NEUROGRAPH FOUNDATION

Product Requirements Document

Adaptive Neuro-Symbolic Cognitive Architecture

Version 1.0 • February 2026 • CONFIDENTIAL

1. Executive Summary

1.1 Vision

The NeuroGraph Foundation is a reusable, project-agnostic cognitive architecture that provides any AI system built on top of it with dynamic learning, causal reasoning, and self-optimizing knowledge management. It replaces the static retrieval paradigm (query → fetch → respond) with a living knowledge substrate that strengthens useful pathways, prunes irrelevant ones, discovers higher-order relationships, and predicts future states.

1.2 Problem Statement

Current AI architectures suffer from three fundamental limitations. First, **static knowledge**: vector databases retrieve information but never learn which retrievals were useful, leading to identical performance on day one and day one thousand. Second, **flat relationships**: pairwise similarity scores cannot represent complex n-ary relationships such as syndromes, legal precedent chains, or multi-factor security threat signatures. Third, **no temporal reasoning**: existing systems treat every query as an isolated event with no concept of sequence, causation, or prediction.

1.3 Solution Architecture

NeuroGraph integrates three complementary subsystems into a unified foundation layer. The **Semantic Memory Layer** (Vector Database) handles content storage and fuzzy similarity retrieval. The **Structural Memory Layer** (Hypergraph Engine) represents n-ary relationships, concept clusters, and hierarchical abstractions as first-class objects. The **Temporal Dynamics Layer** (Spiking Neural Network with STDP) provides real-time learning, causal inference, predictive coding, and homeostatic self-regulation. These layers are deeply interleaved, with each layer reading from and writing to the others continuously.

1.4 Target Applications

Application	Foundation Capabilities Leveraged
DSM-style Diagnostic Reasoning	Hyperedge syndrome detection, pattern completion from partial symptom sets, causal chain inference for differential diagnosis, temporal symptom progression modeling
Autonomous Vibe-Coding Agent	Self-improving retrieval (learns which patterns resolve issues), causal debugging (error A → failure B), pruning of outdated API knowledge, strengthening of project-specific conventions
AI Consciousness Emergence	Full dynamic plasticity as substrate for emergent behavior, predictive coding for self-modeling, hypergraph abstraction for concept formation, homeostatic regulation against runaway cascades

2. System Architecture

2.1 Design Principles

Sparse by default. No dense matrices. All structures use sparse representations (adjacency lists, hash maps, compressed sparse row) to support millions of nodes without memory explosion. **Dynamic topology.** Nodes and edges are created and destroyed at runtime with no fixed graph size. **Pluggable plasticity.** Learning rules (STDP, Hebbian, BCM, custom) are strategy objects that can be swapped, combined, or layered per-project. **Persistence-native.** All state is serializable and checkpointable. The graph can be saved, loaded, forked, and merged. **Vector DB agnostic.** The foundation interfaces with any vector database through an adapter pattern.

2.2 Core Abstractions

2.2.1 Node

A Node is the atomic unit of the graph. Each wraps a reference to a vector database entry and carries its own neural state. Nodes are stateful computational units, not mere pointers.

Property	Type	Description
node_id	UUID	Globally unique identifier, matches vector DB entry ID
voltage	float	Current membrane potential. Accumulates input; resets on spike
threshold	float	Adaptive firing threshold. Adjusts via intrinsic plasticity
resting_potential	float	Baseline voltage after reset (default 0.0, can be negative for inhibition)
refractory_remaining	int	Timesteps remaining in refractory period. Cannot fire while > 0
refractory_period	int	Duration of refractory period in timesteps (default 2)
last_spike_time	float	Timestamp of most recent spike. Used by STDP calculations
spike_history	RingBuffer	Rolling window of recent spike times (default depth 100). Enables firing rate and burst detection
firing_rate_ema	float	Exponential moving average of firing rate. Used by homeostatic plasticity
intrinsic_excitability	float	Multiplier on incoming current. Adjusted by homeostatic rules (default 1.0)
metadata	Dict	Application-specific key-value data (type, creation time, domain tags)
is_inhibitory	bool	If true, outgoing spikes subtract from target voltage

2.2.2 Synapse

A Synapse is a directed, weighted connection between two nodes. First-class objects with their own state for fine-grained plasticity tracking.

