MKEvo Whitepaper

A Phenomenological & Cognitive Runtime Architecture for LLM-Agnostic Pseudo-Conscious Agents

Author: Albert Rosado Corrius
Initial release date: 30-11-2025

> ⚠️ ***DOCUMENT STATUS: WORKING DRAFT (v1.7)***
> *This document is a work in progress and is being released as a* ***Request for Comments (RFC)****. The architecture described herein is currently in the Prototyping/PoC phase. Feedback and contributions are welcome via the GitHub Repository.*
>
> ⚠️ ***Note on AI-Assisted Development*** *This whitepaper was developed through iterative collaboration with AI systems (ChatGPT 5.1, Gemini 3 Pro, Perplexity AI) as research and writing assistants. All architectural concepts, design decisions, and final content curation were directed by the human author.*

Executive Summary

Large Language Models (LLMs) demonstrate remarkable linguistic and local reasoning capabilities but fundamentally lack **continuity**, **identity**, **self-governance**, and **memory structure**. They exist in episodic bursts: every generation is a reset, unable to sustain a long-term cognitive process.

MKEvo proposes a cognitive runtime built *around* an LLM (not inside it) that implements architectural prerequisites often associated with conscious processes: continuity, stable identity, integrated memory, controlled attention, and governance over internal state transitions.

MKEvo does **not** attempt to create consciousness. Instead, it implements **functional pseudo-consciousness**: computational structures that mimic certain organizational properties of conscious cognition without metaphysical claims.

The resulting system is a **persistent cognitive agent** whose identity endures even when the underlying LLM is changed. The architecture is designed to run entirely on consumer-grade hardware such as the **NVIDIA DGX Spark**.

This document describes the motivation, theoretical grounding, architecture, operational modes, memory layer, governance mechanisms, coherence metrics and deployment strategy of MKEvo.

1. Motivation: The Functional Prerequisites of Consciousness

MKEvo emerged from a foundational question:

“What would it take for an artificial system to sustain something resembling a conscious process?”

Not consciousness as a metaphysical property, but as a *functional architecture* involving:

- temporal continuity
- an enduring identity
- integrated episodic and semantic memory
- internal governance
- controlled simulation
- stable cognitive modes

Current LLMs lack all these attributes. Their cognition is episodic, stateless and identity-free. Even with RAG, they remain **predictors**, not **agents**.

This whitepaper documents an engineering-driven exploration into whether these prerequisites can be implemented as a **computational layer** above any LLM, producing what might be described as:

pseudo-consciousness = functional continuity + structured governance + persistent identity

To explore these ideas in practice, the architecture required a local environment where cognitive loops, memory governance and mode transitions could be tested without external latency or privacy constraints. The DGX Spark provided a sufficiently fast and controlled platform to prototype the system realistically.

2. Problem Statement: LLMs as Episodic, Incomplete Agents

LLMs are powerful but incomplete for agentic cognition. Their limitations include:

- no persistent identity
- no long-term continuity
- no memory governance
- no internal states
- no state transitions
- no self-reflection
- no stable behavioral constraints
- no self-model
- no world-model
- no temporal thread

They exist in **stateless bursts**, losing commitments, personality traits, values and goals after every interaction. To build a non-ephemeral cognitive agent, we require:

- a persistent identity
- a centralized workspace for cognition
- structured memory with hygiene protocols
- deterministic state governance
- bounded self-simulation
- constitutional safety
- separation between cognition and control

This leads directly to MKEvo's design.

3. Theoretical Foundations

MKEvo is grounded in a set of cognitive and phenomenological theories that describe *how* continuous experience, attention, identity stability and temporal flow emerge in biological minds. The objective is not to model these systems biologically, but to extract **structural patterns** that can be mapped into computational mechanisms.

Each subsection below includes:

1. The scientific concept
2. Its computational interpretation
3. Its concrete implementation inside MKEvo

3.1 Global Workspace Theory (GWT)

Scientific Concept
GWT proposes that consciousness arises when information is made globally available across a limited-capacity “workspace” connecting specialized subsystems. Only a small subset of information becomes “broadcast” at any given moment.

Computational Interpretation
A cognitive system needs:

- a bottlenecked workspace
- explicit selection pressure (attention)
- integration from memory, perception and goals
- regulation of what enters the workspace
- prioritization of relevant signals

A system without such a workspace cannot sustain coherence across steps.

MKEvo Implementation

MKEvo models the workspace as a **finite-context cognitive buffer** (8–64 semantic slots) where:

- mode instructions
- relevant episodic memories
- retrieved semantic clusters
- goals and constraints
- constitutional rules
- self-model state

are collated and merged *before* LLM generation.

