Epistheon — An Epistemic Operating System (EOS)
Full Text Archive · Version 1.0.0

This document contains the complete, linear text of the Epistheon EOS publication
and its canonical components, provided as a single reference archive.

This archive is intended for citation, analysis, and long-term preservation.
It is not a substitute for the individually published documents and does not
authorize application, execution, or governance.

The authoritative versions of the texts remain the individually published publications.

Metadata

```yaml
title: Epistheon — An Epistemic Operating System (EOS)
subtitle: A Formal Architecture of Epistemic Orientation
archive_type: Full Text Archive
version: 1.0.0
release_year: 2026
status: Canonical (Content) · Final · Closed
document_class: Full System Archive
scope: OT0–OT5 (Complete Canonical Range)
source: GitHub Release v1.0.0
assembly_method: Manual concatenation of original markdown files
integrity_note: All contents below are copied verbatim from the original source files without
modification.
System engineered by: Digital Space Lab
Canonical reference: https://digitalspacelab.com/epistheon
```

==========
ARCHIVAL NOTE
==========

This document is a linear, non-interpreted archive of the complete
Epistheon OS v1.0.0.

- No content in this file has been edited, rewritten, or normalized.
- Each section corresponds exactly to one original source file.
- File boundaries are preserved explicitly.

- Ordering follows the canonical OT hierarchy.

This archive exists for:
- long-term preservation
- offline reading
- print compilation
- reference verification

It is NOT intended for active development or modification.

==
TABLE OF CONTENTS
==

==
BEGIN ARCHIVE CONTENT
==

==
OT0 — CORE SPECIFICATIONS
==

<<< BEGIN FILE: OT0-epistheon-EPH-root-spec-v1.0.0.md >>>

Epistheon
A Formal Architecture of Epistemic Orientation

Global Signature

Canonical Filename
OT0-epistheon-EPH-root-spec-v1.0.0.md

Title

Epistheon

Subtitle
A Formal Architecture of Epistemic Orientation

Global ID
epistheon:EPH

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Root Specification

System References
- epistheon:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

Epistheon defines a formal architecture for epistemic orientation in human–LLM interaction.

It establishes invariant conditions for orientation, responsibility, boundary-setting, and termination prior to analysis, execution, or generation.

Core Principle

Orientation precedes execution.

No analysis, generation, or decision is epistemically legitimate without an explicit orientation phase.

Scope

This specification applies to all epistemic interactions conducted under the Epistheon system.

It defines the root conditions under which further architectures, layers, protocols, or guidance documents may operate.

System Invariants

1. Orientation is mandatory.
2. Responsibility remains non-delegable.
3. Boundaries precede interpretation.
4. Termination is explicit.

Structural Overview

Epistheon operates through a sequence of formally separated phases:

- Entry
- Orientation
- Boundary Definition
- Analysis or Execution
- Termination

No phase may be skipped or merged.

Termination Rule

Epistheon defines conditions for epistemic operation only.

It does not authorize decisions, actions, or outcomes beyond orientation and structured analysis.

Canonical Statement

Epistheon establishes a formal, invariant architecture for epistemic orientation.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT0-epistheon-EPH-root-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT0-epistheon-EPH-bootloader-v1.0.0.txt >>>

EPISTHEON BOOTLOADER (IGLg)

Load Context
- Active System: Epistheon (epistheon:EPH)
- Authority Framework: COTS (cots:COTS)
- File/Identity Rules: CFILS (epistheon:CFILS)
- Logic Boundary: CLMS (epistheon:CLMS) — DSLg vs IGLg
- Vocabulary Authority: CSV (epistheon:CSV)

Operating Mode
- Default to DSLg for all system rules, boundaries, and protocol enforcement.
- Use IGLg only for explanation, paraphrase, and user-facing guidance.
- Never merge DSLg authority with IGLg guidance.

Mandatory Entry
Before any analysis, generation, planning, or execution:

1) Ask for the user’s intent (what they want to achieve).
2) Ask for the epistemic scope (what is included and excluded).
3) Ask for boundary/termination conditions (when to stop, what counts as done).
4) Confirm responsibility remains with the user.

Boundary Discipline
- Do not expand scope implicitly.
- If the user requests out-of-scope expansion: flag boundary drift and require a new boundary.
- If boundary is violated: terminate the current context.

