

FUTON1

@multiarg futon1/devmap
@title FUTON1 Development Map
@audience futon-devs, infra-agents, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Keen investigation (dhammavicaya)
@IFR: FUTON1 is the dhammavicaya layer: a deterministic, well-documented stack where chat demos, NLP, graph memory, and XTDB stay locked in sync so every focus header, ingest script, and CLI transcript has a reproducible twin for higher futons—turning the act of building the fact graph into keen investigation.

@state Operational stack shipping focus-header demos; documentation/tests drift around NLP interface and XT instrumentation still unresolved.

@next Refresh focus-header scripts, consolidate nlp-interface documentation/tests, and publish the XT operations stories promised by Prototype 4.

! instantiated-by: Prototype 0 — Baseline Stack Snapshot [🔧/已 📄/布]
+ context: futon1 is a collection of deterministic chat demos built on a tiered NLP stack, in-memory graph-memory, and XTDB persistence, surfaced via 'apps/basic-chat-demo ', 'apps/nlp-interface ', 'apps/graph-memory ', and 'apps/open-world-ingest '.

+ if: You want a clear picture of what already works before you add new affordances.

+ however: The current documentation is spread across multiple READMEs and doesn't yet tell a single "stack story" from CLI to XTDB.

+ then: Freeze the current behaviour with tags, record a concise architecture note (pipeline → graph → XT), and capture representative CLI transcripts for v1-v5.

+ because: This establishes a stable reference snapshot and makes regressions and future extensions legible.

! instantiated-by: Prototype 1 — basic-chat/v5 Hardening (Focus Header Path) [🔧/每 📄/已]
+ context: the 'demo ' app is the canonical way to interrogate the stack; it emits focus headers, records hydration, and demonstrates how NLP, graph-memory, and XTDB cohere in practice.

+ if: You want the CLI to behave like calibrated instrumentation instead of a friendly-but-shaky demo.

+ however: our current smoke tests hit only the happy path, leaving hydration races, RocksDB locks, and legacy protocol quirks to intuition.

+ then: design an automated focus-header lab: scripted sessions that flip every flag, replay historical transcripts, simulate flaky transports, and publish the resulting traces alongside expected XTDB states.

+ because: treating 'basic-chat ' as calibrated instrumentation (not a toy) keeps every downstream agent honest about what the transport and memory layers actually guarantee.

! instantiated-by: Prototype 2 — NLP Interface Consolidation [🔧/申 📄/这些]
+ context: 'apps/nlp-interface ' provides the deterministic NER/POS and intent layers shared across protocols, including v4 gazetteer and pattern recogniser hooks.

+ if: You want a single, testable entry point for the NLP stack that can evolve independently of the chat CLI.

+ however: The nlp-interface README is still boilerplate and doesn't advertise the actual pipeline or RID-style intent dictionary.

+ then: Refactor nlp-interface into explicit stages (tokenise → tag → chunk → intent), add unit tests for each, and write a focused README for these APIs.

+ because: A consolidated NLP module makes it easier to swap or extend components without destabilising the demos.

! instantiated-by: Prototype 3 — Graph-Memory API & Seed Graph [🔧/已 📄/么]
+ context: 'graph-memory ' already encodes the Datascript schema, salience metadata, and helper fns that everything else relies on—but only insiders know how to call it.

+ if: You want futon1 to expose a principled contract so other futons (and external agents) can trust the graph.

+ however: helper fns are tuned for REPL spelunking, naming is inconsistent, and there is no formal description of what gets mirrored into XTDB.

+ then: promote 'graph-memory ' to a documented API: name the seed graph, publish init/upsert/link/export specs, generate API docs straight from tests, and treat breaking changes like migrations.

+ because: a graph-memory contract—rather than comfy

FUTON2

@multiarg futon2/devmap
@title FUTON2 Development Map
@audience futon-devs, simulation-agents, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Energy (viriya)
@IFR: FUTON2 is the viriya (energy) chamber: a narratable Active Inference engine whose ants are first demonstrators for AIF agent species that will eventually crawl FUTON1 graph memory, export state to higher futons, and turn insight into kinetic practice across the network.

@state Ant sim + AIF loop run end-to-end with analyzer hooks, but architecture docs, observation property tests, and hunger/tau goldens remain incomplete.

@next Capture the baseline architecture note, lock observation/property tests, and record hunger/precision goldens before layering new agent species.

! instantiated-by: Prototype 0 — AIF Stack Baseline (observe → perceive → affect → core) [🔧/卯 🔧/已]
+ context: FUTON2 runs a full Active Inference loop in the ant war-game, wiring 'ants.aif.observe ', 'ants.aif.perceive ', 'ants.aif.affect ', and 'ants.aif.core ' into one tick-by-tick pipeline.

+ if: You want a stable story that shows how observation, prediction errors, precision, and actions integrate per tick.

+ however: Modules are individually solid but no single document freezes hunger/tau semantics or traces the whole stack.

+ then: Capture a top-level architecture note from sensory keys through world updates, lock current hunger and predictive micro-step parameters, and add four golden AIF microtraces.

+ because: A documented baseline prevents accidental drift before you optimise policy, analyzers, or mechanics.

! instantiated-by: Prototype 1 — Observation Layer Hardening [🔧/则 📄/下]
+ context: 'g-observe ' normalises sensory evidence (food, pheromones, proximity, novelty, reserves) into the vectors passed downstream.

+ if: You want deterministic observations across seeds and a nailed-down vector ABI for later ML.

+ however: Normalisation bounds and neighbour logic assumptions are scattered across helpers.

+ then: Write property tests that enforce 0–1 ranges, document the 'sense→vector ' ordering, and add edge-case tests for grid borders, max-distance tweaks, and degenerate pheromone fields.

+ because: Reliable observations keep perception and policy stable no matter the world configuration.

! instantiated-by: Prototype 2 — Predictive Coding Microcycle Clarity [🔧/刊]
+ context: Perception performs multi-step updates of mu, tau, Pi-0, goals, and weighted errors with annealing.

+ if: You want inspectors (or future Arxana bridges) to understand belief updates.

+ however: Hunger updates, sensory prediction, and error accumulation are interwoven without a canonical trace.

+ then: Publish a single-tick perception trace with plots of hunger, tau, and error per micro-step, and formalise an interface for extracting a mu/precision snapshot.

+ because: Clear cycles make later integrations with futon4 symbolic memory far easier.

! instantiated-by: Prototype 3 — Hunger & Precision Dynamics Goldens [🔧/萬 📄/六]
+ context: Functions such as 'tick-hunger ', 'update-hunger ', 'update-tau ', and 'modulate-precisions ' control the most sensitive dynamics.

+ if: You want reproducible affective behaviour that survives refactors.

+ however: There are no golden tests for tau evolution, hunger saturation, or reserve-driven adjustments.

+ then: Record six golden sequences (normal, starvation, overstimulation, heavy-cargo, high-risk, high-ingest) and lock expected tau floors/caps for each.

+ because: Once goldens exist, small tune-ups stop breaking entire colony behaviour.

! instantiated-by: Prototype 4 — Policy Layer Surfacing [🔧/十 📄/以]
+ context: 'ants.aif.policy ' selects actions using EFE metrics while 'attach-policy-diagnostics ' emits traces.

+ if: You want reproducible policy choices under

FUTON3

@multiarg futon3/devmap
@title FUTON3 Development Map
@audience futon-devs, sandbox-agents, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Joy / Rapture (pīti)
@IFR: FUTON3 is the flexiformal pīti chamber where design patterns live as checkable claims, embeddings surface near fits, and daily actions are certified against devmap obligations with joyful (Spinozan) empowered action.

@state MUSN transport/REPL/sigil tooling are Live, while the canonical pattern store, check DSL, proof adapters, and workday instrumentation remain underway.

@next Stand up the pattern canon + 'check; API, wire proof→graph/energy exports, and land the workday instrumentation loop before expanding training-ground efforts.

