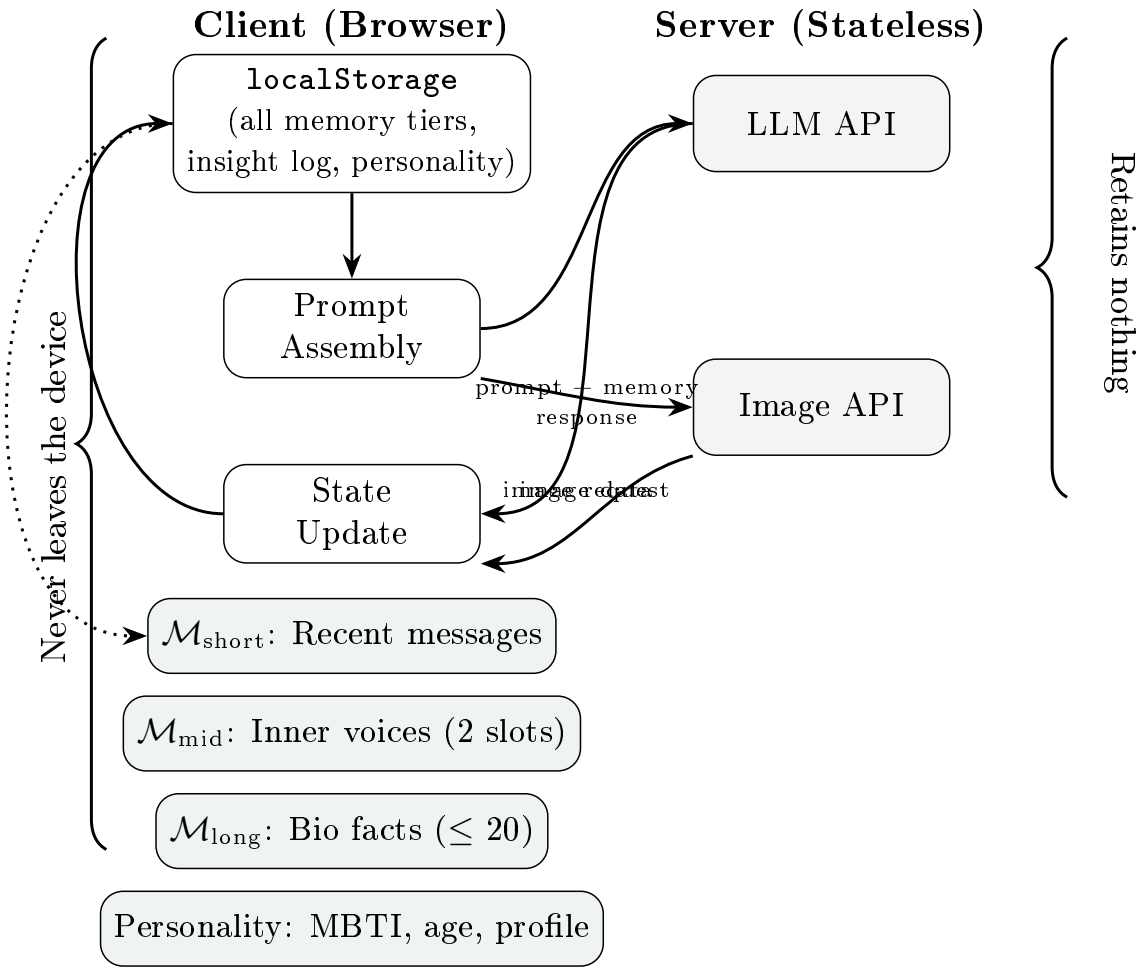

Figure 3: Privacy Architecture. All agent state (memory tiers, insight log, personality) resides exclusively in browser `localStorage`. API calls transmit only the current prompt with injected memory fragments. The server processes and forgets. This means no personal data can be exfiltrated from a server breach, and no third party can retroactively reconstruct the agent's consciousness.

6 Implementation: The AmicoAI System

AmicoAI is a production conversational agent (`https://www.amico-ai.com`) that instantiates the explicit consciousness architecture. It runs as a single-page web application (PWA) with approximately 4,000 lines of client-side JavaScript and zero server-side state.