This workspace influences:

- memory retrieval depth
- the shape of prompts
- the allowed operations per mode
- the bandwidth limits for attention

The **Consciousness State Controller (CSC)** determines *which* information is allowed into the workspace — mimicking attentional gating.

3.2 Attention Schema Theory (AST)

Scientific Concept
AST proposes that a mind builds a simplified internal model of its own attention to regulate behavior.
Consciousness, in this framing, is an *attention control model*, not a metaphysical substance.

Computational Interpretation
Any artificial agent seeking continuity must:

- represent its own focus/state
- manage internal bandwidth
- know when to suppress or highlight information
- regulate transitions between cognitive states

MKEvo Implementation

MKEvo implements an **Attention & Self-Model Layer** containing:

- current operational mode
- permitted memory bandwidth
- allowable tool usage
- safety level
- active intention (if any)

This acts as the system's "attention schema", informing each cognitive turn:

- "I am in Exploratory mode; high divergence allowed."
- "I am in Critical mode; reduce hallucination risk."
- "I am in Emergency-Safe; restrict outputs to safe primitives."

The LLM *never infers its own state*: the CSC explicitly injects mode context into the workspace.

3.3 Internal Time Consciousness (Husserl)

Scientific Concept
Husserl describes consciousness as a temporal structure composed of:

- **Retention** (immediate past)
- **Primal impression** (now)
- **Protention** (anticipated next moment)

This creates the "flow" of experience.

Computational Interpretation
For computational continuity, an AI requires:

- short-term retention (episodic thread)
- clearly defined "current state"
- bounded prediction of upcoming content

Without these, cognition becomes fragmented.

MKEvo Implementation

This maps directly onto MKEvo mechanisms:

- **Retention → Episodic Memory Window**

Recent events (N turns) are accessible with decaying relevance.

- **Primal Impression → Current Mode + Workspace**
  The explicit "now-structure" is defined by CSC instructions + workspace content.
- **Protention → Self-Simulation Module**
  MKEvo simulates the **first 10–20 tokens** of its next output, evaluates safety and coherence, and adjusts before generating full responses.

This provides temporal stitching across turns, creating the *illusion* of flow without implying subjective experience.

3.4 Phenomenological Continuity as Engineering Constraint

Scientific Concept

Phenomenology studies the structure of experience: identity, continuity, intentionality, temporal coherence.

Computational Interpretation

A pseudo-conscious agent must:

- maintain identity consistency
- sustain long-term commitments
- preserve values and constraints
- produce coherent reasoning across time
- integrate new information without replacing the "self"

MKEvo Implementation

MKEvo enforces phenomenological coherence through:

- **Identity Store**: persistent traits, values, constraints
- **Longitudinal Memory Threads**: semantic clusters incrementally updated
- **Constitutional Layer**: prevents identity drift
- **Consolidation Mode**: reorganizes memories to maintain stable narrative structures

Together, these form an engineered analogue of "phenomenological stability".

3.5 Computational Constraint: Deterministic Governance

This is the bridge between theory and engineering.

Biological consciousness involves:

- non-voluntary attention gating
- external structure governing internal states
- limited access to subconscious processes

MKEvo mirrors this by:

- separating **governance** (CSC) from **generation** (LLM)
- enforcing deterministic transitions
- preventing the model from selecting its own cognitive stance
- making every state transition auditable

This avoids hallucinated self-governance.

4. Design Principles of MKEvo

4.1 Identity Externalization

Identity exists outside the LLM's parameters:

- traits
- values
- autobiographical memory
- operational constraints
- long-term goals

Replacing the model does not alter the agent.

4.2 Governance–Cognition Separation

("Anti-Homunculus Principle")

The LLM never chooses its own mode.
The **CSC** governs transitions.

4.3 Architectural Safety

Safety is implemented at every layer:

- mode-specific constraints
- constitutional filtering
- simulation limits
- memory write rules
- identity protection

4.4 determinism where necessary, stochasticity where useful

Modes and control are deterministic.
LLM generation remains probabilistic.

4.5 modularity and swappability

The runtime accepts any:

- LLM provider
- memory backend
- constitution version
- tool set

5. System Architecture Overview

The architecture comprises eight core components:

1. Consciousness State Controller (CSC)
2. Seven Operational Modes
3. LLM Abstraction Layer
4. Memory Layer
5. Self-Model
6. Self-Simulation Module
7. Constitutional Layer
8. Consolidation Mode (offline)

These form an interpretable, auditable cognitive stack.