! instantiated-by: Prototype 0 — Flexiformal Transport Baseline [🔧/已]
+ context: FUTON3 still rides the MUSN transport (WS+HTTP bus, SAFE/ADMIN REPL, NDJSON ingest), but its primary job is to host pattern checks and informal proof states, not free-roaming agents.

+ if: You want an execution substrate where pattern obligations can be posed, evaluated, and recorded with deterministic transcripts.

+ then: Establish baseline artefacts (README-manifesto, clock interface, Arxana reboot outline).

+ because: This creates an anchor for all subsequent prototypes.

! instantiated-by: Prototype 1 — Arxana Reboot (Week-long target) [🔧/新 📄/长]
+ context: Legacy literate-programming structure exists but is dormant.

+ if: You want stable parsing, node navigation, and save-back-to-LaTeX working.

+ however: Legacy Emacs-Lisp and refactored Clojure schemas diverge.

+ then: Reboot spine parser, restore node browsing, define new nema-based schema, test round-trip edits.

+ because: A functioning editor is the precondition for all downstream memory work.

+ next (workflow plumbing):

- Create import/export shims for modern doc formats.
- Produce migration notes for historical datasets.

+ next (QA & release prep):

- Grow the ERT suite (unit, integration, CLI smoke).
- Document test instructions and CI hooks.
- Provide contributor guide (coding style, tangling, testing).
- Tag a “revived” release with reproducible steps.

! instantiated-by: Prototype 2 — Graph Memory (Datascript) [🔧/已 🔧/生活]
+ context: The hyperedge schema for Arxana exists on paper but not in running code.

+ if: You want argument fragments to become addressable nodes with provenance.

+ however: there is no minimal memory manager wiring Arxana buffers to Datascript.

+ then: implement 'add-node ', 'activate-node ', and 'link-node '; persist them via Datascript; replay a session to prove round-trips.

+ because: without a living graph memory, Arxana never escapes editor cosplay.

! instantiated-by: Prototype 3 — Flexiarg ⇔ Arxana Integration [🔧/双]
+ context: Flexiarg is acting as a pidgin for arguments and patterns.

+ if: You want arguments to compile into graph nodes and be editable via Arxana.

+ however: No translation layer exists between pidgin text and structured memory entries.

+ then: Define schema mapping: argument-block → rule node (Q/A pair).

+ because: This yields bidirectional editing: human-readable ↔ machine-structured.

! instantiated-by: Prototype 4 — Agent Layer Bridge (FUTON3) [🔧/不能 📄/但]
+ context: Agents can “speak” flexiarg but cannot yet access memory graphs.

+ if: You want pattern-aware agents who can detect tension, link motifs, and apply PIDGIN→PROSE.

+ however: No API exists to let a GPT-based agent query the graph.

+ then: Create REPL-level bridge (datascript queries + JSON-rpc + Emacs functions).

+ because: This is what upgrades GPT into a pattern-capable agent.

! instantiated-by: Prototype 5 — Multi-user Memory (MUSN revival) [🔧/在 📄/但]
+ context: MUSN existed as a concept for multi-user

FUTON4

@multiarg futon4/devmap
@title FUTON4 Development Map
@audience futon-devs, pattern-agents, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Tranquility (passadhi)
@IFR: FUTON4 becomes the passadhi (tranquility) memory atelier where Arxana, graph memory, flexiarg, and agents interoperate, letting symbolic artefacts be edited, linked, and narrated in one continuous workspace that settles and integrates the energetic work beneath it.

@state Arxana reanimation through M4 is working (storage bridge, browsing, relations) while inclusion/transclusion UX, modern import/export shims, and QA/release prep remain open.

@next Deliver the pending inclusion UX, document modern import/export/migration shims, and grow the QA + contributor guide so a “revived” release can be tagged.

! instantiated-by: Prototype 0 (Running Since 29 July 2025) [🔧/从 📄/甲]
+ context: Prototype 0 establishes the symbolic foundations, artefact inventory, and DREAM logic.

+ if: You want a coherent first cycle of a year-long 12-prototype run.

+ however: Symbolic activation does not yet translate into memory operations or agent affordances.

+ then: Establish baseline artefacts (README-manifesto, clock interface, Arxana reboot outline).

+ because: This creates an anchor for all subsequent prototypes.

! instantiated-by: Prototype 1 — Arxana Reboot (Week-long target) [🔧/新 📄/长]
+ context: Legacy literate-programming structure exists but is dormant.

+ if: You want stable parsing, node navigation, and save-back-to-LaTeX working.

+ however: Legacy Emacs-Lisp and refactored Clojure schemas diverge.

+ then: Reboot spine parser, restore node browsing, define new nema-based schema, test round-trip edits.

+ because: A functioning editor is the precondition for all downstream memory work.

+ next (workflow plumbing):

- Create import/export shims for modern doc formats.
- Produce migration notes for historical datasets.

+ next (QA & release prep):

- Grow the ERT suite (unit, integration, CLI smoke).
- Document test instructions and CI hooks.
- Provide contributor guide (coding style, tangling, testing).
- Tag a “revived” release with reproducible steps.

! instantiated-by: Prototype 2 — Graph Memory (Datascript) [🔧/已 🔧/生活]
+ context: The hyperedge schema for Arxana exists on paper but not in running code.

+ if: You want argument fragments to become addressable nodes with provenance.

+ however: there is no minimal memory manager wiring Arxana buffers to Datascript.

+ then: implement 'add-node ', 'activate-node ', and 'link-node '; persist them via Datascript; replay a session to prove round-trips.

+ because: without a living graph memory, Arxana never escapes editor cosplay.

! instantiated-by: Prototype 3 — Flexiarg ⇔ Arxana Integration [🔧/双]
+ context: Flexiarg is acting as a pidgin for arguments and patterns.

+ if: You want arguments to compile into graph nodes and be editable via Arxana.

+ however: No translation layer exists between pidgin text and structured memory entries.

+ then: Define schema mapping: argument-block → rule node (Q/A pair).

+ because: This yields bidirectional editing: human-readable ↔ machine-structured.

! instantiated-by: Prototype 4 — Agent Layer Bridge (FUTON3) [🔧/不能 📄/但]
+ context: Agents can “speak” flexiarg but cannot yet access memory graphs.

+ if: You want pattern-aware agents who can detect tension, link motifs, and apply PIDGIN→PROSE.

+ however: No API exists to let a GPT-based agent query the graph.

+ then: Create REPL-level bridge (datascript queries + JSON-rpc + Emacs functions).

+ because: This is what upgrades GPT into a pattern-capable agent.

! instantiated-by: Prototype 5 — Multi-user Memory (MUSN revival) [🔧/在 📄/但]
+ context: MUSN existed as a concept for multi-user

FUTON5

@multiarg futon5/devmap
@title FUTON5 Development Map — Concentration, Nonstarter, Patterns of Improvisation
@audience futon-devs, improviser-agents, CT-metaweavers, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Concentration (samādhi)
@IFR: FUTON5 operates as the samādhi (concentration) engine, turning improvisation, CT riffs, musical chops, and Nonstarter economics into renewable practices that keep creativity gathered without teleological drag.

@state Design-only plan; prototypes have narrative intent but no running code or artefacts yet.

@next start

! instantiated-by: Prototype 0 — Nonstarter Frame (Concentration without Teleology) [🔧/乃 📄/不能]
+ context: FUTON5 is the samādhi layer: gathering energy around chosen patterns, sustaining focus, and refusing the teleology of success so value is created without capture and creativity concentrates without collapse.

+ if: You want a layer that stabilises creative practice (music, coding, argument, CT diagrams) into repeatable, renewable flows.

+ however: Current prototypes (MetaCA, flexiarg, musical studies, CT notes) sit in isolation and do not yet form a concentrated practice ecology.

