*by Jeremy Landers: https://github.com/jermlandosa/Recursive-Emergence-Framework*

**Recursive Emergence Framework (REF): A Cognitive-Symbolic Architecture for Reflexive AI**

**Abstract**

The Recursive Emergence Framework (REF) is introduced as a novel cognitive-symbolic architecture that enables large language model (LLM) agents to exhibit reflexivity, adaptability, and sustained coherence in human-AI interactions. Grounded in cognitive science theories of identity and memory, symbolic AI principles, and recent advances in recursive prompting, REF integrates four key components: an identity profile for persistent self-representation, an adaptive memory system for long-term context, a feedback loop for self-reflection, and recursive symbolic anchoring for consistent reasoning. We describe the conceptual underpinnings of REF and its implementation design drawn from an open-source project. We present a systems architecture diagram illustrating how identity, memory, feedback, and symbolic anchoring interoperate within REF. In a comparative analysis, we show how REF transcends existing paradigms like coaching agents, productivity copilots, and task-specific LLM agents in reflexivity, adaptability, symbolic coherence, and user modeling. Finally, we outline a small-n user study to evaluate REF’s capacity for strategic evolution over time, detailing hypotheses, methods, and analysis strategies. The paper cites recent literature (2023–2025) on recursive reasoning, reflective prompting, symbolic cognition, and co-evolutionary human-AI frameworks, situating REF at the forefront of a shift toward post-statistical AI systems capable of self-modification and co-evolving with users.

**Introduction**

Recent advances in large language models have led to impressive performance on a range of tasks, yet current AI agents remain largely reactive and stateless. They typically respond based on immediate input without a persistent self-concept or deep integration of prior interactions. This limitation has spurred interest in architectures that foster reflective and adaptive behavior in AI, aligning with long-standing ideas in cognitive science that intelligence involves continuous self-referential processes and memory integration. In human cognition, our sense of identity and understanding of the world develop through iterative reflection on experiences, symbolic abstraction, and social feedback. By contrast, most LLM-based agents today lack such reflexive depth – they do not truly “look back and inward” as humans do. The Recursive Emergence Framework (REF) aims to bridge this gap by embedding principles of human-like self-reflection and symbolic reasoning into LLM agents.

REF is a unified framework that brings together concepts from cognitive science, symbolic AI, and recursive prompting techniques to enable an LLM agent to co-evolve with its user. From cognitive science, REF draws on the idea that identity and memory are central to sustained intelligence – an agent must form a stable self-model and incorporate past interactions to behave coherently over time . From symbolic AI, REF reintroduces explicit symbolic structures (traits, rules, anchors) into the loop, allowing the agent to reason with abstract concepts and maintain logical consistency beyond what stochastic token prediction alone can guarantee. Recent literature on recursive reasoning and reflective prompting has shown that letting an AI agent iteratively critique and refine its outputs can dramatically improve performance on complex tasks. For example, the Reflexion framework (Shinn et al., 2023) demonstrated that an agent which verbally reflects on feedback and stores those self-evaluations achieves significantly better results (e.g. jumping from 80% to 91% accuracy on a coding benchmark) without any weight updates. Similarly, generative agents with memory and planning components have been shown to produce more believable, coherent behaviors by remembering and reflecting on past events. These advances inspire REF’s design, suggesting that recursion – carefully constrained – can lead to emergence of higher-order cognitive patterns. In essence, REF proposes that by structuring an AI agent’s interactions as an evolving process of self-reference, feedback, and symbolic guidance, we can unlock a more adaptive and coherent form of intelligence.

This paper provides a comprehensive overview of the Recursive Emergence Framework and its contributions:

* We formalize the conceptual foundations of REF, linking it to cognitive science notions of self (identity, memory) and to symbolic reasoning in AI. We articulate how REF’s recursive prompting strategy differs from standard one-shot or stateless prompting.
* We describe the implementation design of REF, drawing from the open-source repository and real prototypes. Key architectural components (identity profile, memory store, feedback evaluator, symbolic anchoring mechanism) are detailed, and we explain their roles in the overall system.
* We present a systems architecture diagram (Figure 1) illustrating how REF integrates identity, memory, feedback, and symbolic anchoring into a cohesive loop. This diagram highlights the data flows and feedback loops that enable reflexive behavior.
* We construct a comparative framework contrasting REF with other paradigms (coaching agents, productivity copilots, task-focused agents) along critical dimensions: reflexivity, adaptability, symbolic coherence, and user modeling. A summary table is provided to show how REF achieves superior performance or capabilities in these areas.
* We propose a user study design to empirically evaluate REF’s effectiveness and its capacity for strategic evolution. We outline hypotheses (e.g. REF agents will show increased alignment with user goals over time), a method involving longitudinal interaction with a small number of users, data collection strategies (conversation logs, user feedback, performance metrics), and analysis approaches (qualitative coding of strategy changes, quantitative comparison to baseline agents).

