

Alioth Model Theory White Paper

Enterprise Data Models from the Perspective of Commutative Ontology: Mathematical Foundations, Structural Invariants, and Machine Verification

Version v2.1 · October 2026 · Hangzhou Yuqi Technology Co., Ltd.

Corresponding implementation: Alioth Model v10.0.44 (release verified 2026-10-19) · AliothStudio Platform

Contents

- Abstract
- Problem Domain and Design Goals
- Symmetric Monoidal Category and Symmetry
- Ontology Space: Coordinate Product and Orthogonality
- The Time Dimension: Total Order of States and Mirror Involution
- Five-Domain Completeness
- Structural Relation Algebra
- Scalar Externalization and Conservation Measure
- Model-Theoretic Perspective: Theory and Multi-Namespace Models
- Machine Verification and the Engineering Loop
- Conclusion

1. Abstract

Alioth is an enterprise data model whose physical substrate is PostgreSQL table inheritance and whose semantic framework is Commutative Ontology. This white paper sets out its mathematical foundations: economic behavior is formalized as morphisms in a [symmetric monoidal category](#), reversible transactions

constitute a **groupoid** equipped with an involutive symmetry functor; every business entity occupies a unique coordinate in the orthogonal product space formed by Scene, Factor, and Function, while the primary state chain supplies a fourth dimension of discrete logical time. The structural properties of the model — symmetric conservation, coordinate orthogonality, five-domain coverage, acyclic relations, measure additivity — are expressed as twelve executable invariants, each anchored to an engineering gate, and five of the core metatheorems have already been machine-verified by the Lean 4 theorem prover. The same theory has been inductively validated in production deployments across six industry namespaces.

2. Problem Domain and Design Goals

The classic predicament of enterprise data models is this: business semantics are scattered across the empirical design of hundreds of tables, with no objective criterion for deciding "what counts as a legitimate business object"; onboarding an additional business domain is done by copy-and-paste, so the model cannot be reused; and correctness depends on configuration discipline rather than structural guarantees. Alioth's design goals are:

Goal	Meaning
Semantic coordinateization	Any business entity can be located by a unique coordinate, and the coordinate itself carries domain semantics
Decidable structural invariants	Every model property is a mechanically checkable predicate, not a documentary convention
Theory defined once, reused across domains	Onboarding a new business domain leaves the model structure unchanged (zero DDL) and merely assigns an interpretation to coordinates
Machine-verifiable theory	Core metatheorems can be independently re-checked by a theorem prover

3. Symmetric Monoidal Category and Symmetry

3.1 Symmetric Monoidal Category

Taking economic agents as objects and transactions as morphisms yields a category $\mathcal{E}$: composition of transactions is a transaction chain, and the identity element is the null transaction. Note that $\mathcal{E}$ is a category,

not a group — a general transaction is irreversible and cannot be formulated as a "group of economic agents" (the morphism set between objects carries no total binary operation).

3.2 Symmetry: An Involutive Functor on the Core Groupoid

The reversible core $\text{core}(\mathcal{E})$ of $\mathcal{E}$ is the groupoid formed by all reversible transactions. The rigorous statement of the symmetry principle is that there exists an involutive endofunctor

$$S : \text{core}(\mathcal{E}) \rightarrow \text{core}(\mathcal{E})^{\text{op}}, S^2 = \text{id}$$

that is, every reversible transaction has a mirror inverse transaction, and the mirror of the mirror is itself. Involutivity guarantees that the mirror relation has no directional bias — this is the precise meaning of "every transaction has a symmetric inverse transaction". The function-level core (an involution is a self-inverse bijection) has already been machine-proved in Lean (five theorems including `Symmetry.Involutive.injective_surjective`).

3.3 Conservation Law

Given an endowment assignment and an equity valuation, the conservation law requires that the equities exchanged by the two mirror sides be of equal value: $v(M) = v(D)$, so that the total system endowment is invariant under any composite of reversible transactions (Lean theorem `Conservation.endowment_conservation`). At the implementation layer, conservation is not a declaration but a witness on every entry: the voucher base class backfills the opening balance and the closing balance for each voucher, making "prior state + net change = posterior state" mechanically checkable on every row; goods in/out vouchers and financial posting vouchers inherit the same base class — physical conservation and financial conservation are isomorphic.

