The Codable World

Design for a World-Scale Intelligence in Which All Knowledge, Past and Future, Becomes Address

J. Konstapel — Leiden, 18 August 2026

Context Header

Programme. Vacuum.Net is the model of the whole net. MAZE applies it. This piece applies it to a machine that holds the knowledge of the world.

This piece. It is a design, not an essay. It specifies one rule, one kernel, four operations, five layers, and five tests. It answers one question: what has to be true for every item of knowledge, from any era, to receive one permanent code without a registry, an authority, or a re-indexing event.

Settled. The axioms N1–N5. Theorem T7 and its corollaries. The capacity ladder 1, 4, 13, 40, 121. The MAZE kernel in code, with exact tests.

Open. O2, the rewrite dynamics. The kernel addresses. It does not yet move.

Conventions. Five statuses: assumed, conditionally proved, corresponded, predicted, open. Every number needed to follow a calculation stands in the text.

Corpus. Paths to the Knot (Complete Edition, 17 August 2026); The Vacuum.Net Theory, Foundational Paper, Edition 6 (9 August 2026); the Right Brain AI series (November–December 2025); MAZE — The Human Correspondence of the Vacuum.Net Theory.

1. What Is Being Designed

A machine that holds knowledge has three jobs. It must take something in. It must put it somewhere. It must find it again.

Every system built so far does the second job badly. Dewey assigns a number by committee. A URL points at a server that will one day be switched off. An embedding gives a position that shifts the moment the model is retrained. In each case the address is granted from outside, and what is granted can be withdrawn.

The design here removes the granting. The address is computed from the thing itself. Nobody assigns it. Nobody can revoke it. Two people on opposite sides of the world, working from the same rule and the same item, arrive at the same integer.

That is the whole system. Everything below is the consequence.

The picture to hold is the net. Not a library, not a database. A fishing net, meshes within meshes, each knot a place where strands close. A piece of knowledge is not an object stored in a box. It is a position in the net — a winding at a certain depth. Coding knowledge means naming that position. Nothing more is claimed and nothing less is needed.

2. The One Rule

The system has exactly one distinction. It is applied over and over.

Relative to the reference at this ring: above, at, or below?

Above is +1. At is 0. Below is −1.

That is the rule. It has three outcomes because a net has three relations at a crossing: the strand passes over, it lies alongside, it passes under. There is no fourth case.

The rule is applied recursively. The answer at ring 1 fixes the reference for ring 2. The answer at ring 2 fixes the reference for ring 3. And so on, as deep as needed.

This recursion is what makes the codes commensurable. A design with five different questions — scale, then direction, then closure, then tempo, then role — produces five unrelated axes. Codes on unrelated axes cannot be truncated, cannot be mirrored, cannot be compared. The single rule, repeated, produces codes that can be.

Status: assumed. The single-distinction rule is the design's foundation. It is not derived here. It is what the rest is built on, and test W2 in §16 is where it breaks if it is wrong.

3. The Kernel

A sequence of answers is a route. A route is a sequence of turns drawn from $\{+1, 0, -1\}$.

Theorem T7. Every route corresponds to exactly one integer, and every integer to exactly one route, through balanced ternary notation.

The value of a route is the sum of each turn times its place value, where places run 1, 3, 9, 27, 81, and so on.

Worked example. Take the route (+1, −1, −1, −1, +1), read from the deepest place down:

$81 - 27 - 9 - 3 + 1 = 43$.

The route (0, 0, 0, 0, 0) has value 0. Rest is address zero.

The route (−1, +1, +1, +1, −1) has value −43. It is the mirror of the first, turn by turn.

This is in code. The file is `server/maze-kernel/maze.ts`. It is 114 lines. It has no database dependency. It carries 96 lines of tests that recompute the published numbers exactly. It uses arbitrary-precision integers so the bijection stays exact at any depth.

Status: conditionally proved. T7 is a standard property of balanced positional notation. The claim that knowledge positions are routes is the design assumption; the claim that routes are integers is arithmetic.

4. Why Every Address Is Permanent

Four properties follow from T7 without further assumption. They are the reason this design can carry the world.

Prefix invariance. Leading zeros do not change the value. A route of depth 3 and the same route padded to depth 20 are the same integer. So a coarse description and a fine description of the same thing are not two records to reconcile. They are one number.

Truncation is rounding. Drop the turns below place n. What remains differs from the full value by less than half of 3^n. That is the exact statement: the dropped tail is at most $1 + 3 + 9 + \ldots + 3^{n-1}$, which equals $(3^n - 1)/2$, just under half of the last place kept.