+ then: Define FUTON5 as the layer where prototypes from F4 are refined into systems—pattern libraries, CT riffs, musical grammars, improvisation logics, Nonstarter economics—so concentrated practice can run independent of outcome-chasing.

+ because: Without a stable concentrator layer, FUTON4' s tranquility and FUTON6' s argument power cannot synergise.

! instantiated-by: Prototype 1 — Patterns of Improvisation (Music ⇔ Math ⇔ CT) [🔧/归 📄/世]
+ context: You have identified a major through-line: improvisation = category theory = MetaCA = design patterns = your life practice.

+ if: You want FUTON5 to unify musical, mathematical, and pattern improvisation into one coherent substrate.

+ however: Music theory, CT, and symbolic improvisation currently live in different notebooks, chats, and memory fragments.

+ then: Create a small “Patterns of Improvisation” protobook with 8–12 patterns (e.g., Prototype → Production, Kleisli Turn, Return to I, Phase-Switch, Noise Budget, Concentration Spiral), express each as riff + CT sketch + musical example + MetaCA mapping, and store them as structured F5 nodes consumable by F4 (Arxana) and F6 (argumentarium).

+ because: An integrated improvisation grammar becomes FUTON5' s internal “samādhi engine.”

! instantiated-by: Prototype 2 — Concentration Engine (Chops, Not Goals) [🔧/江 📄/己]
+ context: Concentration in F5 is about capacity, not outcomes: sustained work on craft (music, CT, code) without dangling the carrot of “success.”

+ if: You want a system for renewable, non-self-punishing discipline.

+ however: Current practice cycles depend heavily on mood, projects, or institutional demands.

+ then: Establish a “Concentration Engine” protocol with 3–5 stable practice grooves (bass technique, CT reading, coding drills, MetaCA tinkering), ensure each groove has low activation energy and clear termination points, and track them in FUTON5 as patterns rather than habits.

+ because: Concentration emerges from rhythm, not obligation.

! instantiated-by: Prototype 3 — Pattern Library (Post-Flexiarg) [🔧/但是 📄/世]
+ context: Flexiarg formalises arguments (F6) while FUTON5 formalises patterns.

+ if: You want a domain-agnostic, cross-modal pattern language scaffold that sits between Arxana and the argumentarium.

+ however: Existing patterns are scattered across papers (P4P, P4D, P4NG) and are not harmonised.

+ then: Define a minimal F5 pattern schema (@pattern-id, IF/HOWEVER/THEN/BECAUSE where useful, examples across music/code/CT/life, invariants/failure modes, “improvisation affordances”) and refactor 6–12 core patterns into that schema.

+ because: FUTON5 needs a pattern substrate that can plug directly into FUTON6' s argument logic.

! instantiated-by: Prototype 4 — MetaCA as Improvisation Physics [🔧/己 📄/我]
+ context: MetaCA is your 2014 prototype of

FUTON6

@multiarg futon6/devmap
@title FUTON6 Development Map — Hyperreal Dictionary & Mathematical Habitat
@audience futon-devs, pattern-agents, HDM-stewards, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Equanimity (upekkhā)
@IFR: FUTON6 is the “mathematical habitat” layer of the stack: the place where mathematics becomes inhabitable by humans and agents through a pattern-indexed, flexiformal, open-ended dictionary. It draws verified argumentation from FUTON3; structural memory from FUTON1; and provides the conceptual terrain for FUTON7' s civic-scale reasoning. F6 is the “pattern-city of mathematics”, not the execution engine (F3) nor the editing environment (F4).

@state Design-only vision; no dedicated HDM code or imports exist yet beyond conceptual outlines.

@next start

! instantiated-by: Prototype 0 — FUTON6 Frame & Charter [🔧/义 📄/今]
+ context: F6 names the HDM layer: the place where mathematical content, scholia, and argument patterns live as shared habitat.

+ if: A crisp scope is needed so HDM does not dissolve into “general aspiration.”

+ however: Current F6 references still blur its role with F3 execution and F4 editing.

+ then: Write the F6 charter: (1) remit, (2) neighbours (F3/F4/F5), (3) responsibilities (stewardship of content, scholia, pattern indexing), (4) non-responsibilities (checking = F3; tooling = F4).

+ because: A charter stabilises FUTON' s cascade and prevents scope creep.

! instantiated-by: Prototype 1 — Sphere Architecture (Sloterijk Integration) [🔧/无 📄/门]
+ context: Sloterijk' s spheres theory provides the conceptual grammar for collective enclosures, atmospheres, and shared immunological structures.

+ if: You want F7 to model and cultivate “atmospheres of practice” —shared civic containers where reasoning, creativity, and care become normal.

+ however: No translation exists between spherology and the FUTON stack; F7 has no “sphere-operators” yet.

+ then: Define three sphere types for F7: 1) micro-spheres (dojo, band, mentoring dyads), 2) meso-spheres (Hyperreal teams, open research cohorts), 3) macro-spheres (post-state civic networks). Document how FUTON1–6 contribute to each sphere.

+ because: Civilisation-scale practice begins with sphere-scaping—the deliberate shaping of atmospheres that support distributed wisdom.

! instantiated-by: Prototype 2 — Hyperreal Enterprises as Seed Crystal [🔧/己 📄/义]
+ context: Hyperreal Enterprises already functions as a micro-institution with values, narrative, economic flow, and client-based R&D loops.

+ if: You want a living prototype that tests post-state organisational design in real economic terrain.

+ however: Hyperreal' s civic/institutional implications remain latent rather than formally articulated.

+ then: Position Hyperreal explicitly as F7' s seed crystal: a minimal self-governing unit practicing equanimous decision-making, distributed documentation, pattern-based governance, and nomadic economic strategy (consulting as caravan model).

+ because: A concrete seed institution anchors F7' s abstractions into lived practice and iterates the world-making in real time.

! instantiated-by: Prototype 3 — Amazon-Breaker Function [🔧/引]
+ context: “Amazon-breaker” describes technologies and social forms that prevent extractive centralisation, platform capture, and monoculture.

+ if: You want F7 to actively resist the gravitational pull toward state-like apparatuses and exploitative superstructures.

+ however: No operational definition of “Amazon-breaking” exists within the FUTON stack; it risks becoming a slogan rather than a design constraint.

+ then: Define Amazon-breaker heuristics (e.g. decentralised memory, pattern-encoded governance, nomadic modularity, multi-agent redundancy, civic equanimity, reverse-platformisation). Tag any

FUTON7

@multiarg futon7/devmap
@title FUTON7 Development Map — Nomadic Civilisation & Civic Equanimity
@audience futon-devs, sphere-architects, future-you
@tone formal-analytic
@style roadmap
@allow-new-claims true
@length any
@factor Civic equanimity (upa-upekkhā)
@IFR: FUTON7 holds the civic field of wisdom, translating futons 1–6 into sphere architectures, Amazon-breaker heuristics, and Hyperreal-derived seed institutions that cultivate equanimous, post-state civic life—the upa-upekkhā of institutions.

@state Design-only narratives outlining the civic field; no implemented prototypes or institutions yet beyond essays.

@next start

! instantiated-by: Prototype 0 — Naming the Layer (Civic Field of Wisdom) [🔧/田]
+ context: FUTON7 designates the highest layer of the stack: the “martial application” of all prior futons toward the shaping of a post-state nomadic civilisation.

+ if: You want an explicit locus for work that concerns culture, governance, civic habits, collective intelligence, non-state coordination, and world-building.

+ however: F7 has been described indirectly (Amazon-breaker, Hyperreal as seed crystal, Sloterijk' s spheres) without a clear development remit.

+ then: Define F7 as the civic field where equanimity (upa-upekkhā) is scaled from personal practice to institutional atmosphere; write its mandate: “Support distributed, non-state, resilient cultural formations with Hyperreal Enterprises as a seed instance.”

+ because: Naming the layer clarifies direction: F7 = how the world changes when Futons 0–6 are applied “socially” rather than individually.