4. Ontology Space: Coordinate Product and Orthogonality

4.1 Life Entities and the Coordinate Functor

All business entities with a lifecycle are inheritance descendants of the root table `zc_id_lifecycle` (the current release v10.0.44 contains 1022 isahl tables and 1204 inheritance edges, of which 374 are lifecycle leaf tables; the three-dimensional coordinate set currently comprises 645 entries — 109 scenes + 118 factors + 418 functions). Three foreign keys constitute the coordinate functor:

```
 $\chi = (\text{dk\_scene}, \text{dk\_factor}, \text{dk\_function}) : \text{Ob}(\mathcal{L}) \rightarrow S \times F \times U$   
S = business scene (Where & When)  F = business factor (Who & What)  U = business  
function (How)
```

The three dimensions constitute a **product category** — each takes values and evolves independently, which is the rigorous meaning of orthogonality. Any lifecycle entity occupies a unique point in the ontology space. Within a dimension, category foreign keys organize a forest partial order; the dimension code `code = category code || symbol[serial]` (42-character set) is the hierarchical address within that forest, allocated automatically by a trigger on write.

4.2 Quaternion Snapshot

State is not carried by entity columns but attached through relation tables. The primary state chain is a totally ordered set (one-way progression, e.g. born→dead) and constitutes discrete logical time; `status_date` projects it onto physical time. The quaternion snapshot of an entity at state node `t` is:

```
 $\Sigma(e, t) = (\text{scene}, \text{factor}, \text{function}, \text{status}) \in S \times F \times U \times St$ 
```

All four dimensions are mutually orthogonal; three are spatial dimensions and status is the time dimension.

5. The Time Dimension: Total Order of States and Mirror Involution

The set of time slices is divided in two by "now": those that have occurred are called events (past facts), and those that have not are called tasks (future to-dos). The two are mirror images of the same time-slice concept on either side of now; completing a task flips the slice (task → event), and the action trace is recorded by the operation carrier. A plan unfolds towards both sides of the time axis through two symmetric bridges (the plan→event group and the plan→task group), organizing cross-time intent around a single anchor; progress is a scalar reference rather than an entity, and the Gantt chart is a natural projection of the plan system rather than an independent model.

6. Five-Domain Completeness

Domain membership is derived through the category chain of the factor dimension:

```

domain = a_type_ ◦ ck_category ◦ dk_factor : Ob( $\mathcal{L}$ ) → {L, C, G, P, F, ~}
L Person (subject)  C Container (medium·container)  G Goods (object)  P Place
(medium·site)  F Finance (process and information)  ~ wildcard

```

A snapshot is complete if and only if the image of the domain map covers all five domains: $\text{domain}(\Sigma) \supseteq \{L, C, G, P, F\}$ — symmetry breaking is not permitted. Completeness is a **decidable predicate** (a finite set-cover decision, which the Lean instance discharges mechanically), verified snapshot by snapshot by the inference engine at generation time; missing domains are filled in by upward closure along the inheritance partial order, and the termination and idempotence of completion have been machine-proved (`Completion.isAncestor_height_lt, closure_idempotent`).

A note on theoretical honesty: five-domain coverage, taken as an assertion about real exchange, is a postulate rather than a theorem — its status is that of a conservation law in physics, supported by falsifiability and by inductive evidence from multiple namespaces (see Section 9). This model does not disguise it as a provable proposition.

7. Structural Relation Algebra

7.1 Inheritance Partial Order

The set of tables together with the inheritance relation forms a directed acyclic graph partial order $(\mathcal{T}, \leq)$: the root is the abstract object table, and the largest business subtree is the lifecycle chain; leaf tables are the minimal elements of the partial order and the only legitimate landing point for data writes; multiple inheritance makes this structure a DAG rather than a tree. Model extension can only be growth of the lower cone of the partial order (registering a subtype and attaching inheritance); duplicating tables independently is forbidden — the structure therefore retains a single source of truth.

7.2 Column Families: Typed Projections

Entity columns are grouped into families by prefix, each family being a typed projection onto a particular inheritance hierarchy: `dk_*` (ontology coordinates), `fk_*` (life-entity associations), `qk_*` (value-space pointers), `sk_*` (units), `ck_*` (categories), `tk_*` (tags), `lk_*` (levels), `tpl_id` (instance→exemplar bridge). All of them are unified as `bigint` — a reference column is a pointer within one type universe, and the type distinction is carried by the prefix (target category).

7.3 Relation Families: Functional Relations and Spans

`r_*` relation tables (unique `ref_right`) encode functional relations (1:N, each child belonging to exactly one parent); `rr_*` bridge tables encode spans (arbitrary M:N relations); `fk_*` columns encode N:1 functions. Relation composition is morphism composition; nested/hierarchical relations require the DAG invariant (the ancestor chain is acyclic, machine-proved as `isAncestor_irrefl`), and rollup is a summation functional over the transitive closure.