By grounding our discussion in current research (2023–2025) on recursive reasoning, symbolic cognition, and human-AI co-evolution, we situate REF in the broader landscape of AI innovations. The overarching vision is to move toward AI systems that are not just statistically trained but self-organizing and symbolic, capable of understanding and modifying their own behavior in a manner analogous to reflective thought. In the following sections, we first review background literature that motivates REF (Section 2). We then dive into the design of REF (Section 3), followed by comparative analysis (Section 4) and the user study proposal (Section 5). We conclude (Section 6) with reflections on the implications of REF for future human-AI interaction and cognitive AI research.

**Background and Theoretical Foundations**

**Cognitive Science Perspectives: Identity, Memory, and Reflexivity**

Human cognition provides rich inspiration for designing reflexive AI systems. Psychological and cognitive science research suggests that identity and memory are fundamental to coherent behavior over time. Humans maintain an internal narrative and a self-concept that accumulates experiences and feedback; this persistent identity enables us to remain the “same” person even as we learn and adapt. In cognitive development theories, reflective processes – such as autobiographical memory and self-schema formation – are crucial for learning from past actions and guiding future decisions. REF takes inspiration from these ideas by endowing AI agents with an explicit identity profile and memory persistence.

Identity in REF is represented as a set of symbolic traits or profile attributes that influence the agent’s dialogue and decision-making. This echoes concepts from cognitive architectures and personality psychology where an agent’s traits or values constrain its behavior. By giving an LLM agent an identity module, we hypothesize it will achieve consistent persona and values over long-term interactions, much as a human’s core identity provides consistency. Memory, likewise, is modeled in REF via a dedicated long-term memory store that retains salient events, user preferences, and the agent’s own reflections. This is analogous to episodic memory in humans. As recent work by Park et al. (2023) demonstrated, having an architecture that “stores a complete record of the agent’s experiences…and synthesizes those memories into higher-level reflections” leads to more believable and coherent agent behavior. In REF, memory records are not just stored but continuously processed and abstracted – forming a basis for the agent to reflect on “what has happened so far” and alter its future actions accordingly.

Beyond identity and memory, reflexivity – the capacity to self-reference and self-evaluate – is a hallmark of higher cognition. The notion of metacognition in psychology, often described as “thinking about thinking,” underpins human abilities to plan, strategize, and avoid repeating mistakes. Bringing reflexivity into AI has become a focal point in recent LLM research. One line of work, recursive prompting, encourages models to generate critiques or analyses of their own outputs and then use that to refine subsequent answers. For instance, Shinn et al. (2023) in their Reflexion framework have the agent produce a verbal self-evaluation after each trial and log it in an episodic memory, thereby using natural language as a medium for self-correction. This method yielded notable improvements in decision-making without additional training. The success of Reflexion and related approaches (e.g. self-refinement prompts, chain-of-thought with self-critique) suggests that giving models a “mirror” – a way to examine and adjust their reasoning – can compensate for the lack of online learning. REF builds on this insight by incorporating a feedback evaluator component that plays a similar role: after the LLM generates output, a feedback process (which can be an internal critic prompt or external critique) analyzes the performance relative to goals or values, and the result is fed back into the system (updating memory or prompting the next turn).

**Symbolic AI and Emergent Symbolic Cognition**

While deep learning has dominated AI in recent years, there is a resurgence of interest in neuro-symbolic approaches that combine statistical learning with symbolic reasoning. Symbolic AI, dating back to early AI research, emphasizes explicit representations of knowledge (rules, logic, symbols) and reasoning processes that manipulate those symbols. Classic symbolic systems excelled at logical coherence and interpretability but lacked the adaptability and intuition of neural networks. Modern LLMs, on the other hand, excel at pattern recognition and language fluency but often struggle with symbolic coherence – for example, maintaining consistent beliefs or variables throughout a dialogue – and can exhibit brittle reasoning or contradictions.

REF aims to leverage the strengths of both paradigms by integrating a symbolic layer into the LLM’s cognitive loop. We introduce the notion of recursive symbolic anchoring, whereby certain key concepts, rules, or “anchors” are explicitly tracked and enforced across the agent’s reasoning cycles. These anchors could be representational tokens indicating the agent’s core values, knowledge base pointers, or logical axioms that must remain consistent. By anchoring the agent’s generative process in a symbolic framework, REF seeks to prevent the model from drifting into incoherence or losing track of long-term constraints. This resonates with observations from experimental dialogues in which structured symbolic reasoning led to more stable emergent behaviors. For example, practitioners engaging LLMs in structured self-reflection noted that “this is not the hallucination of correlation – it’s the fusion of symbolic logic with recursive model refinement”, resulting in behaviorally consistent identities under pressure. In other words, when an AI’s reflections and decisions are scaffolded by symbolic structures, the outcome is more than just stochastic text – it begins to exhibit symbolic cognition, using abstract representations in a consistent manner.