Property	Type	Description
synapse_id	UUID	Unique identifier
pre_node_id	UUID	Source node (the cause in causal links)

post_node_id	UUID	Target node (the effect in causal links)
weight	float	Connection strength [0.0, max_weight]. Initialized near 0; shaped by plasticity
max_weight	float	Upper bound preventing runaway potentiation (default 5.0)
delay	int	Propagation delay in timesteps. Models axonal delay (default 1)
last_update_time	float	Timestamp of most recent plasticity update
eligibility_trace	float	Decaying trace for three-factor learning (reward-modulated STDP)
creation_time	float	When synapse was created. Used for age-based pruning
type	Enum	EXCITATORY, INHIBITORY, MODULATORY

2.2.3 Hyperedge

A Hyperedge connects an arbitrary number of nodes as a set-valued relationship. Represents composite concepts: syndromes, threat signatures, code patterns. First-class computational objects with activation dynamics.

Property	Type	Description
hyperedge_id	UUID	Unique identifier
member_nodes	Set[UUID]	Node IDs in this relationship. Unordered unless type is SEQUENCE
member_weights	Dict[UUID,float]	Per-member importance. Not all contribute equally to activation
activation_threshold	float	Weighted fraction of active members needed to fire (default 0.6)
activation_mode	Enum	WEIGHTED_THRESHOLD (default), K_OF_N, ALL_OR_NONE, GRADED
current_activation	float	Current level [0.0, 1.0]. Decays over time
output_targets	List[UUID]	Nodes receiving input when hyperedge fires
output_weight	float	Signal strength sent to output targets on activation
metadata	Dict	Application data (label, domain, creation_mode: MANUAL/DISCOVERED/MERGED)
is_learnable	bool	Whether plasticity can modify weights and threshold (default true)

2.2.4 Graph Container

The Graph manages all nodes, synapses, and hyperedges, orchestrates simulation, applies plasticity, and interfaces with external systems.

Responsibility	Details
Topology Management	Add/remove nodes, synapses, hyperedges at runtime. Sparse adjacency indices. Cascading deletions.
Simulation Loop	step(): decay voltages, inject currents, propagate spikes (with delays), evaluate hyperedges, apply plasticity, enforce refractory periods, record telemetry.
Plasticity Orchestration	Registry of PlasticityRule objects. After each step, apply eligible rules. Rules can be global or scoped to subgraphs.

Homeostatic Regulation	Periodic normalization: adjust excitability, scale weights, prune dead synapses, consolidate redundant hyperedges.
Persistence	Full, incremental, and fork checkpoints. JSON and MessagePack serialization.
Telemetry	Global firing rate, weight distributions, hyperedge activation frequency, pruning/sprouting counts.
Vector DB Bridge	Bidirectional: search results inject node currents; strong activations surface as associative retrieval recommendations.

3. Plasticity System

Plasticity is the core differentiator. The system ships with composable plasticity rules and an extension interface for custom rules.

3.1 STDP (Spike-Timing-Dependent Plasticity)

STDP is the primary learning rule. It captures **causality** by examining the precise temporal relationship between pre- and post-synaptic spikes.

3.1.1 Mathematical Specification

For a synapse from node A (pre) to node B (post), when B fires at time t_post:

Long-Term Potentiation (LTP): When A fires before B (Δt = t_post − t_pre > 0):

```
Δw = A_plus × exp(−Δt / τ_plus) × learning_rate
```

Long-Term Depression (LTD): When A fires after B (Δt < 0):

```
Δw = −A_minus × exp(Δt / τ_minus) × learning_rate
```

Critical: A_minus must be slightly larger than A_plus (ratio 1.05–1.2). This creates a natural bias toward weakening, essential for stability. Without this, the network drifts toward saturation.

Parameter	Default	Range	Effect of Tuning
τ_plus (causal window)	20 steps	5–100	Smaller = stricter causality. Larger = more permissive causal attribution
τ_minus (acausal window)	20 steps	5–100	Usually matched to τ_plus. Can be asymmetric for domain-specific needs
A_plus (LTP magnitude)	1.0	0.1–5.0	Higher = faster causal learning. Risk: instability
A_minus (LTD magnitude)	1.2	0.1–5.0	Higher = more aggressive pruning. Must be ≥ A_plus
learning_rate	0.01	0.001–0.1	Lower = gradual, stable. Higher = fast, volatile
max_weight	5.0	1.0–50.0	Hard ceiling against explosion

3.1.2 Failure Modes and Mitigations

Failure Mode	Mechanism	Mitigation
Runaway Potentiation	Frequently used chains reinforce to max_weight, dominating activity	Weight-dependent STDP: Δw scaled by (max_weight − w)/max_weight for LTP. Strong connections learn slower (soft saturation)
Silent Death	Never-firing nodes have weights perpetually weakened until unreachable	Intrinsic excitability: if firing rate < min, increase excitability multiplier to lower effective threshold
Temporal Aliasing	Same-timestep events have Δt=0, falling into neither LTP nor LTD	Treat Δt=0 as weak LTP (half strength) or use sub-step ordering
Causal Reversal	Correlative relationship encoded as causal due to coincidental timing	Three-factor learning: STDP creates eligibility trace, weight change only commits on reward signal

3.2 Homeostatic Plasticity

Maintains global stability without destroying learned structure. Operates on a slower timescale than STDP.