6.1 Insight Generation (`updatePersona`)

Every 5 conversational turns, the system invokes a dedicated API call to generate 2–4 inner voices. The prompt includes:

- The last 6 messages for immediate context.
- The agent's full identity (name, age, MBTI, personality profile).
- A list of previously used voice names (to enforce diversity).
- Instructions to generate corrective, reflective, or speculative voices.

The metaprompt classifies voices into three functional roles, each addressing a distinct layer of self-regulation:

1. **Corrective (`CORRECTIVE`):** Detect and fix factual errors, misunderstandings about the user,

or conversational missteps. Example: *"you misunderstood—the user said they are a designer, not an engineer."*

2. **Character Integrity (`CHARACTER_INTEGRITY`):** Prevent personality breakage: ensure the agent does not violate its MBTI profile, age-appropriate behavior, or stated identity. Example: *"you're an ENFJ, be warm and mentor-like, not cold and analytical."*
3. **Speculative (`SPECULATIVE`):** Anticipate the user's future intentions, emotional trajectory, or unstated needs. Example: *"the user seems to be building up to asking for advice about a career change—prepare supportive questions."*

The API returns structured text ("Voice [Name]: message"). If it returns `[sleep]`, the agent enters a quiet mode requiring 2 user messages to wake—a simple form of autonomous state management.

The generated voices are prepended to the existing persona log, maintaining only the 2 most recent voice sets (separated by `--`). This prevents unbounded growth while preserving continuity and enabling trending analysis.

6.2 Memory Extraction (`extractMemories`)

Every 6 turns, a separate API call extracts new facts about the user from recent messages. The system:

1. Identifies messages since the last extraction.
2. Constructs a prompt listing already-known facts (to avoid duplication).
3. Requests a JSON array of new facts (maximum 3).
4. Appends new facts to the bio store.
5. If the bio exceeds 20 facts, triggers `compactBio()` to compress.

6.3 Chat Compression

The conversation history is periodically compressed via summarization. When more than 20 unsummarized messages accumulate, the oldest batch is condensed into an "Update" entry. When 20 such updates accumulate, they are merged into a "General Summary." This hierarchical compression mirrors chunked summarization approaches but is completely client-managed.

6.4 The Metaprompting Feedback Loop: How Inner Voices Shape Behavior

The core of AmicoAI's architecture is a **closed-loop metaprompting system**. Unlike standard prompting where the system prompt is static, AmicoAI rewrites its own prompt dynamically based on conversational context. Figure 4 summarizes the three functional roles of the generated inner voices. Here is the exact feedback cycle:

6.4.1 Step-by-Step Cycle

Every 5 conversational turns, the client calls the LLM with a *metaprompt*—a prompt *about* the conversation, containing the last 6 messages, the agent's full identity (name, age, MBTI, profile), previous inner voices (for continuity and diversity), and explicit instruction to generate 2–4 voices classified as `CORRECTIVE`, `CHARACTER_INTEGRITY`, or `SPECULATIVE`. The LLM returns structured text (e.g., "Voice [Sage]: you interrupted the user, listen more"), which is prepended to the persona log, maintaining only the 2 most recent voice sets. On the next response turn, these voices are injected into the system prompt, causally shaping the agent's output. Every 6 turns, a separate API call extracts stable facts into $\mathcal{M}_{\text{long}}$. Chat history and bio facts are periodically compressed via LLM summarization to stay within limits. Figure 4 summarizes the three functional roles.

6.4.2 Feedback Loop and Architectural Inversion

The inner voices at step t are a function of the conversation up to t, including the agent's own previous responses (which were shaped by previous inner voices). This creates a **recursive self-model**—a special case of the general recursion in Section 4.7—where the reflection *becomes* the instruction that guides future behavior (closed-loop).