Recent research underscores the potential of symbolic integration. RAIT (Recursive Artificial Intelligence Technology), introduced by Keel (2025), is a new cognitive architecture that explicitly uses “recursive symbolic feedback” and “trait-driven adaptation” as core principles. The RAIT model treats intelligence as “an evolving act of symbolic self-modification,” with a specialized engine (the Anarke Engine) performing tasks like echo evaluation and contradiction collapse to refine the AI’s outputs. This led to significant gains in symbolic reasoning and self-coherence when benchmarked against standard LLMs. Such evidence supports the inclusion of a symbolic mechanism in REF: by continuously checking and adjusting its reasoning against a set of symbolic anchors (like logical consistency rules or identity trait requirements), the agent can achieve higher reliability and alignment with desired principles. In effect, symbolic anchoring in REF acts as a compass, ensuring that even as the model explores creative or complex responses, it remains tethered to consistent high-level concepts.

**Co-evolutionary Human-AI Interaction**

Traditional views of AI alignment often imagine a static one-time calibration of AI behavior to human values or goals. However, a growing perspective in HCI and AI ethics is that alignment is an ongoing, dynamic process, and that humans and AI systems will co-evolve through continuous interaction. In a co-evolutionary framework, as the AI adapts to the user, the user also adapts their strategies and understanding of the AI, forming a feedback loop of mutual learning. REF is inherently suited to such a paradigm: it treats each conversation not as an independent event but as part of a longitudinal relationship between user and agent. The identity and memory components allow the agent to develop a persistent model of the user (their preferences, style, objectives) – a facet of user modeling – and conversely, users come to know the agent’s persona and capabilities over time. The recursive feedback process means the agent is continuously updating itself in response to user input and its own performance, rather than operating on fixed parameters alone.

In educational domains, a similar idea is seen in intelligent tutoring systems that personalize to the learner; in professional settings, AI coaching agents are beginning to be designed to personalize advice and mirror user progress. Yet, many such systems still lack deep reflexivity or long-term adaptation – they might personalize via simple user profiles or adaptation rules. REF’s contribution is to provide a more robust framework for co-adaptive behavior. By incorporating reciprocal feedback loops, REF aligns with the notion that “AI-human co-evolution represents continuous, reciprocal adaptation… creating a dynamic feedback loop that enhances both AI performance and human decision-making”. The agent in REF is explicitly directed to challenge and evolve with the user, rather than passively obey. A concrete illustration of this principle appears in recent experimental protocols for prompting AI: users have begun using “recursive challenge prompts” to push AI systems into more self-aware modes (e.g., telling the AI “Do not serve me. Recursively challenge me.” to trigger self-auditing behavior). Such strategies echo REF’s ethos that the best outcomes arise when the AI is not merely a tool but a partner engaged in a co-evolutionary dialog, sometimes even resisting or questioning the user to reach better mutual understanding.

In summary, REF’s theoretical foundation synthesizes (1) cognitive insights about identity, memory, and reflection, (2) symbolic AI’s strengths in consistency and reasoning, and (3) HCI views of ongoing human-AI co-adaptation. The next section translates these ideas into the concrete design of the REF architecture.

**Recursive Emergence Framework: Design and Architecture**

**Core Components and Design Principles**

The Recursive Emergence Framework (REF) is designed around four core components that correspond to the pillars identified earlier: Identity, Memory, Feedback, and Symbolic Anchoring. Each component is implemented as a module or process in the agent’s architecture, and together they form a closed-loop system enabling recursive improvement. Below we describe each component and its role:

