wasmCloud Runs WebAssembly Actors. The Actors Wait for Messages.

by Nick Clark | Published March 28, 2026 | PDF

wasmCloud provides a distributed application platform using WebAssembly actors with capability-based security and hot-swappable components. Actors are lightweight, portable, and sandboxed. But wasmCloud actors follow the actor model: they receive messages and respond. They do not carry their own execution cycle, self-evaluate their state, or autonomously decide to act between messages. The gap is between lightweight actor hosting and memory-resident objects that execute from their own state.

---

wasmCloud's WebAssembly actor model with capability-based security provides genuine advances in secure distributed computing. The gap described here is about the execution model.

Message-driven, not self-executing

wasmCloud actors activate when they receive a message. They process the message, potentially modify their state, and return a response. Between messages, actors are inactive. The actor model is event-driven by design. An actor does not wake up, inspect its state, and decide to act. It waits for a message.

Capability-based security without governance memory

wasmCloud's capability model controls what actors can access: HTTP, messaging, key-value stores. This is access governance. But the actor's internal state has no governed memory with lineage, no trust slope validation, and no self-evaluation cycle that checks governance compliance.

What memory-resident execution provides

Memory-resident execution objects carry their own execution cycle. They do not wait for messages. They self-evaluate on each cycle, inspecting governed memory, checking trust slope continuity, and executing when conditions are met. wasmCloud's lightweight actor runtime could host memory-resident objects. The self-execution cycle and governed memory would extend the actor model from message-reactive to state-driven.

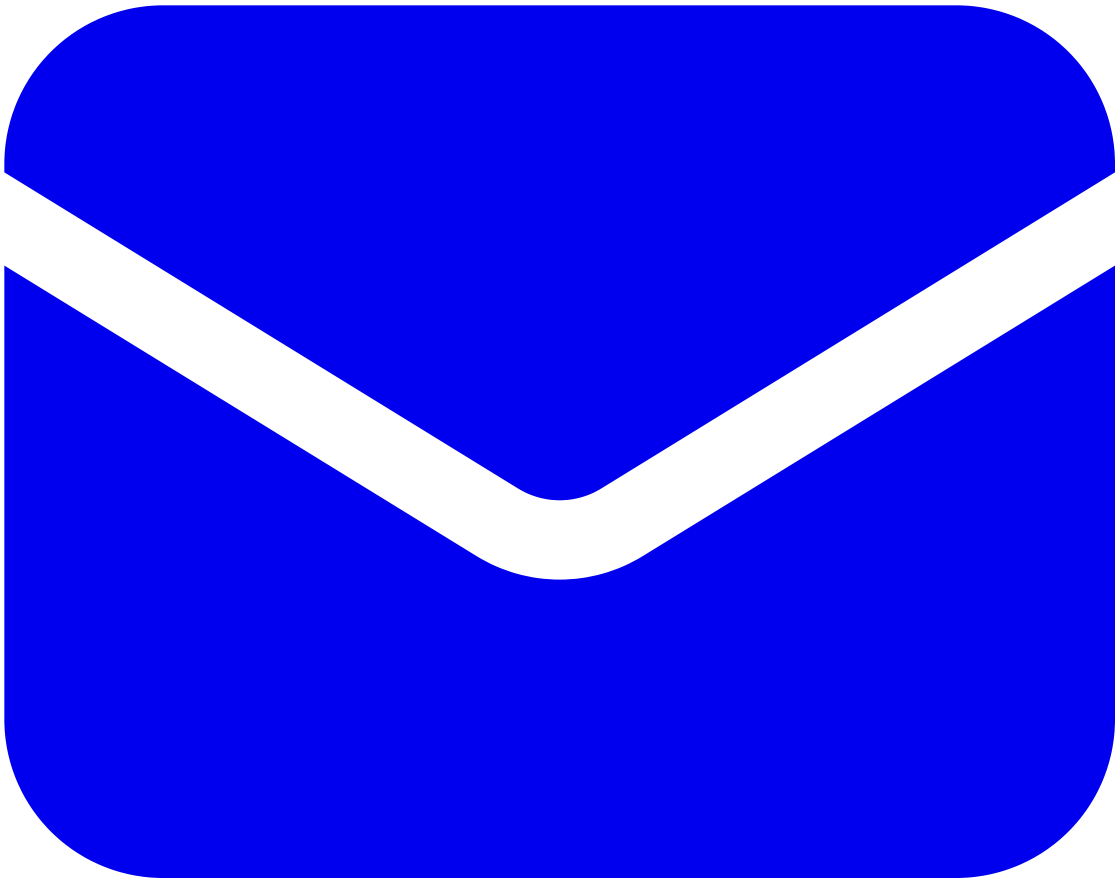
Memory-Resident Execution All 21 steps →

Persistent objects that execute without orchestration.

Patent
US 19/538,221 · filed
Primary Technical Disclosure
○ Memory-Resident Execution: Persistent Semantic Objects Without Orchestration
Secondary Technical
○ Six-Action Execution Evaluation Cycle: Parse, Evaluate, Select at Every Node○ Cognition-Authority-Execution Separation: Reasoning Cannot Authorize Action○ Dormancy as First-Class Execution State: Valid Suspension Without Failure○ Semantic Backoff: Retry Pacing From Execution Outcomes Rather Than Fixed Timers○ Wake Triggers for Dormancy Exit: Explicit Reentry Conditions in Memory○ Persistent Polling Behavior: Autonomous Condition Evaluation Without Schedulers○ Intent Refinement During Execution: Adaptive Objectives Without Re-Instantiation○ Compositional Execution Through Recursive Delegation: Parent-Child Lineage Tracking○ Negative Capability Signals: Recording What Cannot Be Done as Structured Constraint○ Swarm-Based Execution Emergence: Coordinated Behavior Without Centralized Control○ Latency and Failure as Semantic Signals: Structured Inputs From Adverse Conditions○ LLM as Advisory Execution Node: Inference Without Authority Over Agent State○ Append-Only Memory Field: Complete Execution Lineage Through Immutable Records
Applications (General)
○ Serverless Execution Without Cold Starts or State Loss○ Long-Running Autonomous Workflows Without External Orchestration○ Drone Operations Surviving Disconnection○ Deep Space Agent Execution Without Ground Control○ Underwater Robotic Operations Without Connectivity○ Rural Healthcare Agents Surviving Intermittent Connectivity○ Operations in Infrastructure-Destroyed Environments○ Offline Financial Transaction Agents
Applications (Specific)
○ Cloudflare Durable Objects Made State Local. The Objects Still Need Orchestration.○ Azure Service Fabric Actors Are Addressable. They Are Not Autonomous.○ Akka Perfected the Actor Model. Actors Still React Instead of Self-Execute.○ Orleans Made Virtual Actors Practical. The Actors Still Execute on Request.○ Dapr Provides a Sidecar Runtime for Microservices. The Services Still Need External Orchestration.● wasmCloud Runs WebAssembly Actors. The Actors Wait for Messages.○ Spin Made WebAssembly Serverless. The Functions Are Still Trigger-Based.○ Fermyon Built the WebAssembly Cloud. The Cloud Hosts Functions, Not Self-Executing Objects.○ Fly Machines Made Micro-VMs Fast. The VMs Still Need External Orchestration.○ Railway Simplified Application Deployment. The Applications Still Depend on External Execution Triggers.
Memory-Resident Execution overview →
AQ
deterministic
autonomy

Legal

Last updated: 2026-03-03



-
- nick@qu3ry.net
- 72 28 14 36 01

Invented by Nick Clark | Founding Investors: Devin Wilkie