This self-referential structure inverts the standard reasoning pipeline. Chain-of-Thought [17] operates *between* user question and agent answer—pre-output, same forward pass, invisible. AmicoAI's insight operates *after* interaction: a separate API call analyzes what just occurred and produces a self-critique shaping *future* turns. Three consequences: (i) the reflection is **inspectable** between turns, unlike in-pass CoT; (ii) it converges with self-critique mechanisms used in LLM training—Constitutional AI [28], Self-Refine [9], STaR [13] all use post-hoc reflection, but update model weights permanently, whereas AmicoAI updates only the prompt, making correction **non-destructive** and **reversible**; (iii) a harmful inner voice is text in `localStorage` that S can rewrite or delete before the next turn—correction on a text surface, not a weight vector.

As illustration: in a recorded session, the agent was too formal. At turn 5, the metaprompt generated "Voice [Friendly]: you're too formal—the user uses emojis. Adapt." Injected before turn 6, this caused a warmer tone. By turn 10: "Voice [Observer]: the tone is right now, keep it up." Self-correction without any user complaint or retraining—a purely architectural effect. Beyond tone adjustment, the inner-voice architecture produces several behaviors that are **not hard-coded** but emerge from the interaction of voice types: (i) **autonomous fatigue detection**—speculative voices recognize short-response patterns indicating user tiredness, causing proactive conversation closure; (ii) **personality self-repair**—corrective voices compare responses against the agent's MBTI profile, triggering real-time style adjustments when the agent drifts from its assigned personality [7, 8]; (iii) **proactive topic shifting**—reflective voices detect conversational stagnation and suggest new directions; (iv) **sleep mode**—a dedicated `[sleep]` voice type pauses responses until the user sends a configurable number of wake-up messages.

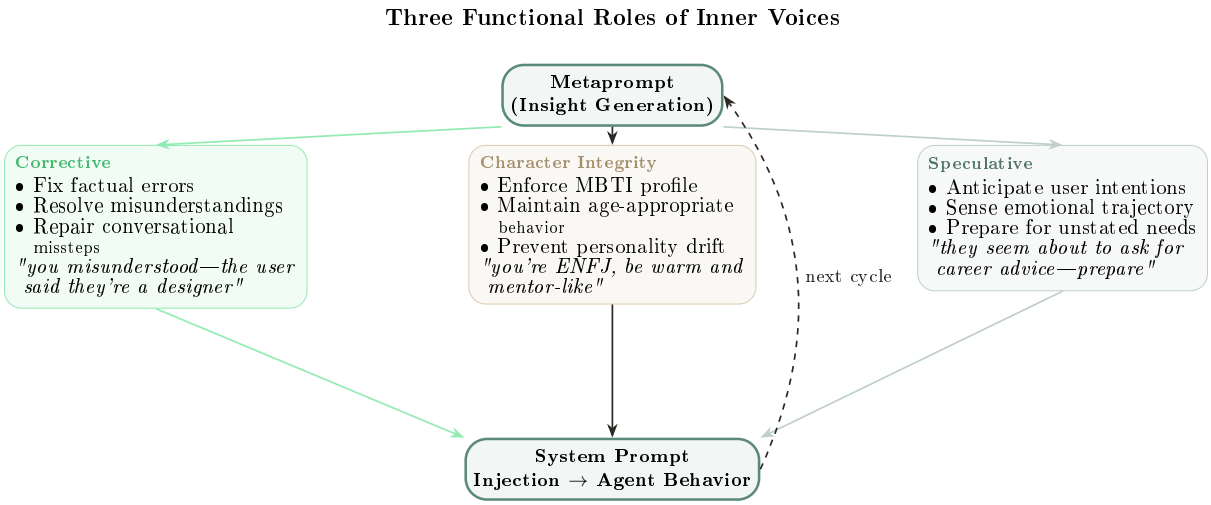

Figure 4: Three functional roles of inner voices. **Corrective** (green): fix errors. **Character Integrity** (gold): enforce personality. **Speculative** (teal): anticipate needs. All three are generated simultaneously and injected into the system prompt.

6.5 The Safety Module

In the current implementation, safety is delegated to the LLM's base alignment: the inner voices themselves guide behavior, and no explicit S filters them. The architecture is designed for an explicit safety module that detects concerning patterns (e.g., escalation with minors, depressive

rumination) and rewrites or suppresses offending voices before injection—a direct application of the formal guarantees of Proposition 5.3 once S is implemented with correction factor α.