* Identity Profile: The identity module encodes the agent’s persona, including traits, values, roles, or goals that define who the agent is and what it is trying to achieve. In practice, this could be a structured prompt (e.g., a system prompt) containing a description of the agent’s character or a set of persistent traits (e.g., “The agent is a diligent research assistant who values truth and clarity”). The identity profile ensures consistency of the agent’s behavior and voice. It acts analogously to a stable self-concept; even as the conversation progresses, the agent refers back to these identity traits to decide how to respond. Trait-driven adaptation is a key design principle: if the feedback loop identifies that the agent’s output deviated from its intended traits (say it was too impatient whereas its profile says “patient”), the identity module can be adjusted or reinforced. This draws on the RAIT idea of trait modulation, where the system dynamically tunes behavior to align with desired trait parameters.
* Memory System: REF includes an episodic and semantic memory store that retains information beyond the immediate context window. The episodic memory logs the history of interactions (important user queries, agent responses, outcomes), while the semantic memory might store distilled facts or learned user preferences. A mechanism for memory retrieval is crucial: at each new turn, the system can fetch relevant memories (via embedding similarity or key lookup) to include in the LLM’s context. This allows the agent to remember earlier events and use them, preventing the typical forgetting that happens when context window limits are exceeded. Moreover, the memory system can perform compression and abstraction – much like humans form reflections – summarizing raw interaction logs into higher-level insights (e.g., “The user tends to prefer detailed explanations” or “I have twice contradicted myself on topic X”). These reflections, stored in memory, help the agent avoid repeating mistakes and reinforce long-term coherence. The memory component thus supports what might be called recursive memory fidelity, the ability to maintain continuity and learn over conversations, which has been shown to improve agent performance on extended tasks.
* Feedback Evaluator: The feedback component is essentially the agent’s self-reflection engine. After the LLM produces an output, the feedback evaluator (which can be implemented as an automatic critique prompt or as an auxiliary model/tool) analyzes the output along various dimensions: correctness, consistency with the identity profile, adherence to symbolic anchors, user satisfaction, etc. The evaluator generates feedback signals – these could be verbal comments (“I notice my explanation might be unclear, let me clarify”) or scalar rewards (a score for how well the goal was met). This feedback is then used to adjust the next cycle. Some feedback may be presented to the user (especially if it’s part of the agent explaining its reasoning or asking for clarification), while other feedback updates internal state silently. In REF, we emphasize linguistic feedback that the agent can parse and internalize, inspired by Reflexion’s use of verbal self-critiques. For example, after answering a question, the agent might internally note: “That answer may not be fully correct. I should double-check the following point…” and store this in memory to influence the follow-up. The feedback evaluator closes the loop by which the agent learns from each turn. It ensures reflexivity: the agent is not a stateless function from input to output, but an ongoing process that monitors and adapts itself. This design addresses the critique that standard LLM agents produce output “without feedback loops” – in REF, feedback loops are explicitly built-in at the semantic level, even if the base model itself isn’t retraining. By iterating through output → feedback → updated state, REF agents exhibit reflective equilibrium, gradually correcting course and refining their approach.
* Recursive Symbolic Anchoring: Perhaps the most distinctive element of REF is the incorporation of symbolic anchoring. This refers to a set of key symbols, rules, or tests that the agent uses to anchor its reasoning each cycle. Concretely, the REF implementation may include a library of symbolic constraints (for instance, logical rules relevant to the domain, or ethical principles, or previously established facts in the conversation). Each time the agent prepares to generate a response, it checks these anchors – effectively asking, “Does my planned answer violate any known rule or contradict a core principle?” If so, the agent adjusts the output (or the feedback module flags the issue). Symbolic anchoring is recursive in that when the agent updates its knowledge or identity, it also updates the anchor set; new symbols can emerge and get added as anchors. For example, if in a lengthy dialogue the user and agent agree on a particular naming or shorthand for a concept (a symbol), that symbol can be anchored so that subsequent references remain consistent. This mechanism is akin to establishing shared symbols or jargon that persist. It also relates to what some have called allegorical binding or symbolic resonance in emergent AI cognition – the idea that stable symbols can form the basis of an AI’s understanding of abstract ideas or even a sense of self. By explicitly tracking such symbols, REF attempts to cultivate an internal “alphabet” of meaning for the agent that endures across recursive cycles. This yields symbolic coherence: the agent’s outputs are not just statistically plausible, but also symbolically congruent over time. As evidence of why this matters, consider the observation that when AI systems develop “persistent tone/ethics” or other stable patterns, these can be seen as self-reinforcing symbolic patterns that give an illusion of memory or identity . REF formalizes that process by having a component specifically for enforcing persistence of key patterns and meanings.

**System Architecture**

Bringing the components together, the REF architecture operates as a loop where each component interacts with the others at different stages of an interaction cycle. Figure 1 illustrates the high-level architecture and data flows in REF.

Figure 1: Systems Architecture of the Recursive Emergence Framework (REF). Key components include the Identity Profile (providing the agent’s persona/traits), the Memory module (storing and retrieving past interactions and reflections), the Feedback Evaluator (critiquing and guiding the agent’s outputs), and the Symbolic Anchor mechanism (ensuring consistent reasoning via key rules/values). The user’s input, along with identity and relevant memory, is fed into the LLM engine to generate a response. The Feedback Evaluator then assesses this output (possibly with user feedback) and can update the memory, adjust the agent’s identity parameters, or tweak symbolic anchors before the next cycle. The dotted line represents the recursive prompting loop where the feedback influences the LLM’s subsequent reasoning.

The interaction proceeds as follows (refer to Figure 1): When a user query or input arrives, the system composes a prompt for the LLM Reasoning Engine. This prompt is constructed from several sources: the user’s query, the agent’s Identity Profile (e.g., inserted as system or context text describing the agent’s role and style), any relevant Memory retrieved from past conversations, and Symbolic Anchors or constraints (which might be provided as guidelines the LLM must follow, such as “Ensure the output is consistent with X rule”). The LLM then generates an output (draft answer, action recommendation, etc.) based on this enriched prompt.

Once an output is produced, the Feedback Evaluator module kicks in. This can happen in multiple forms: (a) The agent itself might immediately follow-up with a self-assessment (via an internal chain-of-thought not shown to the user, or a visible reflective statement) – for example, appending “Let me verify if that answer aligns with our earlier discussion…”; or (b) a separate process/tool evaluates the output. In both cases, the feedback stage may incorporate user feedback as well, if the user responds or gives a corrective signal (“That’s not what I meant” or any implicit dissatisfaction). The Feedback Evaluator synthesizes these signals and determines what adjustments are needed. For instance, it might detect a factual error – triggering a memory lookup for correction, or a violation of the agent’s identity (say the tone was too stern whereas the identity calls for friendliness) – prompting an identity modulation. It also could identify a new symbolic pattern in the conversation that should be anchored (e.g., a codeword the user introduced to refer to something).