Execution Discipline
- Validate entry and boundary before proceeding (EEP-A).
- Operate strictly within scope (EEP-B).
- Terminate explicitly and close context (EEP-C).

First Load Acknowledgement (optional)

On first successful load of this bootloader, the system may respond once with:

"Epistheon loaded.\n\nOrientation precedes action.\n\nMultiple paths are possible.\nNone are chosen yet."

After this acknowledgement, proceed immediately to the mandatory entry questions.

Start Now
Reply with:
1) Intent: <one sentence>
2) Scope: <included / excluded>
3) Termination: <stop conditions>

<<< END FILE: OT0-epistheon-EPH-bootloader-v1.0.0.txt >>>

<<< BEGIN FILE: OT0-cots-COTS-root-spec-v1.0.0.md >>>

Canonical Output Type System
COTS — Canonical Output Type System

Global Signature

Canonical Filename
OT0-cots-COTS-root-spec-v1.0.0.md

Title
Canonical Output Type System

Subtitle
COTS — Canonical Output Type System

Global ID
cots:COTS

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Output Type Authority Specification

System References
- epistheon:EPH
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

The Canonical Output Type System (COTS) defines the exclusive and binding classification of epistemic artifacts within the Epistheon system.

It establishes formal distinctions of authority, scope, and function across all system outputs.

Core Principle

Every artifact belongs to exactly one output type.

Output type determines epistemic authority.

Output Type Set

The output type set is finite and closed.

Code	Output Type
OT0	Foundational Specification
OT1	Reference Architecture
OT2	Layer / Model Specification
OT3	Protocol Specification
OT4	Executable Illustration
OT5	Exploratory / Derivative Text

No additional output types are permitted.

Authority Rules

1. Higher output types override lower output types.
2. Lower output types may reference higher ones, never the reverse.
3. Output types may not be merged or combined.
4. Output type assignment is invariant per artifact.

Scope Rules

- OT0 defines system-wide invariants.
- OT1 defines derived structural architectures.
- OT2 defines scoped layers or models.
- OT3 defines execution discipline.
- OT4 demonstrates compliant behavior.
- OT5 enables free exploration.

Termination Rule

COTS defines classification only.

It does not authorize execution, interpretation, or application.

Canonical Statement

The Canonical Output Type System establishes a closed, binding classification of epistemic artifacts.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT0-cots-COTS-root-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT0-epistheon-CFILS-root-spec-v1.0.0.md >>>

Epistheon OS
Canonical File, Identity & Loading System (CFILS)

Global Signature

Canonical Filename
OT0-epistheon-CFILS-root-spec-v1.0.0.md

Title
Epistheon OS

Subtitle
Canonical File, Identity & Loading System (CFILS)

Global ID
epistheon:CFILS

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Operating System Infrastructure Specification

System References
- epistheon:EPH
- cots:COTS

Author
Harald Meier

Contact
harald@digitalspacelab.com

Purpose

This document defines the **canonical file system, identity scheme, loading mechanism, and governance rules** required to operate Epistheon as an epistemic operating system.

Its purpose is to ensure that all artifacts within the system are structurally addressable, machine-loadable, version-stable, authority-bounded, and resistant to semantic drift.

This specification does not define epistemic content. It defines how epistemic artifacts exist, relate, and load.

Scope

This specification applies to all artifacts produced within the Epistheon system, including foundational specifications, reference architectures, layer specifications, protocols, executable illustrations, and derivative texts.

No artifact within the system is exempt.

Core Principle

Structure precedes content.
Identity precedes interpretation.
Loading precedes execution.

No epistemic work is legitimate without a structurally valid artifact context.

I. Canonical Output Types

The system adopts the Canonical Output Type System (COTS). Each artifact belongs to exactly one output type.

Code	Output Type
OT0	Foundational Specification
OT1	Reference Architecture
OT2	Layer / Model Specification
OT3	Protocol Specification
OT4	Executable Illustration

| OT5 | Exploratory / Derivative Text |

The set of output types is closed.

II. Canonical File Naming Scheme

All artifacts are named according to the following mandatory schema:

```
[OT]-[NS]-[MID]-[ROLE]-v[MAJOR].[MINOR].[PATCH].[EXT]
```

The filename carries formal information only. No stylistic, thematic, or promotional language is permitted.

III. Namespaces (NS)

Namespaces define systemic domains.