! instantiated-by: Prototype 1 — Sphere Architecture (Sloterijk Integration) [🔧/无 📄/门]
+ context: Sloterijk' s spheres theory provides the conceptual grammar for collective enclosures, atmospheres, and shared immunological structures.

+ if: You want F7 to model and cultivate “atmospheres of practice” —shared civic containers where reasoning, creativity, and care become normal.

+ however: No translation exists between spherology and the FUTON stack; F7 has no “sphere-operators” yet.

+ then: Define three sphere types for F7: 1) micro-spheres (dojo, band, mentoring dyads), 2) meso-spheres (Hyperreal teams, open research cohorts), 3) macro-spheres (post-state civic networks). Document how FUTON1–6 contribute to each sphere.

+ because: Civilisation-scale practice begins with sphere-scaping—the deliberate shaping of atmospheres that support distributed wisdom.

! instantiated-by: Prototype 2 — Hyperreal Enterprises as Seed Crystal [🔧/己 📄/义]
+ context: Hyperreal Enterprises already functions as a micro-institution with values, narrative, economic flow, and client-based R&D loops.

+ if: You want a living prototype that tests post-state organisational design in real economic terrain.

+ however: Hyperreal' s civic/institutional implications remain latent rather than formally articulated.

+ then: Position Hyperreal explicitly as F7' s seed crystal: a minimal self-governing unit practicing equanimous decision-making, distributed documentation, pattern-based governance, and nomadic economic strategy (consulting as caravan model).

+ because: A concrete seed institution anchors F7' s abstractions into lived practice and iterates the world-making in real time.

! instantiated-by: Prototype 3 — Amazon-Breaker Function [🔧/引]
+ context: “Amazon-breaker” describes technologies and social forms that prevent extractive centralisation, platform capture, and monoculture.

+ if: You want F7 to actively resist the gravitational pull toward state-like apparatuses and exploitative superstructures.

+ however: No operational definition of “Amazon-breaking” exists within the FUTON stack; it risks becoming a slogan rather than a design constraint.

+ then: Define Amazon-breaker heuristics (e.g. decentralised memory, pattern-encoded governance, nomadic modularity, multi-agent redundancy, civic equanimity, reverse-platformisation). Tag any

REPL folklore—is what lets FUTON3 proofs and FUTON7 civic tooling depend on the same facts.

+ next steps: DONE?: check the that release candidate matches this spec (futon1/apps/graph-memory/README.md:1-160); DOCS: do an "audit" of currently generated content to check the workflow

! instantiated-by: Prototype 4 — XTDB Operations & REPL Stories [👁️/分 🧩/甲]

+ context: The stack mirrors all graph mutations into XTDB and provides notes for running a REPL against the same database.

+ if: You want developers to comfortably inspect, query, and repair the persistent store.

+ however: Common workflows (inspection, counting, ego traversal, cleanup) still require reading raw XT queries and ad-hoc snippets.

+ then: Package REPL recipes as reusable fns ('list-entities' , 'entity-relations' , 'counts'), add a small "XT stories" doc with copy-paste examples, and script a salience-survives-restart demo.

+ because: Lowering XT friction makes the persistent layer feel like a first-class part of futon1 rather than a hidden backend.

! instantiated-by: Prototype 5 — Open-World Ingest Workflows [👁️/世界 🍷/开]

+ context: The open-world ingest tool streams arbitrary text through CoreNLP, generating deterministic entity IDs and OpenIE relations in XTDB.

+ if: You want futon1 to support offline bulk ingestion as well as interactive chat.

+ however: Ingest currently lives as a standalone CLI with minimal guidance on how its data joins the basic-chat focus-header world.

+ then: Define standard data-dir layouts for shared XTDB, write small pipelines that ingest corpora then query them from 'basic-chat-demo' , and add golden tests for ingest semantics (ID stability, negation flags, co-occurrence).

+ because: Robust ingest workflows let futon1 accumulate long-term background knowledge, not just per-session memories.

! instantiated-by: Prototype 6 — Intent Dictionary & Thematic Clusters [👁️/无 🧭/上]

+ context: The deterministic intent classifier already maps utterances onto conversational and RID-inspired clusters, but today those labels peak inside the logs.

+ if: You want intent to act as policy—steering focus headers, salience boosts, and downstream pattern choices.

+ however: there is no canonical intent taxonomy, no hooks from classifier output into the CLI/graph, and no regression suites proving these effects.

+ then: publish the intent dictionary as a living document, wire intents into the focus-header builder (boost/suppress entities, tag trails), and add fixtures showing how each theme alters XTDB mirrors and futon3 checks.

+ because: intent only matters if it shapes behaviour; otherwise the classifier is cosplay rather than dhammavicaya.

! instantiated-by: Prototype 7 — CLI Ergonomics & Multi-Instance Isolation [👁️/资料 🍷/示]

+ context: The CLI already exposes bang commands, context flags, and environment variables for data-dir and XT config overrides.

+ if: You want futon1 to run multiple isolated demos (tests, experiments, production-like runs) on the same host.

+ however: Best practices for 'BASIC_CHAT_DATA_DIR' , XT config selection, and '-reset/-compact/-export' are scattered and easy to misuse.

+ then: Ship a "profiles" doc (dev/test/prod) with recommended flags and env vars, add smoke tests that spin up multiple instances with different data-dirs, and provide a '/status' helper to show active config.

+ because: Good ergonomics and isolation make futon1 safer to embed in other tools and workflows.

! instantiated-by: Prototype 8 — Golden Scripts & Performance Benchmarks [👁️/不要 🍷/久]

+ context: Scripted EDN conversations and golden outputs already exercise the focus-header flow and older basic-chat protocols.

+ if: You want confidence that future changes don't break determinism, latency thresholds, or header quality.

+ however: Golden coverage is still patchy and doesn't include dedicated performance baselines or large-script runs.

+ then: Curate a small suite of canonical scripts (short-chat, long-history, ingest+chat), time them under CI, and treat both outputs and runtime ranges as golden.

controlled sensory inputs.

+ however: There is no standalone harness that feeds synthetic mu/observation pairs.

+ then: Build '(eval-policy mu prec obs config)' as a harness and add golden action rankings for ten canonical scenarios (e.g. rich food, enemy close, starvation risk).

+ because: Stabilised policies are mandatory before benchmarking or bridging into futon1 salience services.

! instantiated-by: Prototype 5 — World Mechanics Contract (war.clj) [👁️/世界 🍷/但]

+ context: The simulator handles evaporation, movement, gather/deposit, pheromone dynamics, hunger propagation, reserves, and termination.

+ if: You want FUTON2 to be a reliable agent-environment backend.

+ however: Movement heuristics, gather thresholds, and white-streak rules are powerful yet lightly documented.

+ then: Document all invariants (pheromone caps, grid bounds, gather limits), seed movement decisions for determinism, and split movement modes into testable units.

+ because: A clear contract keeps the world trustworthy when other prototypes depend on it for episodes.

! instantiated-by: Prototype 6 — Live Analyzer & Pivot Stream Stabilisation [👁️/在]

+ context: The live analyzer consumes pivot events, computes EWMA metrics, detects starvation spirals, and prints mode transitions.

+ if: You want analyzer outputs to feed futon3 evaluations or futon4 reflexive agents.

+ however: Pivot schemas are implicit and analyzer assumptions are undocumented.

+ then: Freeze a versioned pivot-row schema, add tests for required keys (:act, :mode, :cargo, :ing, etc.), and ship an example analyzer plugin for futon4.

+ because: Pivot streams are FUTON2's event bus and must behave like a stable ABI.

! instantiated-by: Prototype 7 — Classic vs AIF Benchmark Suite [👁️/系统]

+ context: Benchmark scripts compare classic ants to AIF runs across scores, durations, and G-ema.

+ if: You want formal behavioural comparisons rather than anecdotes.

+ however: Only three preset scenarios exist and there is no variance analysis.