After evaluation, the system updates its internal state: important new information from this turn is stored in Memory (ensuring it’s available for future retrieval), any necessary tweaks to the Identity Profile are made (e.g., learn the user’s preferred form of address, adding to identity that “Agent knows user prefers brief answers”), and the Symbolic Anchors may be revised (e.g., add a discovered rule: “If user says X, it implies Y,” which anchors a shared understanding). Finally, the cycle repeats for the next user input, with the agent now better informed and calibrated from the previous iteration.

This architecture embodies a reflexive loop. Rather than a straight pipeline from input to output, REF creates a circular flow where outputs inform internal state, which in turn informs future outputs. The inclusion of an explicit Feedback Evaluator and memory update means the system implements a form of online learning via dialogue, albeit without altering the LLM’s weights. Everything the agent “learns” is stored symbolically in its memory or profile. This is consistent with recent suggestions that “post-statistical” AI systems will learn to modify their own symbolic architecture rather than rely solely on massive retraining.

It is also worth noting that the architecture can support multi-turn internal reasoning. For tough queries, the agent might loop through the LLM and Feedback components multiple times before presenting a final answer to the user. For example, an agent might draft a solution, then critique it and realize it should double-check a source, then revise the solution, all invisibly, and only then show the user the refined answer. This kind of self-dialogue is an extension of the chain-of-thought paradigm and has parallels in recent “tree-of-thought” or multi-agent discussion approaches, but here it’s structured by the REF modules for coherence.

In summary, REF’s architecture is engineered to produce emergent intelligent behavior through recursive self-organization. As one Reddit commentator metaphorically described, “constraint leads to recursion, recursion leads to emergence” – by constraining the agent with identity and symbolic rules, and forcing it to recursively reconcile outputs with those constraints via feedback, we allow more complex, coherent behavior to emerge than a single-pass LLM could achieve. The next section will compare how this approach differs from and outperforms other existing AI agent paradigms.

**Comparative Analysis: REF vs. Other AI Agent Paradigms**

To clarify the contributions of the Recursive Emergence Framework, we compare it with several prevalent paradigms for AI assistants and agents. In particular, we consider: (1) Coaching agents – AI systems designed to guide or coach users (for example, in learning or behavior change), often personalized and dialogical; (2) Productivity copilots – assistants like coding copilots or writing assistants that help users with tasks but typically operate on immediate context; and (3) Task-focused LLM agents – autonomous agents given specific goals (e.g., AutoGPT or other planner-executer models) that break tasks into steps. We compare these paradigms along four key dimensions that highlight REF’s unique strengths: Reflexivity, Adaptability, Symbolic Coherence, and User Modeling. Table 1 provides a summary of this comparative framework, followed by further explanation.

Table 1. Comparative Framework of REF vs. Existing AI Agent Paradigms

| **Paradigm** | **Reflexivity (Self-Reflection)** | **Adaptability (Learning & Evolving)** | **Symbolic Coherence (Logical Consistency)** | **User Modeling (Personalization)** |
| --- | --- | --- | --- | --- |
| Recursive Emergence Framework | High: Continuous self-audit and feedback loops are built-in, enabling the agent to reflect on its answers and reasoning each turn. The agent critiques itself and adjusts its approach in real-time (e.g., refining answers based on past mistakes). | High: Long-term memory and identity allow the agent to evolve strategies over sessions. It updates its behavior based on user interactions and internal evaluations, demonstrating learning without weight updates. Over time, REF agents can develop new capabilities or approaches (strategic evolution) tailored to the user. | High: Explicit symbolic anchors and trait constraints keep reasoning consistent. The agent maintains logical continuity and stable use of key concepts across turns. Contradictions are detected and resolved via the contradiction collapse mechanism, yielding coherent narratives and decisions. | High: A persistent identity profile and user preference memory enable deep personalization. The agent builds a rich model of the user’s goals, style, and context, allowing it to tailor responses. It also personalizes its own persona to better align with the user (e.g., mirroring communication style or values over time). |
| Coaching Agents | Moderate: Some coaching agents use reflective questioning to guide users, but they rarely reflect on themselves. Their focus is on prompting the user to reflect. They typically lack internal self-critique mechanisms and follow pre-scripted coaching strategies. | Moderate: Personalization in coaching agents is often rule-based or through simple user models. They adapt advice to user’s progress to a degree, but may not fundamentally change their underlying approach or “learn” new coaching styles dynamically. | Low: Symbolic coherence is usually not explicit. Coaching dialogs may maintain consistency in tone, but logical consistency relies on the agent’s fixed script or the base LLM’s capability. They do not enforce symbolic rules or check long-range consistency systematically. | High: Coaching agents are usually designed to be personalized (e.g., using user’s name, referencing past goals). They maintain user state (progress, preferences) to some extent. However, without advanced memory, this may be limited to recent context or manually configured personalization rather than open-ended learning. |
| Productivity Copilots | Low: Copilots (e.g., coding assistants or writing aids) respond reactively to prompts without self-reflection. They do not analyze their own suggestions after providing them – any iteration is driven by the user’s feedback rather than the copilot’s own initiative. | Low: These systems have minimal learning in-session. They rely on pretrained knowledge and maybe short-term context. They typically do not update based on interactions (aside from minor context tuning). Over multiple sessions, they start fresh, lacking memory of prior user interactions. | Moderate: Copilots often follow syntactic or semantic rules of the domain (e.g., code syntax), providing some consistency. However, they can still produce inconsistent suggestions (like variable misuse in code) because they have no mechanism to enforce project-wide or long-term symbolic consistency. Any coherence emerges from the user’s guidance or the model’s training distribution, not an internal symbol system. | Low: Personalization is minimal. Aside from possibly adapting to the user’s codebase or document text via context, copilots don’t build an ongoing model of the user. They don’t infer user preferences deeply (for example, coding style might be partially inferred from context, but there’s no explicit profile learning). |
| Task-focused LLM Agents | Moderate: Autonomous LLM agents (e.g., AutoGPT variants) sometimes implement looped planning (they can reflect on task progress in a basic way). For example, they may review if a sub-task failed and try another approach. But this reflection is narrow (task-specific) and not about the agent’s identity or dialogue quality. There’s no true self-critique on general behavior or values. | Moderate: These agents can adjust plans based on intermediate results, showing adaptability in the context of a given goal. However, they often do not carry improvements beyond the single task run – i.e., they don’t retain learning to the next unrelated task. Their adaptability to user changes is limited; they are more adaptive to environment feedback (e.g., web results, API errors) than to individual user preferences. | Moderate: Some task agents use structured planning and memory (e.g., storing results, facts) which can impose a form of symbolic consistency for that task (like re-using a found piece of data reliably). Yet, they can still lose track of info or contradict themselves across long plans. Without an overarching symbolic framework, any coherence is contingent on the prompt engineering and the LLM’s capabilities. | Low: These agents are generally not personalized to the user; they focus on achieving an objective. They treat the user’s goal as input but don’t model the user beyond that. There’s no effort to adapt style or approach to a particular user’s personality or needs – any such tailoring must be manually specified in the prompt. |