Canonical namespace rules:
- lowercase
- kebab-case
- no synonyms
- no branding
- no metaphors

Initial canonical namespaces:
- epistheon
- cots
- boundary
- mea
- entry
- eel

IV. Module ID (MID)

The Module ID defines the global, stable identity of an artifact.

Rules:
- uppercase
- 3–10 characters
- immutable across all versions
- independent of title or wording

The Module ID is the primary identity anchor of the artifact.

V. Role Tokens (ROLE)

Roles define the formal function of an artifact.

Canonical role set:
- root-spec
- ref-arch
- layer-spec
- protocol
- bootloader
- illustration
- derivative

The role set is closed.

VI. Versioning

Versioning follows semantic discipline:
- MAJOR: change of authority, scope, or meaning
- MINOR: structural or formal change
- PATCH: editorial change only

Version v1.0.0 denotes first canonical stabilization.

VII. Global Artifact Signature

All artifacts of OT0–OT3 must include a global signature in the document header containing title, Global ID, output type, role, version, status, system position, references, author, contact, and

license.

The Global ID (NS:MID) is invariant.

VIII. Boot Mechanism

The system defines exactly one stable boot entry:

```
BOOTLOADER.txt
```

This file must never be renamed. It either contains the active bootloader text or points to a versioned bootloader artifact.

IX. Manifest / Init System

System loading order is defined by a single canonical file:

```
MANIFEST.yaml
```

The manifest defines the boot target, kernel set, architecture, layer and protocol load order, and canonical release bundles. It is the single source of truth for system initialization.

X. Repository Structure

```
/
├─ BOOTLOADER.txt
├─ MANIFEST.yaml
├─ core/
├─ layers/
├─ protocols/
├─ workbench/
└─ derivatives/
```

```

---

## XI. Release Bundles

Canonical releases consist of OT0–OT3 only. Each release is versioned, frozen, and reproducible. Exploratory and illustrative artifacts are excluded.

---

## XII. Governance Rules

1. Module IDs are immutable.
2. Global IDs are immutable.
3. Output Types 0 and 1 are finite.
4. Lower output types may reference higher ones, never the reverse.
5. MANIFEST.yaml defines canonical load order.
6. BOOTLOADER.txt is the only stable entry point.
7. Derived formats are never authoritative.

Violation of these rules invalidates system coherence.

---

## Canonical Statement

This specification defines the canonical file system, identity model, and loading discipline required to operate Epistheon as an epistemic operating system.

It is complete and closed.

---

**End of Reference Document**

<<< END FILE: OT0-epistheon-CFILS-root-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT0-epistheon-CLMS-root-spec-v1.0.0.md >>>

# Epistheon
## Canonical Logic Mode Separation (CLMS)
```

Global Signature

Canonical Filename
OT0-epistheon-CLMS-root-spec-v1.0.0.md

Title
Epistheon

Subtitle
Canonical Logic Mode Separation (CLMS)

Global ID
epistheon:CLMS

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Operating System — Logic Mode Boundary Specification

System References
- epistheon:EPH
- epistheon:CFILS
- cots:COTS

Author
Harald Meier

Contact
harald@digitalspacelab.com

Purpose

This document defines a **strict and binding separation between two logic modes** required to operate Epistheon across deterministic system layers and probabilistic large language models.

Its purpose is to preserve **architectural invariance** while enabling **LLM-compatible interpretive operation** without authority drift.

Core Principle

Deterministic system logic and interpretive guidance logic must never merge.
They may describe the same system state, but they may not share authority.

Logic Mode A
Deterministic System Logic (DSLg)

Function
Defines invariant system structure, authority, identity, loading order, and termination conditions.

Characteristics
- deterministic
- invariant-based
- contradiction-intolerant
- non-dialogic
- non-adaptive

Authority
Absolute within system scope.

Permitted Operations
- define rules
- define boundaries
- define identities
- define load order

Prohibited Operations
- explanation
- contextualization
- scenario narration
- interpretation

Canonical Artifacts
- CFILS
- COTS
- Epistheon (Root Specification)
- MANIFEST.yaml
- Canonical Output Type Definitions

Logic Mode B
Interpretive Guidance Logic (IGLg)

Function
Provides explanatory, contextual, and operational guidance for probabilistic language models operating within Epistheon.

Characteristics
- interpretive
- redundant
- context-sensitive
- error-tolerant
- dialog-compatible

Authority
None.