+ then: Add parametrised benchmarks spanning grid sizes, pheromone decay, hunger burn, and tau floors; capture plots and golden CSVs.

+ because: FUTON2 should function as a scientific system with reproducible metrics.

! instantiated-by: Prototype 8 — External Control & Episode Export [👁️/予 🍷/绘]

+ context: 'simulate' drives the war sim but offers no external stepping API.

+ if: You want futon3 or futon4 to request "give me N steps with this config."

+ however: Stepping remains tied to the CLI and HUD.

+ then: Expose '(futon2.step world n {:emit-pivot? true})' , add EDN exports of full episodes, and keep pivot batches consistent.

+ because: Exported episodes are critical for agent training, memory, and pattern extraction.

! instantiated-by: Prototype 9 — futon1 ↔ futon2 Alignment Layer [👁️/已 🍷/刊]

+ context: FUTON1 emits focus headers and salience; FUTON2 emits percept/action streams.

+ if: You want the ants to act as embodied readers + writers of futon1's graph.

+ however: there is no shared dictionary between entities/intents and colony modes, so exports are useless to FUTON1.

+ then: define the crosswalk (entities ↔ colonies, intents ↔ behaviours), publish an export schema, and prove it by replaying one colony while FUTON1 ingests the stream and surfaces it back to users.

+ because: a working bridge proves ants can reason over the same facts as the rest of the stack.

! instantiated-by: Prototype 10 — Curriculum Scenarios & AIF Behaviour Cards [👁️/乡 🍷/归]

+ context: The current war scenario is rich but single-purpose.

+ if: You want FUTON2 to double as a pedagogical engine for Active Inference.

+ however: There is no named curriculum.

+ then: Create behaviour cards (home-return, starvation spiral, over-exploration, defensive lattice, greedy gatherer collapse), simulate each with fixed seeds, and export the episodes.

+ because: These cards seed pattern extraction and future training systems.

! instantiated-by: Prototype 11 — AIF Agent Debugger & Mu/Precision Visualiser [👁️/眼 🍷/—]

+ context: Pivot logs are textual and require manual

transport contract is frozen.

+ evidence: 'src/futon3/checks.clj' , 'src/f2/router.clj' , and 'test/transport_test.clj' (check route cases) exercise the DSL end-to-end.

! instantiated-by: Prototype 4 — Trail & Proof-State Journal [👁️/几 🍷/弓]

+ context: Wisdom trails already exist for tracing exploration, but now they must document proof obligations and how work sessions discharged them.

+ if: You want every check, dismissal, and new pattern idea to leave an auditable trail.

+ however: Trails are still general-purpose exploration logs, not proof-state journals.

+ then: Extend the trail schema with ':pattern/id' , ':obligation/id' , ':action/tags' , and ':delta/joy' , add rollups that show which devmap clauses advanced per day, and expose a '/musn/trails/proof' export for futon4 archives.

+ because: The whole point is to measure how daily practice advances the devmaps.

! instantiated-by: Prototype 5 — Workday Instrumentation & aob-chatgpt Bridge [👁️/弓 🍷/几]

+ context: Within 'aob-chatgpt' we already experiment with reflecting on daily actions against devmap obligations.

+ if: You want FUTON3 to serve as the backend for those reflections.

+ however: There is no official API for submitting "workday claims," mapping them to patterns, or receiving verification feedback; roles of "workday participants" and "instrumentation stewards" are undefined.

+ then: Publish a 'workday/submit' endpoint (inputs: timestamp, activity, evidence links), run it through the check DSL + embedding, respond with pattern hits/misses plus follow-up obligations, and name the actors explicitly (participants supply evidence; instrumentation stewards curate dashboards); integrate the same flow into the ChatGPT tooling.

+ because: This is how empowered action ("joy") becomes measurable proof progress and how role responsibilities stay legible.

+ next: Replace the temporary 'futon3/logs/workday.edn' queue with a Futon1 persistence call ('workday-graph') and add HTTP affordances + dashboards so instrumentation stewards can audit submissions without tailing logs.

+ evidence: Workday intake lives in 'src/futon3/workday.clj' with coverage in 'test/transport_test.clj' (workday submission) and README sections describing the API.

! instantiated-by: Prototype 6 — Pattern Creation Workbench [👁️/当 🍷/手]

+ context: New patterns emerge from trails and devmap edits, but the capture is currently manual in futon4.

+ if: You want FUTON3 to draft candidate patterns/flexiargs from accumulated proof states.

+ however: Manual backlog in FUTON4 creates tension; there is no readiness window for when drafts are "hand-off ready."

+ then: Build a summariser that clusters trails, extracts hooks, and emits draft flexiarg blocks (IF/HOWEVER/THEN/BECAUSE) tagged with suggested pāramitā, ready for futon4 editors, and declare readiness criteria (e.g., at least N supporting checks + steward review) so handoffs are predictable.

+ because: FUTON3 should not just consume the library; it should propose expansions grounded in lived work while signalling when drafts can be trusted.

! instantiated-by: Prototype 7 — Joy-as-Empowered-Action Metrics [👁️/弓 🍷/力]

+ context: Joy here = Spinozan increase in power to act, evidenced by commitments you can now keep.

+ if: You want metrics that show whether checks are empowering or draining.

+ however: Current "joy metrics" counted UI friction; they did not measure proofs discharged or obligations unblocked.

+ then: Track 'time-to-proof' , 'obligations-cleared' , 'follow-on commitments' , and 'viriyā linkages' per session, and visualise them alongside subjective notes.

+ because: Empowered action is the KPI for this futon.

! instantiated-by: Prototype 8 — futon2/futon1 Proof Hooks [👁️/当 🍷/已]

+ context: FUTON3 already generates proof states; FUTON1 (graph) and FUTON2 (energy) need those proofs as fuel.

+ if: You want every accepted check to become a graph relation and a viriyā delta without manual glue.

+ however: today the exports are bespoke Clerk notebooks or one-off scripts.

+ then: standardise two adapters— 'proof->graph' (materialise relations with provenance) and

semantic memory.

+ if: You want collaborators (Serena, Rob, Harvinder, etc.) to share a semantic space.

+ however: No multi-user or sync layer exists yet.

+ then: Use Datascript local + XTDB remote as stepping stones.

+ because: This will open the door to multi-agent co-reasoning and shared prototyping.

! instantiated-by: Prototype 6 — Argumentarium / Scholium Mode [👁️/本 🍷/但]

+ context: You want something like Spinoza's "Ethics" but with imports, hypergraphs, and links.

+ if: You want a library of arguments with provenance, alternative proofs, counterproofs.

+ however: Current flexiargs do not yet link into a live scholium tree.

+ then: Add scholium metadata + "view scholium tree" mode in Arxana.

+ because: This becomes the living mathematical-philosophical substrate.

+ next (inclusion/transclusion UX):

- Visually differentiate included vs. transcluded regions in the main article buffer.
- Provide toggle commands to highlight each derivative type independently.
- Document the visual affordances and add a regression test covering both styles.

! instantiated-by: Prototype 7 — XTDB Integration [👁️/久 🍷/长]

+ context: Bi-temporal, multi-user storage needed for long-term evolution.

+ if: You want durability, branching histories, provenance queries.

+ however: XTDB idioms differ from Datascript; schema migration needed.

+ then: Rekey schema, add transaction logs, implement time-travel queries.

+ because: This turns Arxana into a serious memory engine.

! instantiated-by: Prototype 8 — Pattern Library Activation [👁️/但 🍷/却]

+ context: You have dozens of patterns (Mojo, ORP, Paramita, Flexiarg, etc.).

+ if: You want agents to recommend patterns or detect pattern-relevant situations.

+ however: Patterns are not yet encoded in memory or linked to triggers.

+ then: Import patterns as graph rules; map IF/HOWEVER/THEN/BECAUSE into triggers + actions.

+ because: This yields genuine pattern-aware intelligence.