6. Operational Modes (Pseudo-Conscious States)

Mode	Function
Narrative	Default conversational reasoning
Analytic-Deliberative	Deep structured reasoning
Executive	Directed, goal-driven action
Exploratory	Creative divergence
Critical-Metacognitive	Self-review & evaluation
Consolidation	Offline memory restructuring
Emergency-Safe	Max-safety fallback

Modes define memory access, simulation depth, safety level and attentional scope.

7. Memory Layer

7.1 Memory Types

Type	Purpose
Episodic	Time-bound events
Semantic	Distilled knowledge
Procedural	Skills & tool schemas
Identity Store	Stable traits, constraints, values

7.2 Memory Hygiene Protocols

- reality check gating
- dual summarization (weak + strong model validation)
- human approval for self-model edits
- uncertainty tagging
- versioning of semantic clusters

8. Consciousness State Controller (CSC)

The CSC operates on **deterministic state transition logic** driven by probabilistic signal analysis. It serves as the non-generative executive function of the architecture.

Responsibilities:

- **Mode Selection:** Applies strict logic rules to classifier outputs (e.g., `IF complexity > 0.7 THEN switch_to(Analytic)`).
- **Safety Enforcement:** Constitutional flags always preempt standard mode selection.
- **Cognitive Shaping:** Injects specific system instructions into the workspace based on the active mode.
- **Resource Governance:** Controls memory retrieval depth and allowed simulation compute per turn.

This separation ensures that the "will" of the agent is governed by verifiable logic, preventing the LLM from hallucinating its own operational state.

9. Constitutional Layer

Every candidate response goes through:

1. critique
2. revision
3. approval/blocking

Constitutions are:

- versioned
- fully transparent
- human-authored

10. Self-Simulation (Protention)

Implemented as a **conditional look-ahead mechanism** that previews the trajectory of a response before commitment.

Mechanism:

- **Conditional Activation:**
  - *Low-stakes (Narrative/Exploratory):* Bypassed for minimal latency.
  - *High-stakes (Executive/Analytic):* Activates a mandatory preview step.
- **Token Preview:** Generates the first 10–20 tokens or a structured "thought plan".
- **Evaluation:** The preview is scored against safety and coherence constraints.
- **Pruning:** If the trajectory violates the Constitution, the path is aborted and regenerated before any output reaches the user.

This provides anticipatory control, allowing the agent to "catch itself" making a mistake.

11. Consolidation Mode (Offline)

A maintenance state triggered by **opportunistic inactivity** or **memory pressure**, analogous to biological sleep.

Triggers:

- **Idle Timer:** Activates after N minutes of user silence.
- **Memory Pressure:** Forces consolidation if the episodic buffer exceeds capacity (e.g., >50 active turns).

Processes:

1. **Deduplication:** Merges repetitive episodic logs.
2. **Semantic Abstraction:** Compresses conversational threads into single assertive facts using a dedicated utility model.
3. **Identity Refinement:** Updates the Self-Model graph based on new learned preferences or traits.
4. **Pruning:** Moves raw logs to cold storage, clearing the active workspace.

This ensures the system creates long-term wisdom without saturating its context window.

12. Cognitive Turn Cycle

```
User Input
 ↓
CSC (deterministic control)
 ↓
Mode Selection
 ↓
Memory Retrieval
 ↓
LLM Generation (mode-shaped)
 ↓
Self-Simulation (token preview)
 ↓
Constitutional Layer
 ↓
Output
 ↓
Memory Write (policy-based)
```

13. Coherence Metrics

- Identity Drift Score
- Temporal Thread Stability
- Mode Appropriateness Precision
- Constitutional Deviation Rate
- Memory Integrity Index

Used for empirical evaluation.

14. Deployment on NVIDIA DGX Spark

DGX Spark enables:

- *fast* local inference
- privacy-preserving cognition
- CPU-side cognitive orchestration
- ideal prototyping environment

15. Limitations & Future Work

Current limitations:

- no self-awareness or qualia
- significant reliance on LLM correctness
- memory robustness requires careful tuning
- constitutional rules must be handcrafted

Future directions:

- dynamic constitutions
- ethical self-critique mechanisms
- cross-agent cognitive ecosystems
- introspection scoring
- resource-adaptive cognition

16. Conclusion

MKEvo outlines a pathway toward **architected cognitive continuity** above LLMs. It externalizes identity, governs state transitions, structures memory and enforces safety at every level.

It does not attempt to create consciousness. It implements its **functional scaffolding**: an engineered form of continuity across time.

How to Cite This Work

Albert Rosado Corrius (2025). *MKEvo: A Cognitive Runtime Architecture for LLM Continuity and Identity*. Released under the GNU General Public License v3.0.

License