Nick Clark · Published April 25, 2026 · PDF

Execution Continuity for DDIL Coalition C2: Memory-Resident Tasking Across Disconnected, Trust-Divergent Tactical Networks

Coalition command-and-control at the tactical edge runs over disconnected, intermittent, and limited-bandwidth (DDIL) links, where partners share no common scheduler, no persistent connection, and no shared trust authority, yet tasking must survive comms blackouts and resume coherently when nodes reappear. This application is built on Memory-Resident Execution, disclosed in United States Patent Application 19/538,221, which carries execution state inside the task object itself so that progress, decision history, and eligibility conditions persist across nodes and across administrative or trust domains. It lets a tasking object go dormant during a blackout and reenter execution later, evaluated locally by whichever coalition node holds it, without synchronized control or shared authorization infrastructure.

What This Application Specifies

This application specifies a way to carry mission tasking across a coalition tactical-edge network as a self-contained, memory-bearing execution object rather than as a call coordinated by a central tasking server. Each tasking object holds three things internally: an intent field that encodes the machine-readable objective and its

constraints, a context block that encodes identity, trust scope, and execution-relevant metadata, and a memory field that records an append-only history of everything done with the object so far. When a coalition node receives the object, it runs a local execution evaluation cycle: it parses the intent, evaluates the context block against its own locally applicable policy, reads the prior execution records, and selects one action from a fixed set of execution, mutation, delegation, dormancy, reentry, or termination. The node then performs that action and appends a new record to the memory field.

Two properties of the underlying disclosure make this fit DDIL coalition C2. First, evaluation is local: each node decides what to do "without reliance on centralized coordination," and the disclosure is explicit that nodes in different trust zones, resource environments, or policy regimes "may lawfully select different execution actions for the same semantic object" while preserving a single auditable lineage. Second, dormancy is a first-class execution state, not an error: a node can deliberately suspend a tasking object, hold it intact with its full history, and let any later node evaluate reentry conditions derived from elapsed time, accumulated outcomes, or observed context. The disclosure's federated execution configuration states the case directly: semantic objects propagate across multiple administrative or trust domains, each domain evaluates them under its own policy, and outcomes from one domain are preserved in the memory field for evaluation by nodes in other domains "without requiring synchronized control or shared authorization infrastructure."

Why It Matters

DDIL is the defining constraint of coalition operations at the edge. Links drop for minutes or hours, bandwidth is too thin to stream state back to a headquarters, and partner forces operate under separate national systems that do not share a directory, a clock, or a trust root. Conventional tasking assumes the opposite. As the disclosure observes of conventional systems generally, execution state is "maintained externally by runtimes, schedulers, orchestration layers," requiring context to be "reconstructed at each invocation or managed through centralized workflow engines," with external

controllers tracking progress, retries, and failure handling. The disclosure further notes that these approaches introduce "points of failure when execution spans asynchronous systems, disconnected environments, or trust-divergent domains." That is precisely the coalition tactical edge.

When the controller is unreachable, controller-bound tasking stalls or silently drops. When two partners run separate authorization servers, a task authorized in one cannot be acted on in the other without a federation handshake that the link cannot afford. Memory-Resident Execution moves the state and the decision logic into the object that crosses the boundary, so a blackout becomes a dormancy rather than a lost task, and a trust boundary becomes a local policy evaluation rather than a blocked call.

How It Composes With the Domain

A tasking object is instantiated with its intent (the objective and constraints), its context block (originating unit, trust scope, mission policy references), and an initial memory field. It then propagates across the coalition network by whatever transport is available: a mesh radio hop, a periodic satellite window, or physical courier of the serialized object on storage media. Because the disclosure treats the object as a self-contained, memory-bearing entity whose semantics are invariant across "infrastructure topology, connectivity model, or administrative domain," the same object behaves consistently whether it lands on a well-resourced operations node or a constrained edge device.

At each coalition node, local policy evaluation does the work that a shared trust authority would otherwise do. The disclosure specifies a local policy evaluator that interprets policy references embedded in the object "in view of execution context at the execution node," and states that this evaluation "does not rely on centralized authorization servers, shared registries, or global trust authorities." A national node evaluates the incoming tasking against its own rules of engagement, caveats, and resource constraints and reaches its own authorization outcome: permit, defer, restrict,

mutate, or reject. The disclosure is explicit that different nodes may reach different outcomes for the same object, and that each outcome is recorded as a trace entry, producing an auditable history that reflects heterogeneous policy environments. For coalition C2, that audit trail is the after-action record of who acted on a task, under which caveat, and why.

