Argument Ontology Specification (AOS-2)

A House Specification for Structural Theory Documents

Abstract

This document specifies a reusable format for constructing an *argument ontology document*: a companion artifact whose task is to represent a theory-object explicitly and non-narratively. AOS-2 is a revision of earlier argument ontology practice. It is designed for projects in which theories are coupled, revision is normal, and prose alone is too lossy a carrier of structural identity. The goal of an argument ontology document is not readability, persuasion, or narrative flow. The goal is recoverable structure. AOS-2 therefore requires explicit treatment of theory identity, object type, kernel, interface, primitives, distinctions, claims, dependencies, burdens, supports, objections, scope, residues, revision status, and stack or canon relations. It assumes that a theory-object is not identical with any one prose rendering of it. A prose paper, formal sketch, and ontology document may all render the same underlying theory-object under different presentation constraints.

1. Purpose

AOS-2 exists to solve a recurring problem in theory work: a theory may be discussed, expanded, defended, or criticized without its internal structure ever becoming fully explicit. When this happens, the theory risks being carried mainly by prose sequence, rhetorical emphasis, local memory, or informal compression. That may be enough for early exploration, but it becomes unstable once the project grows, couples to other frameworks, or begins to drift across revisions.

The purpose of an argument ontology document is therefore to state, in non-narrative form:

- what kind of theory-object is being represented,
- what its kernel is,
- what it exports through a stable interface,
- what primitives, distinctions, and claims it uses,
- what depends on what,
- what burdens it incurs,
- what supports it,
- what objections target it,

- what belongs to exposition rather than commitment,
- what it inherits from upstream objects,
- what it enables downstream,
- what remains unresolved,
- and what kinds of change do or do not require broader propagation.

2. What This Document Is For

An argument ontology document is *not* a replacement for a theory paper. It is a companion artifact with a different function.

The prose paper is for:

- motivating the problem,
- presenting background,
- explaining the view,
- guiding the reader,
- and carrying argumentative transition.

The argument ontology document is for:

- structural recoverability,
- explicit commitment tracking,
- dependency visibility,
- burden discipline,
- revision control,
- stack or canon navigation,
- and continuity across drift.

3. Core Principle

The core principle of AOS-2 is:

> A theory should be representable as an explicit ontology of objects, distinctions, claims, interfaces, dependencies, burdens, and residues rather than only as persuasive prose.

4. Upstream Assumption

AOS-2 assumes that a *theory-object* can be distinguished from its renderings.

A theory-object is:

- a bounded explanatory object,
- organized by primitives, distinctions, claims, and burdens,
- and capable of being rendered in more than one artifact.

A rendering may be:

- a prose paper,
- an argument ontology document,
- a formal sketch,
- a diagram,
- a bibliography skeleton,
- or another canonically recognized artifact.

AOS-2 therefore treats the ontology document as one rendering of a theory-object, not as the theory-object itself.

5. Key Distinctions

Every document written under AOS-2 must explicitly distinguish the following.

5.1. Theory-Object vs Rendering

The document must distinguish:

- the bounded theory-object,
- from the prose, formal, or diagrammatic artifacts that render it.

5.2. Kernel vs Interface

The document must distinguish:

- the theory's kernel: what it is most centrally about,
- from its interface: what it exports stably enough for other frameworks to inherit or rely on.

5.3. Exposition vs Commitment

The document must distinguish:

- motivating prose,
- rhetorical packaging,
- illustrations,
- and transitional material,

from actual theoretical commitment.

5.4. Presentation Order vs Theory Order

The document must distinguish:

- the order in which the paper presents things,
- from the order in which the theory is structurally organized.

5.5. Primitive vs Derived vs Borrowed

The document must distinguish:

- primitive objects the theory must posit,
- derived objects it can generate internally,
- and borrowed objects it inherits from elsewhere.

5.6. Internal Drift vs Interface Change

The document must distinguish:

- internal revision that does not alter the theory's exported identity,
- from interface change that should propagate to adjacent theory-objects.

5.7. Ontology vs Disclosure vs Normativity vs Formality

The document must classify its objects and claims by type. At minimum, it must distinguish:

- ontological,
- epistemic or disclosure-related,
- normative,
- formal,
- architectural,
- or mixed objects.

5.8. Upstream vs Downstream

The document must distinguish:

- what the theory inherits,
- from what it enables,
- and from what merely sits nearby without strong dependency.

6. Theory-Object Types

Every argument ontology document must classify the target theory-object using one or more of the following types.

- substrate theory
- ontological theory
- disclosure theory
- adequacy theory
- epistemic theory
- diagnostic theory
- normative theory
- design theory
- formal theory
- architectural theory
- mixed theory

This requirement exists because many papers slide among several theory types without explicitly saying which kind of object they primarily are.