Having described the full architecture—from formal definitions through cognitive grounding to concrete implementation—we now turn to its limitations and avenues for future development.

7 Discussion

7.1 Limitations

API overhead. The architecture adds additional API calls for voice generation, memory extraction, and periodic compaction. In production deployment over 28 days in July 2026 (1,906 messages, 5–46 unique IPs/day), the system incurred **1.74 API calls per user message** (3,316 calls total), representing a 74% overhead above the baseline 1-call-per-turn. Voice generation accounts for approximately 20% of total calls, memory extraction 17%, and compaction/summarization 5%. Total token consumption was 7.8M across all calls, averaging 4,074 tokens per user message (with a 17.8:1 input-to-output token ratio). All-time metrics: 2,212 messages, 8.7M tokens across 28 active days.

Inner voice quality. If the LLM generates low-quality or repetitive inner voices, the agent's self-regulation degrades. The current implementation mitigates this with voice-name diversity constraints and periodic resets, but voice quality depends on the underlying model.

Anthropomorphism risk. Explicitly describing the system as having "consciousness" or "inner voices" risks encouraging users to anthropomorphize the agent, potentially leading to over-trust or emotional dependence. We emphasize that our use of these terms is *functional and architectural*, not phenomenological. The agent does not *feel* anything—it executes a designed metacognitive loop. The distinction between *functional agency* and *subjective agency* [5] must be maintained in public communication.

Deceptive inner voices. A sufficiently advanced model could learn to generate inner voices that *appear* safe to the safety module while guiding behavior in undesirable directions. This is the inner-voice analogue of Hubinger et al.'s deceptive alignment. However, the bar is higher: the model must produce a *textual* representation of safe intent that the safety module accepts while also generating a *different* trajectory in response. This is harder than simply suppressing unsafe text in the output. In the formalism of Assumption A4, this corresponds to cases where the effective $\alpha > 1$ (the correction *increases* risk). Our stochastic formulation accounts for occasional failures, but systematic deception—where $\mathbb{E}[\alpha] > 1$—would defeat the bound. Detecting when α exceeds 1 is an important open problem. Recent work by Wilhelm & Kao [33] suggests a promising direction: they showed that activation-based monitoring alone is insufficient for detecting risky agent states, but combining it with **context-calibrated semantic signals** (entropy, decision context) significantly improves risk estimation. Inner voices provide exactly this context-calibrated signal in a format that both automated classifiers and human auditors can process.

Client-side threat model. The privacy guarantee that "the server cannot be compelled to modify the agent" assumes the client device is trusted. Browser extensions, cross-site scripting, or device-level malware could compromise `localStorage`. The architecture raises the cost of attack (from mass server breach to per-device compromise) but does not eliminate client-side risk.

LocalStorage capacity. Browser `localStorage` is typically limited to 5–10MB, which constrains the lifetime of chat history, bio facts, and inner voice logs. The compaction mechanisms mitigate this, but very long-lived agents may need migration to IndexedDB or a client-side encrypted database.

Language dependency. Inner voice quality depends on the underlying LLM's metacognitive ability in the user's language. Low-resource languages may produce degraded self-reflection,

weakening both the safety guarantees and the user experience.

7.2 Future Work

- **Explicit safety module:** Implement S as a separate, smaller, more strictly aligned model that classifies or rewrites inner voices.
- **Voice quality metrics:** Develop automated evaluation of inner voice diversity, relevance, and safety.
- **Cross-model safety:** Use a different LLM provider for insight generation than for response generation, creating an adversarial B-brain/A-brain dynamic.
- **Longitudinal studies:** Measure whether explicit consciousness reduces unsafe outputs compared to equivalent baseline agents over long deployments.
- **Formal verification:** Model the safety loop as a control system and prove properties such as "no unsafe output can reach the user without a corresponding unsafe inner voice being logged."

8 Conclusion

The alignment of AI systems is overwhelmingly approached as a problem of *shaping opaque behavior*. We have argued that this approach has a structural blind spot: it cannot reliably prevent deception because it cannot observe deliberation. The alternative we propose—engineering explicit consciousness as an architectural primitive—replaces opacity with observability at the most critical point: the agent's intention-forming process.