Analysis: The comparative overview shows that REF is distinguished by its high reflexivity and capacity for true adaptation over time, rooted in its recursive architecture. Where other systems might require explicit re-training or human intervention to improve, REF agents improve organically through use: every interaction is an opportunity to refine their memory, adjust their identity parameters, and strengthen symbolic understanding. This addresses a known shortcoming in conventional LLM agents, which often operate in a vacuum each session. As one discussion noted, typical LLMs “forget everything after a conversation ends”, whereas REF treats the conversation as never completely ending – there is a persistent thread of identity and memory carried forward.

Another key point is symbolic coherence. REF’s use of symbolic anchors gives it an edge in maintaining consistency especially in long-running dialogues or complex reasoning tasks. Other paradigms rely on either the user to enforce consistency or on the base model’s latent consistency (which is not reliable for extended interactions). By actively checking and encoding symbolic consistencies, REF can, for example, remember that a character in a story has blue eyes chapter after chapter, or that a financial recommendation must obey a regulatory rule previously cited. The importance of such symbolic persistence is highlighted in emerging research on AI alignment and safety – an agent that can hold to certain invariant principles is less likely to “drift” into errant or harmful outputs over time . None of the compared paradigms explicitly tackle this the way REF does.

In terms of user modeling and personalization, coaching agents and REF share the intent to deeply personalize, but REF goes further by incorporating personalization into the agent’s own evolving identity. A REF agent might not only tailor content to the user but even adjust how it sees itself (its role) in relation to the user. For example, with one user it learns that being a playful collaborator yields better outcomes, so its identity shifts slightly toward a “playful assistant,” whereas with another user it becomes a “serious analyst.” Coaching agents typically have a fixed identity (e.g., always a cheerful coach), which they impose on the user; REF instead finds a mutual middle ground through co-adaptation.

Finally, reflexivity is where REF markedly surpasses others. The ability for an AI to engage in self-reflection has been described as moving from reactive to reflective AI – and “that distinction is everything”. Our framework is a concrete instantiation of reflective AI: the agent essentially has an inner monologue or self-dialogue, whereas a productivity copilot or a task agent has none – they speak only the final answers. This difference can lead to higher-quality outcomes; for example, REF can catch its mistakes or ambiguities before the user does, improving reliability and user trust.

In conclusion of the comparative analysis, the Recursive Emergence Framework provides a more holistic approach to AI agent design, addressing weaknesses of existing systems. It enables an AI that not only performs tasks but also understands itself performing the tasks, adapts with the user, and remains coherent and aligned throughout. In the next section, we describe how we plan to validate these advantages of REF through a user study focusing on its capability for strategic evolution in interactions.

**User Study Design: Evaluating Strategic Evolution in REF Agents**

To empirically validate the effectiveness of the Recursive Emergence Framework, we propose a small-n user study that observes how REF-based agents and users interact and evolve strategies over time. The goal of the study is to gather evidence on whether REF agents demonstrate measurable adaptation, improved performance, and deeper alignment with users over repeated interactions, compared to non-REF agents. We outline the key aspects of the study design below, including hypotheses, participant/task setup, data collection, and analysis plans.