Dormancy and reentry carry tasking across the disconnection itself. When a node determines execution is currently inadvisable, for example because a required asset is unreachable or a release condition is unmet, it transitions the object to dormancy: a deliberate decision that preserves intent, history, and eligibility for future reentry. The disclosure describes wake triggers recorded in the memory field, corresponding to elapsed time, accumulated outcomes, changes in context, or satisfaction of prerequisite conditions, and states that the object "remains dormant until a wake trigger is satisfied, at which point reentry evaluation may occur without centralized scheduling or external notification mechanisms." Polling here is local condition checking, which the disclosure notes "does not require global scheduling, persistent connections, or centralized orchestration." A reentering node resumes from the retained memory field rather than reconstructing state.

Decomposition across partners maps to delegation. A coordinating node can delegate sub-objectives to subordinate tasking objects that execute independently on partner nodes while preserving lineage through memory-linked references; returned outcomes are aggregated back into the parent's memory field. The disclosure describes this compositional execution and the resulting swarm-style behavior, in which autonomous objects pursue related objectives across "heterogeneous execution nodes and trust domains," with coordination arising "implicitly through memory-resident execution state, lineage references, and repeated local evaluation" rather than through a consensus protocol the network cannot sustain.

What This Enables

Concretely, this lets coalition tasking survive the conditions that break controller-bound C2. A tasking object dispatched before a comms window closes continues to be evaluated and acted on by edge nodes during the blackout, going dormant where it cannot proceed and reentering when conditions allow, all without a heartbeat to headquarters. A task that crosses into a partner's domain is evaluated against that partner's policy locally, so national caveats are enforced at the point of action rather than negotiated over the link. Partial progress counts: the disclosure treats partial execution that yields "intermediate results, state advancement, constraint satisfaction, or actionable information" as a meaningful recorded outcome, so a task that advances halfway before a node loses connectivity carries that advancement forward to the next node instead of restarting.

Failure and latency become signal rather than dead air. The disclosure interprets timeouts, non-response, and node failure as "semantic execution signals indicative of environmental constraints, resource availability, trust conditions, or execution feasibility," recorded structurally and used to adjust pacing through semantic backoff and to redirect execution toward an alternative node. In a contested-comms environment, that means a tasking object reasons about where it is likely to succeed and defers where it is not, instead of hammering an unreachable controller. And because every decision, mutation, dormancy transition, and reentry is appended to the memory field with origin node, policy reference, outcome, and a cryptographic signature, the object arrives at debrief carrying its own complete, verifiable provenance across every domain it touched.

Boundary Conditions

The disclosure governs how a tasking object carries state, evaluates locally, and persists across disconnection; it does not supply the radios, waveforms, cross-domain guards, or cryptographic key management that a real coalition network requires, and those remain

the responsibility of the deploying programs and their accreditation authorities. Local evaluation means coalition partners can reach different outcomes for the same task by design; achieving a desired joint effect still depends on the policies each partner encodes, and the framework enforces separation of cognition, authority, and execution rather than guaranteeing mission agreement. The memory field provides auditability and tamper-evident records through per-entry signatures, but securing the object in transit, validating signatures, and resisting capture or spoofing of a physically conveyed object are deployment-level security concerns outside the execution semantics. The absence of centralized coordination does not preclude later verification or eventual consistency; it means no single authority gates execution sequencing, which is a deliberate trade that shifts assurance from a central gate to per-node policy and the auditable lineage. Real-time effects still depend on a node actually receiving the object within a usable window; the framework preserves continuity across intermittency but cannot manufacture connectivity that does not exist.

Disclosure Scope

The execution model described here, namely memory-resident execution of persistent executable objects that carry intent, context, and an append-only execution history, evaluate locally without centralized coordination, and transition through mutation, delegation, dormancy, reentry, and termination, is disclosed in United States Patent Application 19/538,221. All statements in this article about what the invention does trace to that disclosure, including federated execution across administrative or trust domains without synchronized control or shared authorization infrastructure, local policy evaluation without centralized authorization servers or global trust authorities, dormancy as a first-class state with memory-resident wake triggers, and edge-oriented execution under intermittent connectivity. The DDIL coalition command-and-control framing, including references to rules of engagement, national caveats, contested communications, accreditation, and tactical transport, is external domain context provided to illustrate one faithful enabling application; it is not part of the patent

disclosure and is not a claim about any specific defense system, program, standard, or regulatory regime. Generalized references to controllers, schedulers, transports, and partner systems describe categories of implementation rather than any particular commercial product.

Memory-Resident Execution (/memory-resident-execution)

All 40 steps → (/inventive-steps)

Persistent objects that execute without orchestration.