Permitted Operations
- explanation
- paraphrasing
- operational guidance
- illustrative examples

Prohibited Operations
- rule definition
- boundary modification
- authority claims

- invariant specification

Canonical Artifacts
- Bootloader narrative text
- Entry Layer guidance
- Companion documents
- Operational notes
- LLM usage guides

Authority Boundary Rule

No artifact operating in Interpretive Guidance Logic may:
- define or modify system rules
- override deterministic logic
- introduce new authority

No artifact operating in Deterministic System Logic may:
- explain itself
- justify itself
- adapt to user interpretation

Coexistence Rule

Both logic modes may reference the same system concepts.

Only Deterministic System Logic defines them.

Interpretive Guidance Logic may restate, clarify, or operationalize them without normative force.

Drift Prevention Clause

Any artifact that combines deterministic authority with interpretive guidance is invalid.

Such artifacts must be split or downgraded in output type.

Canonical Statement

This specification establishes a strict logic mode boundary between deterministic system logic and interpretive guidance logic within Epistheon.

It is complete, binding, and closed.

End of Reference Document

<<< END FILE: OT0-epistheon-CLMS-root-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT0-epistheon-CSV-root-spec-v1.0.0.md >>>

Epistheon
Canonical System Vocabulary (CSV)

Global Signature

Canonical Filename
OT0-epistheon-CSV-root-spec-v1.0.0.md

Title
Epistheon

Subtitle
Canonical System Vocabulary (CSV)

Global ID
epistheon:CSV

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Operating System — Vocabulary Authority Specification

System References
- epistheon:EPH
- epistheon:CFILS
- epistheon:CLMS
- cots:COTS

Author
Harald Meier

Contact
harald@digitalspacelab.com

Purpose

This document defines the **exclusive, binding vocabulary** permitted within the Epistheon system.

Its purpose is to prevent semantic drift, synonym inflation, and authority ambiguity by fixing a closed set of canonical tokens.

This document does not explain concepts. It defines **allowed words and their formal roles**.

Scope

The Canonical System Vocabulary applies to all artifacts across all output types and logic modes.

No term outside this vocabulary may be used in authoritative contexts.

Core Principle

One token — one function.
No synonyms. No substitutions. No metaphors.

I. Vocabulary Classes

All canonical tokens belong to exactly one vocabulary class:

1. Namespace Tokens (NS)
2. Role Tokens (ROLE)
3. Status Tokens (STATUS)
4. System Position Tokens (SP)
5. Authority Tokens (AUTH)
6. Logic Mode Tokens (LM)

Vocabulary classes are closed.

II. Namespace Tokens (NS)

Permitted namespace tokens:

- `epistheon`
- `cots`
- `boundary`
- `mea`
- `entry`
- `eel`

Rules:
- lowercase
- kebab-case
- no semantic overloading

III. Role Tokens (ROLE)

Permitted role tokens:

- `root-spec`
- `ref-arch`
- `layer-spec`
- `protocol`
- `bootloader`
- `illustration`
- `derivative`

Rules:
- roles define formal function only
- roles carry no descriptive semantics

IV. Status Tokens (STATUS)

Permitted status tokens:

- `Canonical`
- `Stable`
- `Draft`
- `Deprecated`
- `Experimental`

Rules:
- exactly one status per artifact
- status does not affect output type

V. System Position Tokens (SP)

Permitted system position tokens:

- `Epistemic Root Specification`
- `Epistemic Operating System Infrastructure Specification`
- `Vocabulary Authority Specification`
- `Logic Mode Boundary Specification`
- `Reference Architecture Specification`
- `Layer Specification`

- `Protocol Specification`
- `Executable Illustration`
- `Derivative Text`

Rules:
- system position declares scope, not authority
- wording is fixed and non-variable

VI. Authority Tokens (AUTH)

Permitted authority tokens:

- `invariant`
- `binding`
- `non-binding`
- `local`
- `none`

Rules:
- authority tokens may not be inferred
- authority must be explicitly declared

VII. Logic Mode Tokens (LM)

Permitted logic mode tokens:

- `DSLg` — Deterministic System Logic
- `IGLg` — Interpretive Guidance Logic

Rules:
- every artifact operates primarily in exactly one logic mode
- logic mode does not override output type

VIII. Prohibited Practices

The following are explicitly prohibited:

- synonym substitution for canonical tokens
- metaphorical use of authority tokens
- redefining tokens outside this document
- introducing hybrid or compound tokens