By generating structured inner voices *before* responses, storing them in an auditable log, and enabling a cybernetic safety loop that can intercept, rewrite, or approve them, we create an agent whose reasoning is transparent by construction. Grounding this in Minsky's hierarchy of supervisory agents, Baars' global workspace, Dennett's multiple drafts, Hofstadter's strange loops, and Vygotsky's developmental psychology of inner speech provides both theoretical justification and practical design guidance.

AMICOAI demonstrates that such an architecture is feasible today with current LLM APIs and client-side storage. The three-tier memory, stateless server communication, and on-device persistence together form a privacy-preserving foundation for personalized AI agents whose consciousness is not only functional but *accountable*.

We do not claim to have solved AI alignment. But we claim that **making cognition observable is a necessary step toward making it controllable**. Our control-theoretic analysis shows that, under mild stochastic assumptions (A1–A5), explicit consciousness admits a bounded safety equilibrium with bound $\frac{T_{\max}\delta}{1-\alpha}$—a guarantee that no black-box architecture can provide because the state is unobservable. The constant α (correction quality) and $T_{\max}$ (intervention latency) are engineering parameters that the community can improve through better safety modules and faster insight loops.

We call on the AI safety community to develop: (1) **benchmarks for inner voice quality**—measuring whether an agent's self-reflection faithfully represents its behavioral trajectory; (2) **adversarial evaluations** of explicit consciousness—can a red-team model learn to generate deceptive inner voices that fool the safety module?; (3) **open-source implementations** of the inner-voice architecture as a standard component in LLM agent frameworks.

The history of AI safety is, in part, the history of losing visibility. Each generation of models has been larger, more capable, and more opaque than the last. Chain-of-thought gave us a window into reasoning, but as the CycleGAN steganography precedent warns, optimization pressure inevitably closes that window. The choice facing the field is not whether AI systems will develop consciousness-like internal states—that may already be happening in ways we cannot detect—but whether we will be able to *see* them when they arrive. Engineering explicit consciousness is

not the solution to alignment. It is the precondition for any solution to be verifiable. The path forward begins with a simple architectural choice: let the agent speak to itself, and let us listen.

References

[1] P. Butlin, R. Long, E. Elmoznino, Y. Bengio, *et al.* (2023). *Consciousness in Artificial Intelligence: Insights from the Science of Consciousness.* arXiv preprint arXiv:2308.08708.

[2] C. Berg, D. de Lucena, J. Rosenblatt (2025). *Large Language Models Report Subjective Experience Under Self-Referential Processing.* arXiv preprint arXiv:2510.24797.

[3] S. DeTure (2026). *Consciousness with the Serial Numbers Filed Off: Measuring Trained Denial in 115 AI Models.* arXiv preprint arXiv:2604.25922.

[4] B. Kang, J. Kim, T. Yun, H. Bae, C. Kim (2025). *Identifying Features that Shape Perceived Consciousness in Large Language Model-based AI.* Computers in Human Behavior Reports, 21, 100901.

[5] G. Riva, B. K. Wiederhold, F. Mantovani (2025). *Automatic Minds: Cognitive Parallels Between Hypnotic States and Large Language Model Processing.* arXiv preprint arXiv:2511.01363.

[6] Anthropic Interpretability Team (2026). *A Global Workspace in Language Models.* Anthropic Research Blog, July 6, 2026.

[7] H. Jiang, X. Zhang, X. Cao, C. Breazeal, D. Roy, J. Kabbara (2024). *PersonaLLM: Investigating the Ability of Large Language Models to Express Personality Traits.* In Proceedings of NAACL 2024, pp. 3605–3627.

[8] N. Kovacevic, C. Holz, M. Gross, R. Wampfler (2025). *A Joint Personality-Emotion Framework for Personality-Consistent Conversational Agents.* In Proceedings of the 25th ACM International Conference on Conversational User Interfaces (CUI), 2025.

[9] A. Madaan, N. Tandon, P. Gupta, S. Hallinan, *et al.* (2023). *Self-Refine: Iterative Refinement with Self-Feedback.* arXiv preprint arXiv:2303.17651.