U.S. 19/538,221 (/patents/19-538221)

PRIMARY TECHNICAL DISCLOSURE

- Memory-Resident Execution: Persistent Semantic Objects Without Orchestration (/articles/memory-resident-execution-persistent-semantic-objects-without-orchestration)

SECONDARY TECHNICAL

- Six-Action Execution Evaluation Cycle: Parse, Evaluate, Select at Every Node (/articles/memory-resident-execution/execution-cycle)
- Cognition-Authority-Execution Separation: Reasoning Cannot Authorize Action (/articles/memory-resident-execution/cognition-authority-separation)
- Dormancy as First-Class Execution State: Valid Suspension Without Failure (/articles/memory-resident-execution/dormancy-state)
- Semantic Backoff: Retry Pacing From Execution Outcomes Rather Than Fixed Timers (/articles/memory-resident-execution/semantic-backoff)
- Wake Triggers for Dormancy Exit: Explicit Reentry Conditions in Memory (/articles/memory-resident-execution/wake-triggers)
- Persistent Polling Behavior: Autonomous Condition Evaluation Without Schedulers (/articles/memory-resident-execution/persistent-polling)
- Intent Refinement During Execution: Adaptive Objectives Without Re-Instantiation (/articles/memory-resident-execution/intent-refinement)
- Compositional Execution Through Recursive Delegation: Parent-Child Lineage Tracking (/articles/memory-resident-execution/recursive-delegation)

- Negative Capability Signals: Recording What Cannot Be Done as Structured Constraint (/articles/memory-resident-execution/negative-capability)
- Swarm-Based Execution Emergence: Coordinated Behavior Without Centralized Control (/articles/memory-resident-execution/swarm-execution)
- Latency and Failure as Semantic Signals: Structured Inputs From Adverse Conditions (/articles/memory-resident-execution/failure-signals)
- LLM as Advisory Execution Node: Inference Without Authority Over Agent State (/articles/memory-resident-execution/llm-advisory-node)
- Append-Only Memory Field: Preserving Execution Lineage Through Appended Records (/articles/memory-resident-execution/append-only-memory)

APPLICATIONS · GENERAL

- **Execution Continuity for DDIL Coalition C2: Memory-Resident Tasking Across Disconnected, Trust-Divergent Tactical Networks (/articles/memory-resident-execution/defense-tactical-edge-ddil)**
- Stateful Serverless: Eliminating Cold Starts and State Loss in FaaS (/articles/memory-resident-execution/serverless-persistence)
- Long-Running Business Workflows Without an Orchestration Engine (/articles/memory-resident-execution/long-running-workflows)
- Autonomous Drone Operations Surviving Ground Control Link Loss (/articles/memory-resident-execution/autonomous-drone-operations)
- Deep Space Agent Execution Without Ground Control (/articles/memory-resident-execution/space-exploration-agents)
- Autonomous Underwater Vehicle Mission Autonomy Without Surface Connectivity (/articles/memory-resident-execution/underwater-robotics)
- Offline Clinical Agents for Rural Healthcare With Intermittent Connectivity (/articles/memory-resident-execution/rural-healthcare-agents)
- Disaster Response Software That Works When Infrastructure Is Destroyed (/articles/memory-resident-execution/disaster-zone-operations)
- Offline Payment Agents That Stay Compliant When the Network Drops (/articles/memory-resident-execution/offline-financial-agents)

APPLICATIONS · SPECIFIC

- Cloudflare Durable Objects Made State Local. The Objects Still Need Orchestration. (/articles/memory-resident-execution/durable-objects)
- Azure Service Fabric Actors Are Addressable. They Are Not Autonomous. (/articles/memory-resident-execution/azure-actors)

- Akka Perfected the Actor Model. Actors Still React Instead of Self-Execute. (/articles/memory-resident-execution/akka)
- Orleans Made Virtual Actors Practical. The Actors Still Execute on Request. (/articles/memory-resident-execution/orleans)
- Dapr Provides a Sidecar Runtime for Microservices. The Services Still Need External Orchestration. (/articles/memory-resident-execution/dapr)
- wasmCloud Runs WebAssembly Actors. The Actors Wait for Messages. (/articles/memory-resident-execution/wasmcloud)
- Spin Made WebAssembly Serverless. The Functions Are Still Trigger-Based. (/articles/memory-resident-execution/spin)
- Fermyon Built the WebAssembly Cloud. The Cloud Hosts Functions, Not Self-Executing Objects. (/articles/memory-resident-execution/fermyon)
- Fly Machines Made Micro-VMs Fast. The VMs Still Need External Orchestration. (/articles/memory-resident-execution/fly-machines)
- Railway Simplified Application Deployment. The Applications Still Depend on External Execution Triggers. (/articles/memory-resident-execution/railway)

Memory-Resident Execution overview → (/memory-resident-execution)