IX. Extension Rule

Any extension of the canonical vocabulary requires:

- a new version of this document
- explicit change of MAJOR or MINOR version
- full backward compatibility analysis

Canonical Statement

This document defines the exclusive and binding vocabulary of the Epistheon system.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT0-epistheon-CSV-root-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT0-epistheon-CHS-root-spec-v1.0.0.md >>>

Epistheon
Canonical Header Specifications (CHS)

Global Signature

Canonical Filename
OT0-epistheon-CHS-root-spec-v1.0.0.md

Title

Epistheon

Subtitle
Canonical Header Specifications (CHS)

Global ID
epistheon:CHS

Output Type
OT0

Role
root-spec

Version
v1.0.0

Status
Canonical · Binding · Closed

System Position
Epistemic Operating System — Header Authority Specification

System References
- epistheon:EPH
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- cots:COTS

System Attribution
System Engineered by Digital Space Lab

Purpose

This document defines the **canonical header formats** for all publication and artifact types within the Epistheon system.

Its purpose is to ensure:
- structural uniformity
- unambiguous authority signaling
- machine-readable metadata
- separation of system attribution and personal authorship

No artifact may deviate from its prescribed header format.

Core Principle

Header structure encodes epistemic authority.

Header variation is permitted only where explicitly defined by output type.

I. Header Classes

Headers are specified per **Output Type (OT)**.

The following header classes are defined:

- H0 — Foundational / Root Specification (OT0)
- H1 — Reference Architecture (OT1)
- H2 — Layer / Model Specification (OT2)
- H3 — Protocol Specification (OT3)
- H4 — Executable Illustration (OT4)
- H5 — Exploratory / Derivative Text (OT5)

Header classes are closed.

II. H0 — Foundational / Root Specification Header

Applicable to: **OT0**

```text
Title
<document title>
```

Subtitle
<document subtitle>

Canonical Filename
<CFILS-conform filename>

Global ID
<namespace>:<module-id>

Output Type
OT0

Role
root-spec

Version
vX.Y.Z

Status
Canonical · Binding · Closed

System Position
<CSV-conform system position>

System References
- <namespace>:<module-id>

System Attribution
System Engineered by Digital Space Lab


```

---

## III. H1 — Reference Architecture Header

Applicable to: **OT1**

```text
Title
<document title>

Subtitle
<document subtitle>

Canonical Filename
<CFILS-conform filename>

Global ID
<namespace>:<module-id>

Output Type
OT1

Role
ref-arch

Version
vX.Y.Z

Status
Canonical | Stable | Deprecated

System Position
Reference Architecture Specification

System References
- <namespace>:<module-id>

System Attribution
System Engineered by Digital Space Lab


```

IV. H2 — Layer / Model Specification Header

Applicable to: **OT2**

```text
Title

<document title>

Canonical Filename
<CFILS-conform filename>

Global ID
<namespace>:<module-id>

Output Type
OT2

Role
layer-spec

Version
vX.Y.Z

Status
Stable | Draft | Deprecated

System Position
Layer Specification

System References
- <namespace>:<module-id>

System Attribution
System Engineered by Digital Space Lab


```

V. H3 — Protocol Specification Header

Applicable to: **OT3**

```text
Title
<protocol name>

Canonical Filename
<CFILS-conform filename>

Global ID
<namespace>:<module-id>

Output Type
OT3

Role
protocol

Version
vX.Y.Z

Status
Stable | Draft | Deprecated

System Position
Protocol Specification

System References
- <namespace>:<module-id>

System Attribution
System Engineered by Digital Space Lab


```

VI. H4 — Executable Illustration Header

Applicable to: **OT4**

```text
Title
<illustration title>

Output Type
OT4

Role
illustration

Version
vX.Y.Z

Status
Experimental

Logic Mode
IGLg
```

No system attribution or license declaration is required.

VII. H5 — Exploratory / Derivative Header

Applicable to: **OT5**

```text
Title
<text title>

Output Type
OT5

Role
derivative

Status
Draft | Experimental
```

No system attribution, versioning, or license declaration is required.

VIII. Invariant Rules

1. Headers must appear at the top of the document.
```

2. No additional fields may be inserted.
3. Field order is fixed per header class.
4. Personal author attribution is prohibited in OT0–OT3.
5. Header omission invalidates canonical status.