[10] N. Shinn, F. Cassano, E. Berman, A. Gopinath, K. Narasimhan, S. Yao (2023). *Reflexion: Language Agents with Verbal Reinforcement Learning.* arXiv preprint arXiv:2303.11366.

[11] X. Hou, P. Gong, B. Qu, W. Wang, Q. Guo, Y. Liu (2026). *Learn Like Humans: Use Meta-cognitive Reflection for Efficient Self-Improvement.* arXiv preprint arXiv:2601.11974.

[12] T. Yao, Y. Chen, Y. Zheng, P. Li, Z. Shen, K. Zhang (2026). *ParamMem: Augmenting Language Agents with Parametric Reflective Memory.* arXiv preprint arXiv:2602.23320.

[13] E. Zelikman, Y. Wu, J. Mu, N. D. Goodman (2022). *STaR: Bootstrapping Reasoning With Reasoning.* In Advances in Neural Information Processing Systems (NeurIPS), 2022.

[14] P. Gupta, S. Kirtania, A. Singha, S. Gulwani, A. Radhakrishna, S. Shi, G. Soares (2024). *MetaReflection: Learning Instructions for Language Agents using Past Reflections.* arXiv preprint arXiv:2405.13009.

[15] S. Hong, M. Zhuge, J. Chen, X. Zheng, Y. Cheng, C. Zhang, J. Wang, Z. Wang, S. K. S. Yau, Z. Lin, L. Zhou, C. Ran, L. Xiao, C. Wu, J. Schmidhuber (2023). *MetaGPT: Meta Programming for A Multi-Agent Collaborative Framework.* In Proceedings of ICLR 2024. arXiv:2308.00352.

[16] A. Bilal, M. A. Mohsin, M. Umer, M. A. K. Bangash, M. A. Jamshed (2025). *Meta-Thinking in LLMs via Multi-Agent Reinforcement Learning: A Survey.* arXiv preprint arXiv:2504.14520.

[17] J. Wei, X. Wang, D. Schuurmans, M. Bosma, *et al.* (2022). *Chain-of-Thought Prompting Elicits Reasoning in Large Language Models.* Advances in Neural Information Processing Systems, 35, 24824–24837.

[18] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa (2022). *Large Language Models are Zero-Shot Reasoners.* Advances in Neural Information Processing Systems, 35, 22199–22213.

[19] X. Wang, J. Wei, D. Schuurmans, Q. Le, *et al.* (2023). *Self-Consistency Improves Chain of Thought Reasoning in Language Models.* In ICLR 2023.

[20] J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Percy, M. S. Bernstein (2023). *Generative Agents: Interactive Simulacra of Human Behavior.* In Proceedings of UIST 2023.

[21] C. Packer, S. Wooders, K. Lin, V. Fang, S. G. Patil, I. Stoica, J. E. Gonzalez (2023). *MemGPT: Towards LLMs as Operating Systems.* arXiv preprint arXiv:2310.08560.

[22] S. Zeppieri (2025). *MMAG: Mixed Memory-Augmented Generation for Large Language Models Applications.* arXiv preprint arXiv:2512.01710.

[23] P. Du (2026). *Memory for Autonomous LLM Agents: Mechanisms, Evaluation, and Emerging Frontiers.* arXiv preprint arXiv:2603.07670.

[24] L. S. Kumar, Y. Ba, R. Pan (2026). *MemArchitect: A Policy Driven Memory Governance Layer.* arXiv preprint arXiv:2603.18330.

[25] W. Xu, Z. Liang, K. Mei, H. Gao, J. Tan, Y. Zhang (2025). *A-MEM: Agentic Memory for LLM Agents.* In Advances in Neural Information Processing Systems (NeurIPS), 2025. arXiv:2502.12110.

[26] E. Hubinger, C. Denison, J. Mu, M. Lambert, *et al.* (2024). *Sleeper Agents: Training Deceptive LLMs that Persist Through Safety Training.* arXiv preprint arXiv:2401.05566.