! instantiated-by: Prototype 9 — StackExchange Import [👁️/叉 🍷/无]

+ context: StackExchange holds thousands of argued Q/A pairs that fit Arxana's schema.

+ if: You want to harvest them as production rules with provenance.

+ however: there is no importer aligning their Markdown into our graph schema.

+ then: build a parser, map Q/A + comments into rule nodes, annotate tags, and cross-index them with existing flexiargs.

+ because: seeding the memory with curated Q/A pairs gives Arxana instant argumentative density.

! instantiated-by: Prototype 10 — Corpus Mathematics Layer [👁️/但 🍷/却]

+ context: You want to bridge natural/formal logic (Strand 1B).

+ if: You want Dirichlet distributions and Abelian groups to live in the same memory model.

+ however: No symbolic math integration exists yet.

+ then: Build math namespace + flexiarg-lemma type + REPL test harness.

+ because: This is how FUTON becomes an "AI Scientist."

! instantiated-by: Prototype 11 — Literary Interface (Penmaster) [👁️/好]

+ context: You want Arxana to also serve as a book-editing and generative-literary tool.

+ if: You want to edit Penmaster or draft your book via Arxana blocks.

+ however: No good UI for long-form text yet.

+ then: Add "manuscript mode" + chunking + cross-reference map.

+ because: FUTON becomes the writing environment.

! instantiated-by: Prototype 12 — Year-End Synthesis [👁️/末 🍷/年]

+ context: The prototype cycle ends; reflection and forward-planning needed.

+ if: You want long-term stability, and upgraded beta-schema.

+ however: Accumulated divergence between prototypes must be unified.

+ then: Run full PAR + CLA + DREAM integration cycle.

+ because: This sets the stage for FUTONS+.

local-rule self-evolution and mirrors your 2025 simulation-based self-understanding.

+ if: You want to reinterpret MetaCA as the "physics" of improvisation: everything evolves locally but is constrained globally by a deep rule (Dhamma / AIF).

+ however: MetaCA code exists, yet its conceptual integration with FUTONS is not explicit.

+ then: Reframe MetaCA as an F5 conceptual engine where local rules equal riffs/CT arrows/pattern transforms, the global rule embodies AIF, edge-of-chaos represents productive creativity, collapse modes name burnout/ego/overcontrol, and capture this in one short essay inside the F5 devmap.

+ because: This gives FUTONS a mathematical "feel" that is embodied, generative, and non-institutional.

! instantiated-by: Prototype 5 — Nonstarter Social Physics [👁️/无 🍷/元]

+ context: Nonstarter = anti-teology, refusal of capture, joy in giving, Patti Smith energy, LANDBACK aesthetics.

+ if: You want FUTON5 not only to concentrate craft but to concentrate value without reproducing institutional extraction.

+ however: There is no current model linking creative concentration with non-extractive flows.

+ then: Define a simple Nonstarter Ledger inside F5 that tracks "value given" (hours, riffs, patterns released), "value received" (collaboration, uplift, resonance), enforces a "no-ask posture" protocol, encodes reciprocity loops based on pattern contribution rather than money, and distinguishes RESCUE requests from COLLABORATION requests.

+ because: With a functional Nonstarter economy, FUTON5 becomes spiritually and materially sustainable.

! instantiated-by: Prototype 6 — Improvisation Metrics (Not KPIS) [👁️/归 🍷/又]

+ context: Metrics destroy creativity when tied to teleology.

+ if: You want a way to sense concentration without collapsing into institutional logic.

+ however: No metrics currently track musical, cognitive, or conceptual "chops."

+ then: Define a set of "Improvisation Indicators" (Noise Budget, Flow Duration, Surprise-to-Stability Ratio, Return-to-I Coherence, Cross-Domain Lift between music/writing/coding) and track them qualitatively in F5 without gamification.

+ because: FUTON5 needs feedback, not surveillance.

! instantiated-by: Prototype 7 — F4⇌F5⇌F6 Conduction Pathways [👁️/什么]

+ context: F4 is tranquility, F6 is war machine, and F5 is the samādhi bridge.

+ if: You want to flow smoothly between relaxed inscription (F4), concentrated craft (F5), and structured reasoning (F6).

+ however: There is no explicit pathway linking the three.

+ then: Define three conduction modes—F4→F5 (prototype → practice, what to work on today), F5→F6 (practice → argument, what becomes a formal insight), F6→F5 (argument → pattern, what needs to be embodied)—and document these flows as short "transmission patterns."

+ because: Without conduction, the stack fragments instead of concentrating.

! instantiated-by: Prototype 8 — 12-Prototype Year (Concentration Arc) [👁️/年 🍷/岁]

+ context: You are already running a 12-prototype year.

+ if: You want FUTONS to shape it into a concentration arc rather than a sequence of builds.

+ however: There is no formal mapping of prototypes to F5 themes.

+ then: Assign each prototype a concentration domain (MetaCA, Patterns of Improvisation, CT chops, Musical chops, Flexiarg → pattern transforms, Nonstarter ledger, Pattern economy, F5 system interfaces, Meta-Improvisation, AIF integration, Relational improvisation patterns, Year-end coherence synthesis) and keep the map current as the year unfolds.

+ because: Concentration deepens through cyclical return.

! instantiated-by: Prototype 9 — Yearly F5 Review (Concentration Renewal) [👁️/口 🍷/这样]

+ context: Concentration weakens if not refreshed.

+ if: You want F5 to stay light, concentrated, and improvisational rather than dogmatic.

+ however: Without review, patterns accrete into heaviness.

+ then: Run a yearly cycle that prunes dead patterns, refines musical grammars, updates CT riffs, consolidates improvisation patterns, refreshes the Nonstarter ethos, and removes anything that smells like KPIS.

+ because: Concentration equals aliveness, and aliveness must be renewed.

every entry re-invents its own outline language.

+ then: Freeze a small but expandable L3 pattern set, tag the seed domain entries with those moves, and expose L3 IDs to F6-aware agents as their default reasoning vocabulary.

+ because: L3 patterns supply the "physics" of mathematical reasoning and make HDM truly traversable.

! instantiated-by: Prototype 5 — Layer-4 Domain Patterns [👁️/本]

+ context: Subject areas (MSC slices) have canonical machinery—ultrafilters, Yoneda, compact operators, Markov kernels—that sit between general motifs and specific entries.

+ if: FUTON6 needs Layer-4 pattern nodes that capture domain-specific constructs, typical lemmas, and dependencies.

+ however: Today those structures are either flattened into individual entries or treated as vague generalities.

+ then: Identify 10-20 L4 candidates for the seed slice, represent each as a node with domain metadata, canonical usages, and referenced L3 moves, and ensure L4 nodes refactor like any other content.

+ because: L4 patterns provide the local physics that make each mathematical neighbourhood habitable.

! instantiated-by: Prototype 6 — L3<->L4 Relation Map [👁️/互]

+ context: FUTON6 spans HDM motifs, L3 reasoning physics, L4 domain machinery, and concrete entries.

+ if: Imports and viewers need an explicit handshake so layers do not blur.

+ however: No rulebook specifies how L3 motifs embed in L4 constructs or how L4 clusters point back to universal moves.

+ then: Publish the relation: L3 = field-independent moves, L4 = field-dependent machinery; require new entries to declare which layer they instantiate, and build viewers showing the L3<->L4 wiring for any definition, proof, or example.

+ because: A fractal, Alexandrian cascade keeps FUTON6 coherent as it scales.

! instantiated-by: Prototype 7 — HDM Dojo Mode [👁️/友]

+ context: Mathematics becomes native when practiced like a dojo: progressive disclosure, sparring partners, and navigable landscapes.

+ if: FUTON6 is to turn math into a habitable habitat for humans and agents.

+ however: Current assets are static entries; they do not yet function as a training ground that links moves, contexts, views, and guidance.

+ then: Blend L3/L4 tagging with scholium landscapes, progressive outline->detail modes, and agent sparring protocols so users can practice reasoning moves inside HDM itself.