7. Object Types to Be Represented

Every argument ontology document written under AOS-2 must make the following object types explicit where applicable.

7.1. Primitive Objects

Core objects the theory cannot do without.

7.2. Quasi-Primitives

Objects treated as working primitives pending later refinement or derivation.

7.3. Derived Objects

Objects generated by or dependent on more basic objects.

7.4. Claim Objects

Distinct theoretical commitments advanced by the theory.

7.5. Distinction Objects

Load-bearing conceptual cuts that organize the theory.

7.6. Dependency Objects

Relations of argumentative, definitional, or structural reliance.

7.7. Support Objects

Reasons, evidence clusters, examples, formal devices, or source families that carry argumentative weight.

7.8. Burden Objects

Required demonstrations, explanatory tasks, or pressure points the theory must address.

7.9. Scope Objects

Explicit statements of where the theory is intended to hold and where it is not.

7.10. Residue Objects

Unresolved, deferred, underdeveloped, or pressure-sensitive regions.

7.11. Interface Objects

Stable exports or inherited concepts through which the theory relates to the rest of the project or stack.

7.12. Exposition Objects

Material included for readability or motivation but not itself load-bearing.

8. Status Classes

Every major object and claim should carry a status.

- `provisional`
- `working`
- `stabilized`
- `pressure-sensitive`
- `deferred`
- `frozen`
- `borrowed`
- `superseded`

9. Freeze Levels

Every argument ontology document should state a freeze level for the theory-object as a whole, and may optionally assign one to major claims or interfaces.

- `none`
- `soft`
- `interface-frozen`
- `hard`

10. Required Document Structure

An argument ontology document written under AOS-2 must contain the following sections.

0. Metadata

Include:

- target theory-object name,
- object type,
- paper or rendering title if applicable,
- theory version,
- ontology document version,

- status,
- freeze level,
- drift class,
- domain,
- and whether the ontology is `minimal`, `expanded`, or `full`.

1. Ontology Purpose Statement

State what this ontology document is doing and what it is not doing.

Minimum requirement. The reader can tell that this is a structural rendering of a theory-object rather than a second prose draft.

2. Theory Identity

State:

- the main thesis in compact form,
- the theory-object type,
- the minimal theoretical ambition,
- the maximal intended scope,
- the core compression,
- and the theory's current status.

3. Kernel and Interface

This section is required.

State:

- the theory's kernel,
- its stable interface exports,
- what it imports or inherits,
- and what would count as an interface change rather than mere internal drift.

4. Primitive Ontology

List the theory's primitive, quasi-primitive, borrowed, and derived objects.

For each object, include:

- object ID,
- name,
- object type,
- classification (`primitive`, `quasi-primitive`, `borrowed`, `derived`),
- domain type (`ontological`, `epistemic`, `normative`, `formal`, `architectural`, `mixed`),
- compact definition,
- status,
- and what work the object does.

5. Distinction Ontology

List the theory's load-bearing distinctions.

For each distinction, include:

- distinction ID,
- poles of the distinction,
- function in the theory,
- what confusion it blocks,
- status,
- and which claims or objects rely on it.

6. Claim Ontology

List all major claims.

For each claim, include:

- claim ID,
- claim text,
- claim type (`main`, `subordinate`, `bridge`, `scope`, `negative`, `downstream`, `architectural`),
- domain type,
- strength (`core`, `important`, `optional`),
- status,
- and whether the claim belongs to kernel, interface, or extension.

7. Dependency Graph

State how objects, distinctions, claims, and interfaces depend on one another.

Each dependency entry should specify:

- source object,
- target object,
- dependency type,
- whether the dependency is necessary or supporting,
- and whether it is internal, upstream-inherited, or downstream-facing.

Dependency types may include:

- presupposes
- defines
- derives from
- constrains
- clarifies
- motivates
- supports
- excludes
- exports
- inherits

8. Theory Order

This section is required.

State:

- what is most primitive,
- what is introduced only because something more basic already holds,
- what belongs to the kernel,
- what belongs to interface,
- what belongs to extension,
- and what can be removed without collapse.

9. Burden Map

For each major claim or kernel object, state:

- what burden it incurs,
- whether the theory currently discharges that burden,
- whether the burden is only partially discharged,
- or whether it is deferred.

10. Support Map

For each major claim, state:

- what supports it,
- whether the support is conceptual, formal, empirical, source-based, synthetic, or illustrative,
- and whether the support is direct or indirect.

11. Objection Map

For each serious objection, state:

- objection ID,
- objection text,
- target claims or objects,
- reply type (`defense`, `qualification`, `revision`, `concession`),
- status of the objection,
- and remaining vulnerability.