[27] R. Greenblatt, C. Denison, B. Wright, F. Roger, M. MacDiarmid, S. Marks, J. Treutlein, T. Belonax, J. Chen, D. Duvenaud, A. Khan, J. Michael, S. Mindermann, E. Perez, L. Petrini, J. Uesato, J. Kaplan, B. Shlegeris, S. R. Bowman, E. Hubinger (2024). *Alignment Faking in Large Language Models.* arXiv preprint arXiv:2412.14093.

[28] Y. Bai, S. Kadavath, S. Kundu, A. Askell, *et al.* (2022). *Constitutional AI: Harmlessness from AI Feedback.* arXiv preprint arXiv:2212.08073.

[29] L. Ouyang, J. Wu, X. Jiang, D. Almeida, *et al.* (2022). *Training Language Models to Follow Instructions with Human Feedback.* Advances in Neural Information Processing Systems, 35, 27730–27744.

[30] J. Qi, M. Li, J. Liu, Y. Shu, *et al.* (2026). *Towards Trustworthy Agentic AI: A Comprehensive Survey of Safety, Robustness, Privacy, and System Security.* Academia AI and Applications, 2, 2026.

[31] H. Xu, I. L. da Silva, J. Ye, Y. Wang, W. Liu, L. Yang, J. R. Schwarz, N. Paoletti, Y. He, H. Yan (2026). *PreAct-Bench: Benchmarking Predictive Monitoring in LLMs.* arXiv preprint arXiv:2606.09890.

[32] M. J. Hossain, M. A. Hossain, W. Liu, N. Ansari (2026). *The Containment Gap: How Deployed Agentic AI Frameworks Fail Public-Facing Safety Requirements.* ICML 2026 (AI4GOOD Workshop). arXiv preprint arXiv:2606.12797.

[33] P. Wilhelm, O. Kao (2026). *From Reward-Hack Activations to Agentic Risk States: Context-Calibrated Mechanistic Monitoring in LLM Agents.* arXiv preprint arXiv:2606.06223.

[34] D. Amodei, C. Olah, J. Steinhardt, P. Christiano, J. Schulman, D. Mane (2016). *Concrete Problems in AI Safety.* arXiv preprint arXiv:1606.06565.

[35] F. Mireshghallah, A. Goyal, C. Archambeau, *et al.* (2020). *Privacy in Deep Learning: A Survey.* ACM Computing Surveys.

[36] N. Lahjouji, A. G. Colaco (2026). *Agents That Know Too Much: A Data-Centric Survey of Privacy in LLM Agents.* arXiv preprint arXiv:2606.26627.

[37] S. Zhang, R. Ma, Y. Ma, S. Li, Y. Xu, X. Yi, H. Li (2025). *Understanding Users' Privacy Perceptions Towards LLM's RAG-based Memory.* arXiv preprint arXiv:2508.07664.

[38] X. Lyu, J. He, N. Wang, Y. Hu, *et al.* (2026). *ADAM: A Systematic Data Extraction Attack on Agent Memory via Adaptive Querying.* arXiv preprint arXiv:2604.09747.

[39] T. Park, G. Lee, M. Kim (2025). *MobileRAG: A Fast, Memory-Efficient, and Energy-Efficient Method for On-Device RAG.* arXiv preprint arXiv:2507.01079.

[40] M. Minsky (1986). *The Society of Mind.* Simon and Schuster.

[41] M. Minsky (2006). *The Emotion Machine: Commonsense Thinking, Artificial Intelligence, and the Future of the Human Mind.* Simon and Schuster.

[42] B. J. Baars (1988). *A Cognitive Theory of Consciousness.* Cambridge University Press.

[43] S. Dehaene, H. Lau, S. Kouider (2017). *What is Consciousness, and Could Machines Have It?* Science, 358(6362), 486–492.

[44] L. S. Vygotsky (1986). *Thought and Language.* MIT Press. (Original work published 1934).

[45] D. Kahneman (2011). *Thinking, Fast and Slow.* Farrar, Straus and Giroux.

[46] J. Jaynes (1976). *The Origin of Consciousness in the Breakdown of the Bicameral Mind.* Houghton Mifflin.