So looking at something coarsely is not a different query against a different index. It is the same address with fewer digits, and the error is bounded and known in advance.

Mirror is negation. Turn every +1 into −1 and every −1 into +1, and the integer becomes its negative. The opposite of a thing is one arithmetic operation away. No thesaurus of antonyms. No trained model of contrast.

Distance is subtraction. How far apart are two items? Subtract their addresses. The result is a number, not a similarity score between zero and one whose meaning depends on the model that produced it.

Together these four give the property that no existing knowledge system has: **the index never needs rebuilding.** Add a million items, add a discipline that did not exist, add a century of new work — every address already granted stays exactly what it was. This is proved in §12.

5. How Deep Must the World Go

Depth n gives 3^n distinct routes. The largest value representable at depth n is $(3^n - 1)/2$. That ceiling runs 1, 4, 13, 40, 121, 364, 1093, and onward.

Capacity triples per turn added. So the question "how deep must the world archive be?" has a short arithmetic answer.

Nineteen turns. 3^{19} = 1,162,261,467. Just over one billion addresses. That covers every book ever printed, every article in every encyclopedia, every catalogued artefact in every museum, several times over. Estimates of distinct published books run near 150 million. Nineteen turns leaves a factor of seven in hand.

Twenty-six turns. 3^{26} = 2,541,865,828,329. Two and a half trillion. That covers every document ever written, including the uncatalogued — letters, ledgers, inscriptions, notes.

Thirty-two turns. 3^{32} = 1,853,020,188,851,841. Roughly two quadrillion. That covers every sentence in every document.

Forty-two turns. $3^{42} \approx 1.09 \times 10^{20}$. Take one hundred billion humans who have ever lived. Give each one a billion distinguishable moments of experience. That is 10^{20}. Forty-two turns addresses every one of them, individually.

Now the storage arithmetic. One turn is one trit. One trit carries $\log_2 3 = 1.585$ bits. Forty-two turns is 66.6 bits, which fits in nine bytes.

Nine bytes to name any moment in the history of human experience.

But the archive does not store those addresses. It computes them. Storage is needed only where something actually sits. The address space is free; only occupancy costs.

Status: predicted. The item counts are published estimates and are used here only to fix the order of magnitude. The arithmetic on 3^n is exact.

6. The Four Operations That Replace Search

A conventional knowledge system needs an index, a query language, a ranking function, and a relevance model. This design needs four integer operations.

Question	Operation	Cost
Are these the same thing?	equality of integers	one comparison
Same thing at another scale?	is one a prefix of the other	length of the shorter route
What is the opposite?	negate	one pass over the turns
How far apart?	subtract	one subtraction
Coarser view?	truncate	drop turns

Every cost in that table is proportional to the number of turns, not to the size of the corpus. A billion-item archive and a hundred-item archive answer these questions at the same speed. That is not an optimisation. It is what happens when position replaces similarity.

Note what is absent. No nearest neighbour. No vector distance. No ranking. No relevance. Those exist because embeddings give a position whose meaning depends on everything else in the space. A computed address does not.

7. The Encoder

The encoder is the only place where language enters. Its job is narrow: produce the turns.

It asks one question, n times. The question is the single rule from §2. Above, at, or below the reference at this ring?

Any agent can be the encoder. A human with the rule written down. A language model with the rule as its prompt. An instrument measuring a physical quantity against a threshold. They are interchangeable, because the rule is not about language. It is about relation to a reference.

This is where the design becomes world-scale, and it is worth stating flatly. There is no registry. There is no naming authority. There is no consensus protocol. Two encoders agree because they applied the same rule, not because a body ratified their agreement. Determinism does the work that institutions do elsewhere, and it does it at zero coordination cost.

The encoder is the weak point of the design, and the design says so. If the same item yields different routes on different days, everything above collapses. That is test W1 in §16, and it is the first thing to run.

The remedy when W1 fails is fixed in advance: sharpen the reference at the failing ring. Never widen the model. A vague rule produces a vague code, and a larger encoder makes vagueness more fluent, not more exact.

8. The Gate

Not everything offered may occupy an address. The gate decides.

Content proposed for address a is re-encoded. That yields an address a'.

- If a' equals a, admit.
- If a' is a prefix of a, admit and mark as coarser.
- Otherwise, refuse.

This is the nilpotent gate already running in SWARP: `propose()` with residue below epsilon. Here the residue is exact, because both sides are integers.

The gate is what replaces editorial authority. There is no board deciding what belongs in the world archive. What closes, enters. What does not close, does not. Human and machine pass through the identical gate, which means the archive cannot be corrupted by scale — a million machine-generated proposals face the same test as one human one.