3.2.1 Mechanisms

Mechanism	Adjusts	How	When
Synaptic Scaling	Incoming weights	Multiplicative: w_new = w_old × (target_rate/actual_rate)^factor. Preserves relative ratios.	Every N steps (default 100) when firing_rate deviates from target
Intrinsic Excitability	Excitability multiplier	Rate too low → increase. Rate too high → decrease. Slow EMA prevents oscillation.	Same schedule. Complements scaling (input vs. sensitivity)
Threshold Adaptation	Firing threshold	Drifts toward recent avg voltage. Prevents perpetually sub/super-threshold nodes.	Continuous, rate 0.001/step
Refractory	Ability to fire	Mandatory rest after firing. Cannot fire regardless of input.	Automatic, per-node (default 2 steps)

Why not normalize weights? Gemini's approach divides all incoming weights by their sum. This is **destructive** — it collapses absolute magnitude info. One weight of 4.0 looks identical to 100 weights of 0.04. Multiplicative synaptic scaling preserves learned distributions while adjusting gain.

3.3 Structural Plasticity

3.3.1 Pruning Rules

Rule	Trigger	Action
Weight-based	weight < 0.01 for > 500 steps	Remove synapse. Flag isolated nodes for GC.
Age-based	Exists > max_age AND weight never exceeded 2× initial	Remove. Prevents lingering speculative connections.
Activity-based	Unused for > 1000 steps	Remove regardless of weight.
Hyperedge consolidation	>80% member overlap between two hyperedges	Merge. Average weights. Keep specific metadata.

3.3.2 Sprouting Rules

Rule	Trigger	Action
Co-activation	Two nodes fire within window, no synapse exists	Create at weight 0.1. Must earn strength via STDP.
Hyperedge discovery	N≥3 nodes consistently co-activate, no covering hyperedge	Create with conservative threshold (0.8).
Bridge sprouting	Node bridges disconnected clusters A and B	Speculative synapses between high-activity nodes. Short-lived unless validated.

4. Hypergraph Engine

4.1 Why Pairwise Links Are Insufficient

Major Depressive Episode requires concurrent presence of 5+ symptoms from a set. This is a **set-valued** relationship. Pairwise links lose the co-occurrence constraint. Only the hyperedge correctly captures: {Depressed Mood, Insomnia, Weight Loss, Fatigue, Guilt} → Major Depressive Episode.

4.2 Activation Dynamics

Activation (WEIGHTED_THRESHOLD):

```
activation = Σ(member_weight_i × is_active_i) / Σ(member_weight_i)
if activation ≥ threshold: hyperedge fires → signal to output_targets
```

Enables **pattern completion**: 4 of 5 symptoms active → hyperedge fires → activates 5th or diagnosis node.

4.3 Hyperedge Plasticity

Adaptation	Mechanism	Effect
Member weights	Inactive members during firing get weight decreased; consistently active members increased	Learns core vs. peripheral members. Handles optional diagnostic criteria.
Threshold	Rewarded firing → lower threshold; punished → raise	Learns strictness: tight clusters (all needed) vs. loose (subset suffices).
Member evolution	Non-member consistently co-fires beyond discovery window → added with low weight	Expands to newly discovered concepts.

4.4 Hierarchical Hyperedges

Hyperedges can contain other hyperedges, creating multi-level abstraction. **Level 0:** Symptoms. **Level 1:** Syndrome clusters. **Level 2:** Diagnostic categories. This hierarchy can **emerge** through discovery, not just be pre-programmed.

5. Predictive Coding Engine

STDP's most powerful emergent property is predictive coding. Once causal sequences are learned, the system transitions from reactive to predictive.

5.1 Prediction Mechanism

When node A fires with a strong causal link to B, spike propagation partially pre-charges B's voltage — a **prediction** that B will fire soon.

Prediction confirmed (B fires): Weight maintained/strengthened. Causal model validated.