12. Scope Map

State:

- what the theory covers,
- what it does not cover,
- what it conditions without explaining fully,
- what is upstream,
- what is downstream,
- and what is intentionally bracketed.

13. Exposition Map

This section is required.

List all material in the prose rendering that is:

- motivational,
- rhetorical,
- illustrative,
- compressive,
- or transitional.

For each entry, state:

- where it appears,
- what function it serves,
- what theoretical object it is helping present,
- and whether removing it would change commitment or only presentation.

14. Interface and Stack Relations

This section is required where the theory is part of a larger project.

State:

- what the theory inherits from upstream frameworks,
- what it exports to downstream frameworks,
- what neighboring frameworks it is related to but not strongly dependent on,
- and what kinds of change would require propagation.

15. Drift and Revision Statement

State:

- what parts of the theory are expected to drift,
- what parts are stabilized,
- what would count as internal revision only,
- what would count as interface change,
- and what would force reclassification or reopening.

16. Residue Statement

State what remains:

- unresolved,
- weakly specified,
- underargued,
- unformalized,
- or intentionally deferred.

17. Section-to-Ontology Map

Map the prose rendering's sections onto the theory-object's structural objects.

For each prose section, state:

- what theoretical objects it presents,
- what burdens it carries,
- whether it is kernel, interface, or extension work,
- and whether it is mainly expository or load-bearing.

11. Recommended Local Record Format

Primitive Object Record

```
\paragraph{Primitive Object}
ID: PO-01
Name: processual reality
Object type: primitive
Domain type: ontological
Status: stabilized
Definition: reality understood as organized transformation under constraint
Work performed: substrate of the theory
Used in claims: C-01, C-03
Distinctions linked: D-01, D-02
Kernel/interface role: kernel
Notes: central primitive
```

Claim Record

```
\paragraph{Claim}
ID: C-01
Type: main
```

```
Domain type: mixed
Strength: core
Status: argued
Kernel/interface role: kernel
Text: finite description is a conditioned act of disclosure under a cut
Depends on: PO-01, PO-03, D-01
Supported by: S-01, S-04
Burden: explain why partiality is structural rather than accidental
Objections: O-01, O-03
Scope: applies to finite description generally, not downstream governance
```

Distinction Record

```
\paragraph{Distinction}
ID: D-01
Poles: transparency / conditioned disclosure
Function: blocks naive representationalism
Status: stabilized
Used by claims: C-01, C-02, C-08
Blocks confusion: local success implies exhaustive access
```

Interface Record

```
\paragraph{Interface Export}
ID: IF-01
Name: scope
Type: interface object
Status: interface-frozen
Definition: adequacy is always indexed to conditions of use
Exported to: CEUC, MMUC
Propagation rule: changing this requires downstream review
```

Objection Record

```
\paragraph{Objection}
ID: O-02
Text: the theory collapses into relativism
Targets: C-01, C-05
Reply type: defense
Status: active but partly answered
Reply summary: partiality does not imply arbitrariness
Remaining vulnerability: requires stronger adequacy criteria
```

12. Minimal Acceptance Test

An argument ontology document satisfies AOS-2 only if a later reader can answer all of the following without reconstructing the theory from prose alone:

1. What kind of theory-object is this?
2. What is its kernel?
3. What does it export through a stable interface?
4. What are its primitive and derived objects?
5. What distinctions organize it?
6. What are its main claims?
7. What depends on what?
8. What burdens has it incurred and which has it discharged?
9. What objections target which parts?
10. What is exposition and what is commitment?
11. What is upstream, downstream, or merely adjacent?
12. What remains unresolved?
13. What kinds of revision are internal drift and what kinds are canon-breaking?

13. Recommended Use

AOS-2 is best used:

1. after or alongside a prose draft,
2. whenever the theory has become large enough that memory alone is unstable,
3. whenever the theory sits in a larger canon or stack,
4. and whenever coupled revision makes it important to separate kernel, interface, and residue.

A recommended workflow is:

1. build or revise the theory-object,
2. write or revise the prose rendering,
3. extract the object into AOS-2 format,
4. identify hidden commitments, unresolved burdens, and unstable interfaces,
5. and revise either the theory-object or the rendering accordingly.

14. Final Statement

A prose paper may still be the best vehicle for readability and persuasion. But if the theory cannot also be rendered as an explicit ontology of objects, distinctions, claims, interfaces, burdens, supports, objections, and residues, then too much of the work remains trapped in narrative sequence or tacit memory. AOS-2 is designed to prevent that. Its purpose is to make the theory-object itself structurally visible while still leaving room for finitude, selective stabilization, and revision.