And refusal is productive. When content fails at a, the first candidate to try is $-a$, the mirror. A statement that does not close where it was offered very often closes at its opposite.

9. The Store

The store is a sparse map from integer to content. Nothing else.

Most addresses are empty. That is not a defect. Empty means: the net has no closure there, at this depth, yet.

Retrieval handles emptiness without searching. Ask for address a. If occupied, return it. If empty, drop one turn and ask again. Repeat until occupied. Address 0 is always occupied.

The answer that comes back is coarser than the question, never different from it. The error is bounded by §4: less than half the place value of the last turn kept. The caller is told which depth answered.

This replaces "no results found" with a graded, honest answer, and it replaces guessing with rounding. A language model asked something outside its knowledge produces a fluent sentence. This produces a coarser true one, and says so.

The empty regions are themselves a result. A map of unoccupied addresses at depth 5 — 121 positions — is a map of what the model does not yet say. That list is short enough to read in an afternoon and is the natural work queue.

10. Federation: Why This Scales to the World

Two archives are built independently. One in Leiden, one anywhere else. Different corpora, different languages, different centuries of source material.

Merging them is set union on integers.

There is no schema mapping. There is no ontology alignment. There is no crosswalk table. Those exist because two systems that were granted their addresses by different authorities have no common ground. Two systems that computed their addresses from the same rule share every address by construction.

Conflicts are exactly one case: the same address, occupied by different content. That is not a technical failure. It is a substantive disagreement, and the address localises it precisely. Two encoders read the same position in the net and put different things there. Now the disagreement can be examined instead of averaged away.

This is the property that makes the design world-scale rather than merely large. It does not need everyone to join one system. It needs everyone to use one rule, and then the systems are already joined.

11. Past Knowledge

The rule is not about writing. It is about relation to a reference. So it reaches material that no text-based index reaches.

Oral knowledge. A tradition that transmits a relation — this stands above that, this comes before that, this belongs under that — is already producing turns. The turns were never written down, but they were carried.

Symbolic and ideographic material. A figure holding an object, a labyrinth path, a sequence of glyphs: each encodes relative position. The single rule reads those directly, without the intermediate step of translating them into a modern proposition first. That translation step is where most of the meaning of pre-literate material is currently lost.

Measured material. A sediment layer, a tree ring, an isotope ratio against a threshold: above, at, below. The rule accepts instruments as encoders on the same footing as people.

The consequence is worth stating plainly. A myth, a measurement, and a modern paper can occupy addresses in the same space and be compared by subtraction. Not because they are the same kind of statement, but because the code records position in the net, and position is what they all have.

Status: corresponded. This section claims a correspondence, not a proof. The test is whether independent readers of the same pre-literate material produce the same turns — W1 applied to old sources.

12. Future Knowledge

This is the section the whole design exists for.

Every knowledge system built so far fails on growth. Add material that the classification did not anticipate, and you must extend the classification. Extend it, and previously assigned numbers change meaning. That is why library classifications are revised, why URLs rot, why an embedding space must be rebuilt when the model is retrained.

Here, growth costs nothing already granted. The proof is one line.

An address at depth n is the truncation of the same item's address at depth n+k. Truncation drops trailing turns and leaves the leading ones untouched. So refining an address never alters it. It only extends it.

Therefore:

- New material at greater depth leaves all existing addresses exactly as they were.
- Capacity triples per turn added. Running out of room costs one trit.
- No item ever needs re-coding. There is no re-indexing event, ever.

This is monotone refinement, and it is what "all knowledge of the future" means operationally. Not that future knowledge is predicted. That future knowledge fits without breaking anything.

A worked case. Suppose the archive is built at depth 26 — two and a half trillion addresses, every document ever written. A field opens that did not exist. Its material needs distinctions finer than depth 26 supports. Add six turns. Capacity becomes $3^{32} \approx 1.85 \times 10^{15}$, a factor of 729. Every address assigned at depth 26 is still the exact prefix of its refinement. Nothing is migrated. Nothing is versioned. Nothing is deprecated.

Status: conditionally proved. It follows from prefix invariance, which follows from positional notation. It holds as long as the single rule of §2 is what is being applied at every ring. It fails immediately if different rings use unrelated questions.

13. The Renderer

The last layer turns an address and its content into sentences. A language model does this well and should do nothing else.

It adds no knowledge. Everything it adds beyond the retrieved content fails the gate in §8 and is discarded.

This is the architecture that the cheapest current systems are converging on from the other direction: knowledge in the engine, language at the edge. Here it is not a cost optimisation. It is a structural requirement. Language is how the net is spoken about, not where the net is.