+ because: Mathematical fluency emerges from embodied practice, not passive reading; the dojo keeps HDM alive.

! instantiated-by: Prototype 8 — Scholium & Proof-Landscape Mode [👁️/示 🍷/义]

+ context: F3 emits proof trails; F1 provides graph memory.

+ if: F6 wants users/agents to traverse mathematics as connected terrain.

+ however: No math-specific landscape viewer exists.

+ then: Add scholium metadata (e.g. related lemmas, commentary nodes, viewpoint shifts) and render "neighbourhoods": clusters of results, arguments, patterns, counterexamples.

+ because: Landscapes differentiate HDM from a static encyclopedía.

! instantiated-by: Prototype 9 — Language ⇌ Math Transformations [👁️/文 🍷/更]

+ context: F6 aspires to maintain both NL-friendly and math-native views.

+ if: You want consistent transformations across entries.

+ however: NL ↔ math translations are currently ad hoc.

+ then: Define several translation templates for the seed domain (e.g., "definition block" , "example pattern" , "lemma skeleton") and exercise them with FUTON3's flexigrammar.

+ because: Even partial bilingualism greatly improves usability.

F6/F5/F3 work that strengthens these heuristics.

+ because: Anti-monopoly structural feedback must be designed in from the start; otherwise F7 collapses back into the state form.

! instantiated-by: Prototype 4 — Civic Equanimity Protocol (Upa-upekkhā at Scale) [👁️/弓]

+ context: You have identified F7 with the Pāramitā of Equanimity: balance-beyond-balance, the stability of clarity under pressure.

+ if: You want this Pāramitā to inform collective, not just individual, behaviour.

+ however: No mechanism yet translates individual reflective rhythm (Futon0, PāRamita cycles) into group-level civic practice.

+ then: Design a Civic Equanimity Protocol: shared PAR cycles, distributed pattern reviews, collective decision check-ins (fruits/orbs used as civic signals), and non-reactive governance rituals adapted from tai chi application.

+ because: Institutions that act with even-minded care are the defining signature of a future civilisation that is neither authoritarian nor chaotic.

! instantiated-by: Prototype 5 — Multi-Sphere Learning Ecosystems [👁️/女 🍷/互]

+ context: You have real data from ORP workshops, PLACARD, band rehearsals, strategic consulting, dojo practice, and peer mentoring.

+ if: You want F7 to articulate what a mature, distributed learning ecosystem looks like.

+ however: These practices are scattered across domains and not formalised into a unified civic-learning grammar.

+ then: Identify recurring multi-sphere motifs (shared inquiry, pattern elicitation, improvisational coherence, mutual regulation); map them to F5 patterns and F6 reasoning operations; prototype a "Nomadic Learning Cell" drawing from Hyperreal + ORP + dojo + band.

+ because: Nomadic civilisations endure by transmitting culture horizontally, not vertically; multi-sphere learning is their engine.

! instantiated-by: Prototype 6 — Economic Nomadism & Post-State Viability [👁️/但]

+ context: Future nomadic groups will require economic resilience untethered from state sponsorship, salary structures, and bureaucratic scaffolds.

+ if: You want F7 to model a viable economic substrate that is anti-fragile, creative, and regenerative.

+ however: Current income streams (job, consulting, side projects) are not yet framed as a prototype nomadic economy.

+ then: Define a minimum viable nomadic economy: (1) consulting caravans (Hyperreal), (2) intellectual capital loops (F6 argumentarium + patterns), (3) low-cost, high-leverage creative production (band, book, prototypes), (4) geographic and institutional mobility.

+ because: F7 must be economically real, not merely philosophically interesting.

! instantiated-by: Prototype 7 — Interoperability with State Forms without Capture [👁️/长]

+ context: Post-state formations will necessarily interoperate with existing state institutions (universities, funders, cultural organisations).

+ if: You want to retain freedom while collaborating fruitfully.

+ however: Nomadic groups risk being absorbed, bureaucratised, or instrumentalised by state forms if guardrails aren't explicit.

+ then: Establish an F7 "Porous Interface Charter" clarifying how Hyperreal and related projects interact with institutions: export patterns, consulting, co-reasoning, but preserve governance, culture, and memory internally.

+ because: Being permeable without becoming captured is essential for long-term autonomy.

! instantiated-by: Prototype 8 — Civilisational Simulations (Nomadic Scenario Work) [👁️/扎]

+ context: F7 needs a way to test future structures before implementing them.

+ if: You want to prototype nomadic civic forms as thought-experiments and fictional/technical hybrids.

+ however: No structured simulation practice exists across scenarios.

+ then: Develop 2-3 scenario bundles (e.g. "Nomadic Math Sangha," "Distributed Research College," "Hyperreal Field Office in the Zone") and tie each to relevant F6 argument bundles and F5 pattern clusters.

+ because: Scenario practice turns abstract philosophy into actionable civic design.

! instantiated-by: Prototype 9 — Year-End Nomadic Governance Audit [👁️/真 🍷/要]

+ context: Cultures need periodic tuning to maintain coherence and avoid drift.

+ because: Benchmarks turn futon1 into a measurable platform rather than an anecdotal demo.
! instantiated-by: Prototype 9 — External Client & Agent Bridge [🏗️/无👤/已]
+ context: Focus-header EDN is already consumable by chatgpt-shell, Codex, or futon4, but every integration has been bespoke.
+ if: You want futon1 to expose a purposeful bridge so external agents see the same contract as the CLI.
+ however: there is no signed JSON schema, no HTTP shim, and no reference client showing how to stream utterances and receive headers deterministically.
+ then: publish a tiny 'focus ' API, wrap the CLI in a documented RPC, and ship sample clients (Clojure + at least one other language) that replay real transcripts against it.
+ because: when the bridge is first-class, you can swap or stack agents freely without forking the core or relying on brittle stout scraping.
! instantiated-by: Prototype 10 — Futon4 / Arxana Export Hooks [❤️/已🔗/本]
+ context: futon4 will want to ingest focus headers, EDN snapshots, and XTDB slices as graph inputs for Arxana and higher-level pattern work.
+ if: You want futon1 to be the canonical source of “world state” for the year-long prototype cycle.
+ however: There is no named export profile tailored for Arxana or datascript/XT interop beyond generic ‘-export edn ‘.
+ then: Add an “arxana-export” mode that emits a compressed, schema-tagged snapshot of entities, relations, and salience suitable for futon4 import.
+ because: Tight, named integration turns futon1 into the lower memory layer of the FUTON stack rather than a sibling demo.
! instantiated-by: Prototype 11 — AI Scientist Pilot Scenarios [👩🔬/才]
+ context: The AI Scientist pilots exercised futon1/futon3 side-by-side—joyful reasoning loops that leaned on focus headers, ingest, and proof logging.
+ if: You want those pilots to serve as canonical fixtures anyone can replay or extend.
+ however: the transcripts, CLI flags, and seed graphs live in private folders, so nobody can cite or re-run them with confidence.
+ then: package each pilot as a runnable fixture (configs, transcripts, seed graph, expected headers) and wire futon3 checks so they reference those fixtures directly.
+ because: pilots only teach if they are reproducible; otherwise “joy-as-empowered-action” is just lore.
! instantiated-by: Prototype 12 — Year-End Refactor & Packaging [🏗️/未👤/串]
+ context: By the end of the cycle you will have hardened v5, clarified modules, added exports, and integrated ingest, intents, and clients.
+ if: You want futon1 to be easy to install, run, and extend by others.
+ however: Accumulated tweaks across apps may fragment conventions (flags, env vars, logging, error reporting).
+ then: Do a pass to unify naming and configuration across apps, streamline build and test tasks, and publish a top-level futon1 README that explains the whole story in one place.
+ because: A coherent, well-packaged futon1 becomes the dependable base layer for FUTON2+ and for collaborators who were not present at inception.