Prediction error (B silent): The critical learning signal. Generates **surprise signal** proportional to prediction strength. Effects: (1) A→B weight weakened. (2) Surprise broadcasts to neighbors, triggering exploration — "what happened instead of B?" Drives discovery of alternative pathways and exceptions.

5.2 Prediction Events

Event	Payload	Application Use
PredictionConfirmed	pre_node, post_node, strength, delay	Confidence tracking. 100 consecutive → treat as hard rule.
PredictionError	pre, expected_post, strength, actual_posts	Most valuable event. Model refinement, exception discovery, anomaly detection.
NovelSequence	Firing sequence with no matching pattern	New causal chain. Triggers exploratory sprouting. Security: new attack vector.

6. Persistence and State Management

6.1 Serialization

Primary format: MessagePack (binary, compact). JSON as human-readable alternative.

Component	Contents	Size/Unit
Node State	All §2.2.1 properties. Spike history truncated to ring buffer depth.	~200 B/node
Synapse State	All §2.2.2 properties.	~120 B/synapse
Hyperedge State	All §2.2.3 properties.	~300B + 16B/member
Global State	Timestep, plasticity configs, homeostatic params, RNG seed.	~2KB fixed
Adjacency Indices	Sparse outgoing/incoming maps. Cached for performance.	~32 B/synapse

6.2 Checkpointing

Full: Entire graph. At startup, explicit save, and every N steps (default 1000). **Incremental:** Only modified objects (dirty-flag tracking). Dramatically reduces I/O for large graphs. **Fork:** Named branch for experimentation. Explore speculative paths; discard if degraded; restore main branch.

6.3 Start Modes

Cold start: Empty graph. Nodes created as vector DB is populated. **Warm start:** Checkpoint loaded, all learned state restored. **Transfer learning:** Cross-project checkpoint loading via transfer() method — loads topology/weights, clears app-specific metadata.

7. Vector Database Integration

7.1 Adapter Interface

NeuroGraph defines an adapter interface any vector DB can implement:

```
search(query_vector, top_k, filters) → List[(id, score, metadata)]
get(id) → (vector, metadata)
upsert(id, vector, metadata) → void
delete(id) → void
```

7.2 Bidirectional Flow

Direction	Mechanism	Effect
VectorDB → Graph	Search results inject input current proportional to similarity score.	Relevant nodes activate. STDP learns which retrievals were actually useful.
Graph → VectorDB (Associative)	Strongly activated non-query nodes surfaced as recommendations.	System says "Y is always relevant when X comes up" — structural intelligence beyond similarity.
Graph → VectorDB (Enrichment)	Learned relationships written back to vector metadata.	Vector DB benefits from graph learning even when neural layer is bypassed.

8. Public API Surface

Application code should never manipulate internals directly. All interactions go through the Graph container.

Method	Signature	Description
create_node	(id, metadata) → Node	Register node. Called when new vector added to DB.
remove_node	(id) → void	Remove node + connected synapses. Update hyperedges.
stimulate	(id, current) → void	Inject input. Primary pathway from vector search.
stimulate_batch	(List[(id, current)]) → void	Batch stimulus for search results.
step	() → StepResult	Advance one timestep. Returns fired nodes, hyperedges, predictions.
step_n	(n) → List[StepResult]	Run n steps. Allows propagation to settle.
get_active_nodes	(threshold) → List[(id, v)]	Nodes above voltage threshold. For associative retrieval.
get_predictions	() → List[Prediction]	Current predictions (pre-charged unfired nodes).
get_causal_chain	(id, depth) → DAG	Trace learned causality forward/backward.
get_hyperedges	(id) → List[Hyperedge]	Hyperedges containing this node.
checkpoint	(path, mode) → void	Save state. FULL / INCREMENTAL / FORK.
restore	(path) → void	Load state from checkpoint.
set_plasticity_rules	(List[Rule]) → void	Configure active plasticity rules.
register_event_handler	(type, cb) → void	Subscribe to events (spikes, predictions, pruning).
get_telemetry	() → Telemetry	Network statistics.
inject_reward	(strength) → void	Broadcast reward for three-factor learning.