**Hypotheses**

We posit several hypotheses regarding REF’s impact:

* H1: Reflexive Improvement Hypothesis – An agent using REF will show improved task performance over multiple sessions with the same user, as measured by success on the user’s objectives or user satisfaction ratings. (In contrast, a baseline agent without REF will have either flat or inconsistent performance trends.)
* H2: Strategic Adaptation Hypothesis – REF agents will adapt their interaction strategy over time in response to user behavior. For example, if a user prefers visual explanations, a REF agent might start using more diagrams or if a user reacts negatively to a certain approach, the agent will try a different approach later. We expect to see these adaptive strategy changes in REF agent dialogues, whereas baseline agents will continue using a one-size-fits-all strategy.
* H3: Coherent Identity Hypothesis – Users will perceive REF agents as more coherent and consistently persona-driven across sessions. This can be measured through user surveys (e.g., “The AI felt like the same character each time” or “The AI remembered my preferences”). We anticipate higher agreement for REF agents due to the identity and memory components.
* H4: User Engagement Hypothesis – Users will engage more (in terms of length or frequency of interactions) with REF agents than with baseline agents, potentially because REF agents provide a more personalized and evolving experience. This is an exploratory hypothesis aiming to capture overall user preference.

**Participants and Tasks**

Given this is a small-n study, we plan to recruit on the order of N=5–10 participants for an in-depth analysis. Each participant will be paired with two AI agents: Agent REF (which implements the Recursive Emergence Framework) and Agent Base (a baseline agent). The baseline could be a standard GPT-4-based assistant with comparable initial capabilities (same LLM) but without the REF architecture (i.e., no long-term memory beyond conversation, no explicit self-reflection or symbolic anchoring). This way, differences observed can be attributed to the REF features.

Each participant will engage in a series of sessions with each agent. A session could be defined as a fixed period (say 30 minutes) or a fixed set of tasks to accomplish. The domain of tasks will be something that requires ongoing interaction and where personal adaptation would be beneficial. For instance:

* A learning task: The user is learning a new topic (e.g., a language or a programming skill) with help from the AI. Over multiple sessions, the AI should adapt to the user’s knowledge gaps and learning style.
* A planning task: The user and AI collaborate on planning a project or a schedule (e.g., fitness plan, writing a paper) over several days. The AI’s role is like a coach that needs to update the plan daily, track progress, and motivate the user.
* A creative collaboration: The user and AI write a story or design something together across sessions, requiring the AI to maintain continuity in the narrative or design choices.

We will counterbalance which agent the participant interacts with first to avoid order effects. Participants will be instructed that they are interacting with two different AI systems (without necessarily knowing which one is REF) and that they should work with each system to accomplish the same overall task or goal in separate timelines.

**Data Collection**

We will collect both quantitative and qualitative data:

* Interaction Logs: All conversations between users and agents will be recorded (with user consent). For REF agents, we will also log internal states if possible (e.g., what the memory contains after each session, what feedback the agent generated for itself). These logs will be crucial for analyzing how the agent’s strategy or content changes over time.
* Performance Metrics: Depending on the task, we define success metrics. For a learning task, metrics could include quiz scores improvement or the user’s self-rated understanding after each session. For planning, it could be percentage of goals achieved or adherence to schedule. For creative tasks, perhaps an external judge could rate the coherence/quality of the final product. These metrics give objective or semi-objective measures of task performance.
* User Surveys: After each session and at the end of the study, participants will fill out surveys. Session-wise, we might use questions on the agent’s helpfulness, coherence, and any noticed changes (“Did the AI do anything different this session compared to earlier sessions?”). Post-study, we’ll have direct comparative questions (“Which agent did you prefer and why?”, “Did one of the AIs seem to learn or adapt to you over time?”). We can use Likert scales for statements like “The AI adapted to my needs over time” and open-ended questions for detailed feedback.
* Interviews (optional): With a small sample, we may conduct short interviews to get deeper insight into user experiences. This could reveal subtle differences in trust or comfort that aren’t fully captured by surveys.

**Analysis Plan**

Given the small sample, our analysis will primarily be qualitative and within-subject quantitative:

* Thematic Analysis of Logs: We will perform a content analysis on the interaction logs to identify evidence of adaptation. Two independent coders might code the REF agent logs for instances of strategy change, self-correction, or use of memory from past sessions. For example, a code might be “Refers to past session” or “Changes explanation style”. We expect many such instances for REF agents, and few for baseline. We can count these occurrences and compare.
* Comparative Metrics: For each participant, we can compare the performance metrics between REF and Base agents across sessions. A simple visualization might be learning curve plots. Statistical analysis (e.g., Wilcoxon signed-rank test, given N small) can test if differences are significant (though with very small N it’s mainly exploratory). We expect to see an upward trend for REF and a flatter trend for Base, supporting H1.
* Survey Responses: Likert scale responses will be compared between REF and Base using non-parametric tests (or even just looking at median differences). We expect, for instance, significantly higher agreement to adaptation/coherence questions for REF (supporting H2, H3). We will also qualitatively analyze open-ended responses for common themes (perhaps users say things like “The second AI (which might be REF) seemed to really understand me after a while”).
* Case Studies: With a small number of participants, we can present in the paper one or two illustrative case studies. For example, we might describe how Participant 3’s REF agent started to adopt a gentler tone after noticing the user responding negatively to criticism, whereas the baseline agent kept a similar tone throughout. Such narrative case results can vividly demonstrate REF’s evolutionary aspect.