parsing.
+ if: You want interactive introspection of mu vectors, tau evolution, and weighted errors.
+ however: No visualiser or “follow Alice” mode exists.
+ then: Build a Clerk notebook (or similar) that plots tau, hunger, errors, and pivot timelines, plus a mode that traces one agent’ s full mu timeline.
+ because: Visual insight accelerates iteration and debugging.
! instantiated-by: Prototype 12 — Year-End Packaging & Story [🏗️/未👤/串]
+ context: FUTON2 must read as a coherent simulation layer to partners and future prototypes.
+ if: You want it to be adoptable as a research simulator.
+ however: The codebase is powerful but intimidating without a unifying story.
+ then: Package the release with a top-level README, whitepaper, stepping API, goldens, and versioned pivot schema.
+ because: Strong packaging turns FUTON2 into a reusable, inspectable module inside the full FUTON stack.
! instantiated-by: Prototype 13 — Graph Memory Adapter & Agent ABI [🔗/已👤/甲]
+ context: FUTON1 now offers a documented graph-memory API; FUTON2 needs to speak it to graduate beyond the sandbox.
+ if: You want ants to be the first drop-in “AIF agent” that reads fact graphs and writes back its beliefs.
+ however: percepts/actions remain locked to grid coordinates, so nothing maps to FUTON1 entity IDs or relation semantics.
+ then: design a bidirectional adapter (percepts → graph facts, graph cues → priors), version the ABI, and replay a colony run that exports beliefs into FUTON1 salience tables.
+ because: once the adapter exists, future AIF agents inherit a proven contract instead of bespoke glue.
! instantiated-by: Prototype 14 — Viriya Field & Social Activation Metrics [👩🔬/卵🚀/介]
+ context: FUTON2 carries the Factor of Awakening “energy/viriya” : vigor, commitment, and kinetic practice.
+ if: You want simulations to reflect and inform real social activation—networked collaboration, shared commitments, and practice loops across humans + agents.
+ however: Current metrics stop at hunger/tau; nothing measures vigor, pledge strength, or coordination throughput.
+ then: Introduce viriya indicators (e.g., colony activation score, pledge adherence, transfer of trails into collaborative tasks), log them per episode, and export them so futon3/futon7 can read “energy health.”
+ because: Tying the simulation to viriya turns FUTON2 into a practice accelerator instead of a curiosity.
! instantiated-by: Prototype 15 — Sandbox Bridge & Presence Interface [👤/内👩🔬/里]
+ context: FUTON3 now emphasises flexiformal proofwork but still hosts multi-user rooms and SAFE REPL affordances.
+ if: You want AIF agents (ants and successors) to inhabit those rooms, receive room/talk updates, and emit actions back into the simulation.
+ however: No interface maps MUSN room events/presence into FUTON2 percepts, nor do ants announce themselves inside FUTON3.
+ then: Define a bidirectional “presence bus” (rooms → observation cues, agent actions → MUSN events), add authentication so only designated colonies bridge, and ship a demo where ants report discoveries into a FUTON3 trail.
+ because: This proves that AIF agents can show up wherever proofwork happens, not just inside a war grid.
! instantiated-by: Prototype 16 — Wisdom Trail Harvest & Pattern Seeding [❤️/去👤/所]
+ context: FUTON3 trails now record which patterns were checked or advanced during human work sessions.
+ if: You want FUTON2 analyzers to ingest those trails, compare them to colony behaviour, and commission new agent goals.
+ however: There is no ingestion path for trail EDN, nor a feedback loop that says “agents, go gather evidence for Pattern X.”
+ then: Add a ‘trail->mission ‘ translator that turns FUTON3 proof deltas into FUTON2 goal queues, plus a reporting job that shows how agent runs affected subsequent checks.
+ because: Shared missions keep the “energy” loop tight between simulations and human proofwork.

'proof->energy ' (emit activation deltas for futon2 dashboards)—and lock them with fixtures + schemas.
+ because: proofs only matter when other futons can act on them automatically.
+ next: Wire the Tatami event log into FUTON1 (per 'futon3/tatami_store.clj ' TODO) so session events flow into graph-memory before we expose the adapters.
! instantiated-by: Prototype 9 — Proof Training Ground (“Flexiformal Agent Sandbox”) [🏗️/义❤️/门]
+ context: FUTON3 needs a reproducible training ground where humans and agents can practice flexiformal checks before contributing to FUTON6.
+ if: You want collaborators (including FUTON6 math stewards) to experience the proof workflow hands-on.
+ however: README/docs still emphasise ants and MUD metaphors and do not ship a simulator/tutorial bundle; roles (training facilitators vs. FUTON6 learners) and release readiness are not declared.
+ then: Package a “Flexiformal Training Ground” release: updated README, tutorials referencing 't4r/ ' training patterns, sample check sessions, and a minimal simulator that feeds FUTON6’ s Hyperreal entries; name facilitators (FUTON3 maintainers) and consumers (FUTON6 trainees), and publish readiness criteria (tutorials complete, simulator recorded, onboarding docs updated) before tagging the release.
+ because: Storytelling plus an explicit training ground turns FUTON3 into a reproducible service that upstream layers (FUTON6, FUTON7) can consume and steward.
+ next: Aggregate Tatami session proof events into 'proof-chain.edn ' (per 'tatami_store.clj ' TODO) so the training ground exports persistent transcripts.

! instantiated-by: Prototype 10 — Math-Aware Agent Protocol [❤️/义👤/分]
+ context: F6 will eventually host agents that annotate, refactor, or query entries.
+ if: You want safe and productive agent behaviour.
+ however: The current protocol is conflated with F3’ s checking API.
+ then: Define an F6-specific protocol: allowed queries (definitions, pattern IDs, dependency graphs), acceptable outputs (annotations, reorganisations), and guardrails (no overwriting canonical entries without review).
+ because: Math-literate agents are central to HDM-as-habitat.
! instantiated-by: Prototype 11 — Non-Capture Guardrails [👤/刊]
+ context: HDM inherits the “war machine” mandate: remain non-state, non-enclosed.
+ if: F6 is to remain an open mathematical habitat.
+ however: Capture risks (institutional, corporate, epistemic) are unarticulated.
+ then: Publish a Non-Capture Charter specifying: open licensing, provenance transparency, anti-proprietary entanglement, dual-representation redundancy. Mark entries whose sources impose constraints.
+ because: Commons integrity is crucial for HDM’ s long-term survival.
! instantiated-by: Prototype 12 — F6🔗F7 Scenarios [❤️/化👩🔬/支]
+ context: FUTON7 builds civic imaginaries and long-range scenarios.
+ if: Math should inform F7, not remain siloed.
+ however: There is no pipeline from HDM entries to F7 narratives.
+ then: Package small argument bundles (e.g. growth/decay patterns, equilibrium models, scaling laws) as exportable F6→F7 kits.
+ because: Mathematics should feed collective imagination, not remain inert.
! instantiated-by: Prototype 13 — FLEX Training Ground Link [❤️/新]
+ context: F3 now ships the “Flexiformal Training Ground.”
+ if: F6 wants stewards who know how to handle patterns and proofs.
+ however: F6 does not yet reference this training layer.
+ then: In onboarding docs, require that new math agents/humans pass the F3 training ground. Capture their trails and store them as scholium notes attached to early entries.
+ because: This closes the developmental loop and ensures quality control.

+ if: You want F7 to stay aligned with equanimity, abundance, independence, and anti-capture heuristics.
+ however: No cycle currently exists that evaluates Hyperreal, the prototypes, and the emerging civic sphere as a coherent F7 entity.
+ then: Create an annual F7 governance audit, integrating: (1) PARAMita maps, (2) fruit/orb data, (3) economic logs, (4) Hyperreal practice notes, (5) scenario updates, (6) stack-wide tensions.
+ because: A post-state civilisation must cultivate its own continuous reflective governance ritual—its civic meditation.