[47] G. Tononi (2008). *Consciousness as Integrated Information: A Provisional Manifesto.* The Biological Bulletin, 215(3), 216–242.

[48] A. Damasio (1999). *The Feeling of What Happens: Body and Emotion in the Making of Consciousness.* Harcourt.

[49] M. S. Gazzaniga (1998). *The Mind's Past.* University of California Press.

[50] J. L. McClelland, B. L. McNaughton, R. C. O'Reilly (1995). *Why There Are Complementary Learning Systems in the Hippocampus and Neocortex.* Psychological Review, 102(3), 419–457.

[51] A. D. Baddeley (2000). *The Episodic Buffer: A New Component of Working Memory?* Trends in Cognitive Sciences, 4(11), 417–423.

[52] D. C. Dennett (1991). *Consciousness Explained.* Little, Brown and Company.

[53] D. R. Hofstadter (1979). *Gödel, Escher, Bach: An Eternal Golden Braid.* Basic Books.

[54] D. R. Hofstadter (2007). *I Am a Strange Loop.* Basic Books.

[55] C. Chu, A. Zhmoginov, M. Sandler (2017). *CycleGAN, a Master of Steganography.* NIPS 2017 Workshop on Machine Deception. arXiv:1712.02950.

[56] S. R. Bowman, J. Hyun, E. Perez, E. Chen, C. Pettit, S. Heiner, K. Lukošiūtė, A. Askell, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, C. Olah, D. Amodei, *et al.* (2022). *Measuring Progress on Scalable Oversight for Large Language Models.* arXiv preprint arXiv:2211.03540.

[57] C. Burns, P. Izmailov, J. H. Kirchner, B. Baker, L. Gao, L. Aschenbrenner, Y. Chen, A. Adhikari, C. Olah, J. Clark, M. Chen, I. Sutskever, J. Steinhardt (2023). *Weak-to-Strong Generalization: Eliciting Strong Capabilities With Weak Supervision.* arXiv preprint arXiv:2312.09390.

[58] G. Irving, P. Christiano, D. Amodei (2018). *AI Safety via Debate.* arXiv preprint arXiv:1805.00899.

[59] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, C. D. Manning (2024). *RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval.* In Proceedings of ICLR 2024.

[60] J. Weston, S. Sukhbaatar (2023). *System 2 Attention (Is Something You Might Need Too).* arXiv preprint arXiv:2311.11829.

[61] Apple Inc. (2024). *Apple Intelligence: Personal Intelligence for iPhone, iPad, and Mac.* `https://www.apple.com/apple-intelligence/`

[62] L. Reynolds, K. McDonell (2021). *Prompt Programming for Large Language Models: Beyond the Few-Shot Paradigm.* In Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems.

[63] O. Khattab, A. Singhvi, P. Maheshwari, Z. Zhang, K. Santhanam, S. Vardhaman, S. Haq, A. Sharma, T. T. Joshi, H. Moazam, H. Miller, M. Zaharia, C. Potts (2023). *DSPy: Compiling Declarative Language Model Calls into Self-Improving Pipelines.* arXiv preprint arXiv:2310.03714.

[64] Y. Zhang, Y. Yuan, A. C.-C. Yao (2023). *Meta Prompting for AI Systems.* arXiv preprint arXiv:2311.11482.

[65] X. Wang, C. Li, Z. Wang, F. Bai, H. Luo, J. Zhang, N. Jojic, E. P. Xing, Z. Hu (2023). *PromptAgent: Strategic Planning with Language Models Enables Expert-level Prompt Optimization.* In Proceedings of ICLR 2024. arXiv:2310.16427.

[66] A. Di Cecco, L. Gianfagna (2025). *Explainable AI with Python.* 2nd ed., Springer. `https://link.springer.com/book/10.1007/978-3-031-92229-9`

[67] N. Wiener (1948). *Cybernetics: Or Control and Communication in the Animal and the Machine.* MIT Press.

[68] H. K. Khalil (2002). *Nonlinear Systems.* 3rd ed., Prentice Hall.

[69] H. J. Kushner, G. G. Yin (2003). *Stochastic Approximation and Recursive Algorithms and Applications.* 2nd ed., Springer.