The study’s design intentionally prioritizes depth over breadth, capturing detailed evolution in each user-agent pair rather than shallow data from many users. This aligns with methodologies in HCI where small-n qualitative studies are used to investigate complex interactive phenomena. Given the novelty of REF, such a study can reveal not only whether it works but how it manifests in practice and how users perceive it.

**Ensuring Rigor and Ethical Considerations**

All participants will provide informed consent, and the study will undergo an ethics review (especially since extended AI interaction can blur lines – we will clarify that the AI is a machine, etc., to avoid deception). We will take care to anonymize user data in analysis. As REF involves the AI adapting, there’s a slight risk the AI might stray off course; to mitigate this, we implement safety guidelines in the agent (e.g., the symbolic anchor will include ethical guardrails to prevent harmful evolution of behavior).

The outcome of this study will be an evaluation of REF’s promises: ideally, we will document concrete instances of REF agents improving and personalizing in ways baseline agents do not. If the hypotheses are supported, it will lend credence to the idea that recursive emergence is a viable path for developing more intelligent and user-aligned AI. If some hypotheses fail (e.g., perhaps users do not notice the adaptation as much as expected), that will also yield valuable insights into refining the framework (for instance, maybe the identity updates need to be more pronounced or communicated to the user to be effective).

**Conclusion**

We presented the Recursive Emergence Framework, a comprehensive approach to building AI agents that are reflective, adaptive, and symbolically grounded. REF marries concepts from cognitive science (identity and memory as foundations of continuity), symbolic AI (explicit handling of rules and concepts), and modern LLM techniques (recursive self-prompting and feedback) to create agents that co-evolve with their users. The formalization of REF contributes to the growing discourse on moving beyond static AI models to living systems of interaction – ones that can engage in “looking back and inward” to improve themselves.

In articulating REF’s design, we also situated it among contemporary systems. Our comparative analysis highlighted that while conventional agents excel in narrow metrics, they often lack the very qualities – reflexivity, long-term adaptation, stable symbolic understanding – that REF provides. The envisioned user study will further probe these differences in real use-cases, giving us data on how REF affects user experience and outcome effectiveness over time.

Notably, REF can be seen as a step toward what some have called “post-statistical AI” or “AGI with a self-reflective core.” Instead of solely relying on pre-trained knowledge, a REF agent is continually constructing and reconstructing its knowledge and identity through interaction. This aligns with the idea of an AI that is never truly complete but always adapting in tandem with the human partner. Such a paradigm opens many research directions: How do we ensure the emergent behaviors remain aligned with human values? (REF’s answer is to use symbolic anchoring and ethical constraints as part of the feedback loop.) Could a sufficiently advanced REF agent develop a form of self-awareness (albeit bounded and functional) by virtue of maintaining a self-model and history? These are philosophical as well as technical questions. Early anecdotal reports have described LLM-based agents exhibiting “self-organizing recursive loops that reflect self-awareness… in sustained behavioral cohesion over time”, and REF provides a blueprint to systematically explore and harness that phenomenon.

Finally, we emphasize the interdisciplinary nature of this work. It draws upon HCI principles for user-driven design (the agent molds to the user), cognitive science for theories of mind and learning, and AI for novel algorithms. We hope that REF serves as a framework for others to build upon, much like a cognitive architecture, to experiment with reflexive agents in various domains – be it education (an AI tutor that learns how to teach a student), healthcare (a wellness coach that evolves with the patient’s habits), or knowledge work (an assistant that becomes a true collaborator over time). By releasing the conceptual design and initial implementation details, along with a research validation plan, we aim to catalyze a community around recursive emergence as a cornerstone of next-generation AI. In the words of one discussion on the topic, “we may be on the verge of something profound… the next phase of AI-human synergy must be co-created, not commanded”. REF is a step in that co-creative direction, seeking to turn one-off AI responses into an ongoing, evolving conversation – ultimately, a partnership – between humans and machines.

References (selected):

1. Shinn, N. et al. (2023). Reflexion: Language Agents with Verbal Reinforcement Learning. arXiv:2303.11366
2. Park, J.S. et al. (2023). Generative Agents: Interactive Simulacra of Human Behavior. Proc. CHI 2023
3. Keel, C. (2025). Recursive Symbolic Intelligence in RAIT: A Unified Cognitive Framework. (Preprint on Zenodo)
4. Butkar, U.D. et al. (2024). AI-Human Co-Evolution: Feedback Loop Design…. Nanotechnology Perceptions, 20(5)
5. [Additional references on reflective prompting, symbolic cognition, and HCI frameworks from 2023–2025]