14. The Cost

Two arithmetics, side by side.

A current language model. The working state is the context window. Take 200,000 tokens over a vocabulary of about 130,000 symbols. Each token carries $\log_2(130{,}000) = 17.0$ bits. The whole state is $200{,}000 \times 17.0 = 3.40 \times 10^6$ bits, which is 2.15×10^6 trits. Each step emits one token: 17.0 bits, or 10.7 trits.

To produce those 10.7 trits, the weights are traversed. At 4×10^{11} parameters stored in 8 bits each, that is 3.2×10^{12} bits, or 2.0×10^{12} trits. The parameter count is not published; the order of magnitude is what matters.

Ratio of what moves to what comes out: about 10^{11}.

This design. An item's address at world depth is 42 trits. It is computed, not looked up. Retrieval, comparison, mirroring and coarsening are integer operations proportional to 42.

The difference is not that this design uses fewer trits. It is that its trits are **defined**. Each one carries one stated distinction: above, at, below. Of any turn you can say what it means. In a language model, a trit carries an unnamed portion of an unnamed distinction — there is no turn, only a position.

Two million undefined trits give no address. So they give no mirror, no scale prefix, no distance as a number. Forty-two defined trits give all of them.

This is the detour, in figures. The chain from question to answer is long, and each link costs. The market prices only the endpoints.

15. What Is Not Solved

O2. The rewrite dynamics.

The kernel addresses. It does not move. Without a rewrite rule, this design is an archive with an excellent index and no motor. It can say where everything is. It cannot say what happens next.

The requirement is sharp: an operator R on routes that is its own fixed point. Not "the action that minimises the net" — the rewriting that reproduces itself.

Two things are now known about it, and both come from the design above.

The trivial fixed point is already in hand. Truncation is a rewrite: route → prefix. Truncating a truncated route truncates again. It is idempotent, therefore a fixed point. But it loses depth. So the non-trivial rewrite is characterised by exactly what distinguishes it from truncation: **it must preserve depth and still close.** That is a sharper statement of O2 than the one in the foundational paper.

The search space is small enough to exhaust. At depth 5 there are $3^5 = 243$ routes and a ceiling of 121. Every candidate depth-preserving rewrite on 243 routes can be enumerated and checked for fixed points by brute force. This belongs on paper and in a small exhaustive run, not in the application.

Status: open. No claim is made here that O2 is solved.

16. Five Tests

Each test states what would settle it against the design. Thresholds are preregistration choices and are revisable only before data contact.

W1 — Reproducibility of the encoder. Two independent encoders code the same 200 items to depth 5. Measure exact agreement per ring. *Fails if:* agreement at ring 1 falls below 0.90, or the first ring below 0.75 occurs at ring 3 or shallower. *Remedy on failure:* sharpen the reference at the failing ring. Not a larger encoder.

W2 — Prefix invariance in practice. Code 200 items at depth 3 and, separately and later, at depth 5. *Fails if:* the depth-3 address is not the truncation of the depth-5 address in more than 10% of items. This is the direct test of the single-rule assumption of §2; unrelated axes fail it immediately.

W3 — The mirror. For 50 items, negate the address and retrieve. Ask readers, blind, to name the opposite of the original item. *Fails if:* fewer than half the retrieved items are recognised as the named opposite.

W4 — Federation. Two people build stores of 200 items independently from disjoint corpora, same rule. Merge. *Fails if:* collisions occur that both parties judge to be false collisions — same address, unrelated content — in more than 10% of collisions.

W5 — Growth. Extend a depth-5 store to depth 8, a capacity factor of 27. *Fails if:* any depth-5 address changes value.

W5 should not fail. If it does, the implementation has departed from positional notation, and the fault is in the code, not the design.

17. Status Ledger

Claim	Status
Single recursive distinction as the only rule	assumed
Route ↔ integer bijection (T7)	conditionally proved
Prefix invariance, truncation-as-rounding, mirror-as-negation	conditionally proved (from T7)
Four operations replace search	conditionally proved
Growth without re-indexing (§12)	conditionally proved (from prefix
Federation as set union	conditionally proved (given one rule)
Encoder reproducibility	predicted — W1
Depth estimates for world coverage (19, 26, 32, 42)	predicted — corpus estimates, exact arithmetic
Pre-literate and measured material as turns	corresponded
Cost ratio ≈ 10^{11}	predicted — parameter count not

Rewrite dynamics (O2)	open

18. Build Order

1. **Fix the rule.** One sentence, five applications. Nothing else is written until this sentence is fixed.
2. **Run W1.** Two hundred items, two encoders. This decides whether the design is buildable at all.
3. **Run W2.** Same corpus at two depths. This decides whether the single rule is really single.
4. **Fill depth 5.** 121 addresses from the existing corpus. Publish the empty ones as the work queue.
5. **Gate and store.** Four endpoints: encode, resolve, verify, render. The kernel is untouched.
6. **O2 exhaustively at depth 5.** 243 routes, on paper and in a small program.
7. **Federation trial.** W4, with one other person.

