Anduril's Lattice Plans Missions Without Speculative Containment

by Nick Clark | Published March 27, 2026 | PDF

Anduril's Lattice platform represents a serious approach to defense autonomy: fusing sensor data from diverse assets, maintaining a common operating picture, and coordinating autonomous systems across domains. The engineering is capable and the operational concept addresses real military requirements. But Lattice's mission planning generates and evaluates courses of action without maintaining speculative containment boundaries, branch classification, or governed promotion thresholds. Plans are evaluated and selected. They are not contained, matured, and promoted through a structured cognitive process. Defense forecasting requires this discipline.

What Anduril built

Lattice solves a genuine integration problem in defense operations. Sensors, effectors, and command elements from different manufacturers, different services, and different classification levels need to share information and coordinate action. Lattice provides the data fusion and command integration layer that makes cross-domain autonomous coordination possible. The platform handles sensor data ingestion, object tracking, threat classification, and asset allocation across heterogeneous systems.

Mission planning within Lattice generates courses of action based on available assets, threat assessments, rules of engagement, and operational objectives. The system evaluates options and recommends actions to human operators. The human-on-the-loop model ensures that lethal decisions require human authorization. This is appropriate and reflects responsible engineering.

The gap between course-of-action generation and cognitive forecasting

Course-of-action generation produces options and evaluates them against criteria. Cognitive forecasting maintains those options as speculative branches with independent state, evolving consequences, and structural containment that prevents speculative reasoning from contaminating the active mission plan. The distinction matters when plans must adapt to evolving situations.

A defense system that generates three courses of action and selects the best one is performing optimization. A system that maintains all three as live speculative branches, continuously updating their projected outcomes as the situation evolves, classifying each by risk profile and time-to-commitment, and structurally containing them so that branch evaluation does not affect the currently executing plan, is performing cognitive forecasting. The second system can switch plans faster and with better information because the alternative plans have been maturing in parallel.

In defense operations, the difference between these approaches is measured in the quality of decisions under time pressure. When the situation changes rapidly, a system with contained speculative branches already has alternative plans at varying stages of maturity. A system that must regenerate courses of action from scratch faces computational and cognitive delay at exactly the moment speed matters most.

Why containment is critical for defense

The containment boundary is not a convenience. In defense applications, it is a safety requirement. Speculative reasoning about engagement options must be structurally isolated from the active engagement plan. A system that allows speculative branch evaluation to influence current asset positioning or sensor allocation before the branch is formally promoted creates unpredictable behavior that degrades operational trust.

Branch classification matters equally. An exploratory branch that imagines a flanking maneuver must be structurally distinguished from a viable branch that has been partially validated against current intelligence. Both exist in the planning graph. They carry different promotion thresholds and different operational implications. Without classification, all speculative branches look the same to the system, and the operator cannot distinguish mature alternatives from raw speculation.

What a forecasting engine enables for defense

With planning graphs as first-class cognitive structures, Lattice maintains a persistent tree of mission alternatives. Each branch evolves independently as the situation changes. Branches that become infeasible due to asset losses or intelligence changes are automatically reclassified or pruned. Branches that gain viability as conditions shift are promoted toward the activation threshold. The operator sees not just the current plan but the maturity state of all alternatives.

The executive graph aggregation property provides a cognitive record of planning decisions across the mission. Why was this branch promoted? What intelligence triggered the reclassification of that alternative? When did the system first consider the option that ultimately became the active plan? This record is essential for after-action review and for building institutional planning knowledge.

The structural requirement

Anduril's Lattice solves the integration and coordination problem. The remaining gap is in the planning layer: the ability to maintain speculative branches with containment discipline, classify them by maturity and risk, and promote them through governed thresholds. Defense operations require planning systems that think ahead without acting prematurely. That requires forecasting as a cognitive primitive, not course-of-action generation as an optimization routine.

Forecasting Engine All 21 steps →

Plan before you act. Contain speculation. Promote only what passes.

Primary Technical Disclosure
○ Forecasting and Executive Graphs in Autonomous Cognitive Systems
Secondary Technical
○ Planning Graphs as First-Class Cognitive Structures○ Containment Layer and Delusion Boundary○ Branch Classification System○ Personality Field as Structural Modifier○ Executive Engine Multi-Agent Graph Aggregation○ Branch Dormancy and Deferred Promotion○ Proactive Speculative Maintenance (Dream State)○ Planning Graph Archival for Cognitive Forensics○ Cross-Agent Planning Graph Visibility○ Slope-Constrained Speculative Simulation○ Structural Separation From Verified Memory○ Forecasting Engine Architecture○ Forecasting Execution Cycle○ Emotional Modulation of Planning○ Executive Graph Conflict Resolution○ Planning Graph Delegation and Forking○ Temporal Anchoring and Lifecycle Management○ Forecasting as Coordination Primitive○ Forecasting-Modulated Discovery Traversal○ Forecasting as Confidence Input○ Integrity-Constrained Forecasting○ Forecasting for Training Curriculum○ Biological Signal to Forecasting Coupling○ Substrate-Agnostic Forecasting Deployment
Applications (General)
○ Surgical Robot Planning Through Governed Speculative Branches○ Defense Tactical Planning With Contained Speculation○ Forecasting Engine for Logistics Planning○ Forecasting Engine for Disaster Response Planning○ Forecasting Engine for Financial Portfolio Planning○ Forecasting Engine for Construction Project Planning○ Forecasting Engine for Epidemic Response Planning○ Forecasting Engine for Space Mission Planning
Applications (Specific)

○ da Vinci Plans Trajectories, Not Consequences● Anduril's Lattice Plans Missions Without Speculative Containment○ Boston Dynamics Plans Motion, Not Missions○ Shield AI's Hivemind Cannot Contain Its Own Speculation○ MuJoCo Simulates Physics Without Planning Governance○ NVIDIA Isaac Sim Renders Worlds Without Governing Plans○ Unity ML-Agents Trains Without Governing Speculation○ Gazebo Simulates Robots Without Governing Their Plans○ Drake Optimizes Trajectories Without Governing Planning Structures○ robosuite Benchmarks Manipulation Without Governing Plans
Forecasting Engine overview →
AQ
deterministic
autonomy

Legal

Last updated: 2026-03-03

-
- Inventive Steps
- Licensing
- Patents
- Articles
- Legal
- Opportunities
- Sitemap

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

Invented by Nick Clark | Founding Investors: Devin Wilkie