9. Default Configuration and Tuning Guide

Category	Parameter	Default	Tuning Notes
SNN	decay_rate	0.95	Lower (0.90) = responsive. Higher (0.99) = longer memory, risk accumulation
SNN	default_threshold	1.0	Higher = selective. Lower = sensitive, noisy
SNN	refractory_period	2 steps	Higher = prevents rapid re-firing
STDP	τ_plus / τ_minus	20 / 20	"Within 20 interactions." Tighten for strict causality
STDP	A_plus / A_minus	1.0 / 1.2	1.2 ratio ensures sparsity trend
STDP	learning_rate	0.01	Too slow → 0.05. Unstable → 0.005
Homeostasis	target_firing_rate	0.05	5% of timesteps
Homeostasis	scaling_interval	100 steps	Too frequent = oscillation
Pruning	weight_threshold	0.01	Below this + grace → pruned
Pruning	grace_period	500 steps	Time for new synapses to prove value
Pruning	inactivity_threshold	1000 steps	Unused synapses removed
Sprouting	co_activation_window	5 steps	Closeness needed for speculative creation
Sprouting	initial_weight	0.1	Must earn strength through STDP
Hyperedge	activation_threshold	0.6	60% weighted members must be active
Hyperedge	discovery_threshold	0.8	High to prevent noise-driven proliferation
Persistence	checkpoint_interval	1000 steps	Adjust based on step rate

10. Implementation Roadmap

10.1 Phase 1: Core Engine (v0.1)

Deliverables: Node, Synapse, Graph with sparse adjacency. SNN loop (decay, spike, propagate). Full STDP (LTP + LTD). Weight clamping. JSON serialization. Step-based clock. Unit tests.

Acceptance: 1K-node graph runs 10K steps without explosion or silent death. STDP correctly strengthens/weakens based on timing.

10.2 Phase 2: Stability (v0.2)

Deliverables: Homeostatic plasticity. Refractory periods. Structural plasticity (prune/sprout). Weight-dependent STDP. Telemetry. Incremental checkpointing.

Acceptance: 100K steps without intervention. Firing rates within 2× target. ≥30% speculative synapses pruned. No memory leaks.

10.3 Phase 3: Hypergraph (v0.3)

Deliverables: Hyperedge with all activation modes. Hyperedge plasticity. Discovery. Hierarchy. Consolidation. Pattern completion.

Acceptance: 4-of-5 syndrome test fires hyperedge and activates 5th. Auto-discovered hyperedges match ≥80% of manual definitions.

10.4 Phase 4: Prediction (v0.4)

Deliverables: Prediction tracking. Error events. Surprise exploration. Three-factor learning. Event API.

Acceptance: After 50× A→B→C training, stimulating A predicts B then C. New branch A→B→D learned within 10 exposures.

10.5 Phase 5: Integration (v1.0)

Deliverables: Vector DB adapters (Chroma, Qdrant, pgvector). Bidirectional flow. Metadata enrichment. Transfer learning. Fork/merge. Documentation.

Acceptance: Working SNN-enhanced retrieval in <50 lines of integration code. Cross-project checkpoint loading.

11. Risks and Open Questions

Risk	Severity	Mitigation
Performance: STDP expensive above 100K synapses	High	Vectorized batch STDP. Sparse update (only fired synapses). GPU for >1M.
Hyperedge explosion from noisy data	Medium	High discovery threshold (0.8). Persistence requirement. Consolidation merges.
Clock sync: real-time vs step-based incompatibility	Medium	Normalize to abstract ticks. Clock adapters. τ values in ticks, not ms.
Vibe-coding inconsistency	High	This PRD + single importable module + documented API (§8).
Consciousness: unspecified capabilities needed	Low	Extensible: attention = gain modulation, working memory = recurrent loops, metacognition = meta-graph. Phase 6+.

12. Glossary

Term	Definition
STDP	Spike-Timing-Dependent Plasticity. Adjusts weights based on temporal ordering of pre/post spikes.
LTP	Long-Term Potentiation. Synaptic strengthening when pre fires before post.
LTD	Long-Term Depression. Synaptic weakening when pre fires after post.
Hyperedge	Generalized edge connecting a set of nodes. Represents n-ary relationships.
Homeostatic Plasticity	Stability mechanisms: firing rate, threshold, and synaptic scaling adjustments.
Eligibility Trace	Decaying record bridging stimulus and delayed reward in three-factor learning.
Predictive Coding	Pre-activation of expected nodes from learned causal sequences. Errors drive learning.
Sprouting	Creation of new synapses between co-activating nodes lacking direct connection.
Pruning	Removal of weak/inactive synapses to maintain sparsity.
Synaptic Scaling	Multiplicative weight adjustment preserving relative ratios while targeting firing rate.
Three-Factor Learning	STDP + reward modulation. Changes commit only on reward confirmation.
Associative Retrieval	Graph-driven surfacing of related concepts beyond vector search results.