Steps 1 to 3 are the decision. Steps 4 and 5 are a week. Step 6 is the research.

Annotated References

Knuth, D. E. *The Art of Computer Programming*, Vol. 2: Seminumerical Algorithms. Section 4.1, positional number systems. Why read? The standard treatment of balanced ternary, including the property used throughout §4: truncation in this notation is rounding to nearest, with no separate rounding step. Knuth calls it the prettiest of the number systems and shows why. *Reading advice:* section 4.1 alone is enough; skip the rest of the chapter.

Brusentsov, N. P., and the Setun computer, Moscow State University, 1958. Why read? The only production computer ever built on balanced ternary. About fifty were made. It demonstrates that this notation is an engineering option and not a curiosity, and the reasons it lost — component supply, not performance — are instructive.

Leibniz, G. W. *Explication de l'Arithmétique Binaire*, 1703. Why read? The origin of the idea that a positional notation is not a convenience but a claim about how distinctions compose. Leibniz took the encoding of all knowledge as a serious programme, not a metaphor, and this design is in that line.

Otlet, P., and La Fontaine, H. The Mundaneum and the Universal Decimal Classification, from 1895. Why read? The most serious previous attempt at a world archive with universal coding. It failed on exactly the point §12 addresses: the classification had to be maintained and revised by an authority, so the codes were granted and could be withdrawn. *Reading advice:* read it as the control case for this design.

Ranganathan, S. R. *Colon Classification*, 1933. Why read? Faceted classification is the closest prior art to recursive distinction: build the code from components rather than pick it from a list. Comparing it with §2 shows precisely what the single-rule constraint adds — Ranganathan's facets are unrelated axes, so his codes cannot be truncated or mirrored.

Bush, V. "As We May Think," *The Atlantic*, July 1945. Why read? The memex essay states the problem this design solves — retrieval by association rather than by shelf position — and proposes trails as the solution. Worth reading for what it did not have: a computable address.

Nelson, T. *Literary Machines*, 1981. Why read? Xanadu's central demand was permanent addressing: a reference that never breaks. Nelson diagnosed link rot decades before it happened. He solved it with a central registry, which is the assumption §7 removes.

Merkle, R. Hash-based tree structures, 1979; and content-addressable storage generally. Why read? The nearest working relative of this design: an address computed from content rather than assigned to it. The instructive difference is that a hash destroys structure by design — nothing can be truncated, mirrored, or subtracted. That is what a hash is for, and it is what makes it useless for knowledge.

Geohash (Niemeyer, 2008), and quadtree spatial indexing generally. Why read? A working, deployed system with the exact prefix property of §4: drop characters and the address names a larger region containing the smaller one. Proof that prefix-invariant addressing scales in production. This design generalises it from two spatial dimensions to arbitrary distinctions.

Shannon, C. E. "A Mathematical Theory of Communication," 1948. Why read? Fixes the accounting used in §5 and §14: how many bits a distinction carries, and therefore how many trits a position costs. The conversion $\log_2 3 = 1.585$ comes from here.

Kolmogorov, A. N. (1965); Solomonoff, R. (1964); Chaitin, G. (1966), on descriptive complexity. Why read? Sets the floor. The shortest description of an item is a property of the item, not of the encoder. This design claims a well-behaved address, not a minimal one, and this literature is where the difference is made precise.

Borges, J. L. "The Library of Babel" (1941) and "Funes the Memorious" (1942). Why read? The two failure modes of a total archive, stated better than any technical source. Babel is a complete address space with no way to compute an address. Funes is total retention with no coarsening. §5 and §9 are the answers to each. *Reading advice:* both are short; read them together.

Peirce, C. S., on triadic relations and the irreducibility of the third. Why read? The argument that three is where relation begins and that a triadic relation does not decompose into dyads. The single rule of §2 has three outcomes; Peirce is the strongest existing case that this is structural rather than a design taste.

Companion pieces: The Maze as Machine (18 August 2026); MAZE — The Human Correspondence of the Vacuum.Net Theory. The kernel is `server/maze-kernel/maze.ts` *in the SWARP repository.*