7.4 Six Quadrants: Phase $\times$ Abstraction Level

The functional code prefix carries the product structure $\{\text{creative, design, implementation}\} \times \{\text{exemplar, instance}\}$ (six quadrants); entity classification is computed mechanically from the code prefix through a derived homomorphism — only these six pairs exist and hand-writing them is forbidden. The exemplar/instance axis is also the ownership axis: exemplars are organization-level public assets (non-deletable once referenced, versions non-erasable), while instances are private outputs (freely managed by their owner); only two one-way legitimate cross-quadrant transitions exist, private $\rightarrow$ public.

7.5 Derived-Concept Adjudication: Samples Do Not Enter the Structure

Ontology adjudication (ALIOTH_ONTOLOGY_SPEC § 8.5.11, October 2026): a sample is a derived concept of batch/branch — product entities are forbidden from adding new sample identifier columns, and creating product sub-tables for samples is forbidden; sample semantics are uniformly expressed through derivation from batch/branch records and category/relation bridges. This is the latest application of the "single source of truth for structure" principle: any semantics derivable from existing coordinates and relations may not solidify into a new structural point. In the same month, the goods concept family was re-divided at the model layer (4 sets of identifier renames), and the platform runtime, seeds, and derived artifacts converged in the same batch — a complete demonstration of the "model as the center, code as the artifact" propagation mechanism.

8. Scalar Externalization and Conservation Measure

Magnitudes are not embedded in entity rows but externalized into a scalar inheritance hierarchy (value + unit + precision): the entity holds a pointer, and a given magnitude is stored exactly once in the whole database — externalizing value objects makes "equal value" a mechanical decision of "same pointer", so arithmetic on pointer columns is therefore meaningless and must be resolved first. Association relations carry validity periods, and statistics at time t evaluate only effective relations (spatiotemporal association). Inventory is not an entity attribute but an additive signed measure over voucher facts:

$$\mu_t(P, S) = \Sigma \text{ voucher net change (income - outgo)} + \Sigma \text{ nested sub-store rollup}$$

The measure satisfies additivity and signedness; the materialized value is a cache of the measure, maintained incrementally through a single entry point. Stocktaking is an atomic measure-correction event: the difference between the physical count and the materialized value (overage/shortage) is absorbed into the measure through a calibration voucher rather than concealed.

9. Model-Theoretic Perspective: Theory and Multi-Namespace Models

To summarize in the language of model theory: the model structure (the frozen inheritance partial order + dimension codes + invariant set) is the **theory** $\mathbb{T}$; each industry-namespace deployment is a **model** $\mathcal{M} \models \mathbb{T}$ of $\mathbb{T}$ — business semantics fall into local interpretations of coordinate codes, with zero structural divergence. As of October 2026, the same theory already carries six namespaces:

Namespace	Apps	Modules	Blocks	Services	Business domain
Alioth	3	11	50	20	The platform itself
WZ	4	16	90	16	Logistics external collaboration
AVIC-CAASEC	2	5	65	11	Research project approval
SE	1	6	32	6	Budget management
Cosmic-Tools	4	23	56	28	Tools / WMS / ERP suite
MAT	1	0	0	1	New domain onboarding (in progress)
Total	15	61	293	82	—

Onboarding a new namespace (such as SE and MAT) is accomplished through "inheritance localization + coordinate binding + configuration assembly" — zero DDL, zero model patches. The theory's expressive power over new business domains has not been broken, and this is the inductive evidence for coordinate orthogonality and five-domain completeness.

10. Machine Verification and the Engineering Loop

The value of the theory is judged by its executability. The twelve mathematical invariants (conservation, orthogonality, coverage, acyclicity, monotonicity, six-quadrant consistency, value externalization, structure preservation, non-deletability of the public domain, inheritance purity, time mirroring, measure additivity) map one by one onto the engineering gates in force (current registered gates: 229 — 137 at the commit stage / 92 at the push stage); five of the core metatheorems are machine-verified by the Lean 4 theorem prover (green builds since 2026-09-18, zero external-library dependencies, zero proof placeholders):

Invariant	Machine proof
Value conservation (T3)	Conservation.endowment_conservation
5-domain coverage (decidability)	Domain.complete_iff_image_covers
Acyclic relations (T1 corollary)	Completion.isAncestor_irrefl
Inheritance purity / completion soundness (T4)	Completion.closure_superset / closure_idempotent
Time mirroring (T2, involution layer)	Symmetry.Involutive.injective_surjective

This yields a three-layer loop: mathematical statement (this white paper) → engineering gates (continuous enforcement of computable predicates) → machine proofs (independent re-checking by a theorem prover).

11. Conclusion

Alioth's fundamental claim is that the correctness of an enterprise data model should rest on structure and mathematics, not on discipline and experience. Coordinateized semantics, decidable invariants, a theory defined once and reused across domains, and metatheorems that can be machine-verified together form the complete chain of evidence for this claim. We welcome scrutiny, challenges, and improvements to this theory from academia and industry.