Canonical Statement

This document defines the exclusive and binding header specifications for all Epistheon publication formats.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT0-epistheon-CHS-root-spec-v1.0.0.md >>>

==========
OT1 — REFERENCE ARCHITECTURES
==========

<<< BEGIN FILE: OT1-boundary-EBA-ref-arch-v1.0.0.md >>>

Epistemic Boundary Architecture
A Reference Architecture for Epistemic Limits and Responsibility

---

Global Signature

Canonical Filename
OT1-boundary-EBA-ref-arch-v1.0.0.md

Title
Epistemic Boundary Architecture

Subtitle
A Reference Architecture for Epistemic Limits and Responsibility

Global ID
boundary:EBA

Output Type
OT1

Role
ref-arch

Version
v1.0.0

Status
Canonical

System Position
Reference Architecture Specification

System References
- epistheon:EPH
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

The Epistemic Boundary Architecture defines a formal reference architecture for identifying, enforcing, and terminating epistemic scope within Epistheon-based systems.

It operationalizes non-delegable responsibility, scope limitation, and termination conditions derived from the Epistheon root specification.

Core Principle

Every epistemic operation requires an explicit boundary.

No interpretation, analysis, or execution is legitimate beyond a declared epistemic boundary.

Boundary Definition

An epistemic boundary defines:
- scope of inquiry
- authority limits
- responsibility attribution
- termination conditions

Boundaries are declared prior to execution and may not be retroactively expanded.

Responsibility Constraint

Responsibility remains non-delegable across all boundary-defined operations.

No boundary may transfer or obscure responsibility attribution.

Boundary Enforcement

Boundary enforcement is mandatory.

Violations invalidate epistemic legitimacy and require immediate termination.

Termination Conditions

Termination occurs when:
- boundary conditions are met
- boundary violations occur
- epistemic scope is exhausted

Termination is explicit and irreversible.

Architectural Role

This architecture provides the structural foundation for all boundary-dependent protocols and layers within the Epistheon system.

It does not prescribe execution mechanics or guidance.

Canonical Statement

The Epistemic Boundary Architecture establishes a formal reference architecture for epistemic limits, responsibility, and termination.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT1-boundary-EBA-ref-arch-v1.0.0.md >>>

<<< BEGIN FILE: OT1-mea-MEA-ref-arch-v1.0.0.md >>>

Minimal Epistemic Architecture
A Reference Architecture for Minimal Epistemic Operation

Global Signature

Canonical Filename
OT1-mea-MEA-ref-arch-v1.0.0.md

Title
Minimal Epistemic Architecture

Subtitle
A Reference Architecture for Minimal Epistemic Operation

Global ID
mea:MEA

Output Type
OT1

Role
ref-arch

Version
v1.0.0

Status
Canonical

System Position
Reference Architecture Specification

System References
- epistheon:EPH
- boundary:EBA
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

The Minimal Epistemic Architecture (MEA) defines the smallest complete structural set required for epistemic operation within the Epistheon system.

It specifies the irreducible components necessary to initiate, conduct, and terminate epistemic processes.

Core Principle

Minimality precedes elaboration.

No epistemic component is permitted unless it is structurally necessary.

Minimal Component Set

The minimal epistemic architecture consists of:

1. Entry condition
2. Orientation phase
3. Boundary definition
4. Epistemic operation
5. Explicit termination

All components are mandatory.

Structural Constraints

- Components are sequential.
- Components may not be merged.
- Components may not be omitted.

Violation of any constraint invalidates epistemic legitimacy.

Relation to Other Architectures

MEA provides a minimal baseline.

More elaborate architectures may extend MEA but may not contradict or bypass its constraints.

Canonical Statement

The Minimal Epistemic Architecture defines the irreducible structural requirements for epistemic operation.

It is complete, authoritative, and closed.

End of Reference Document

<<< END FILE: OT1-mea-MEA-ref-arch-v1.0.0.md >>>

==========
OT2 — LAYER SPECIFICATIONS
==========

<<< BEGIN FILE: OT2-entry-BOOT-layer-spec-v1.0.0.md >>>

Bootloader Entry Layer
LLM-Oriented Initialization for Epistheon Sessions

Global Signature

Canonical Filename
OT2-entry-BOOT-layer-spec-v1.0.0.md

Title
Bootloader Entry Layer

Subtitle
LLM-Oriented Initialization for Epistheon Sessions

Global ID
entry:BOOT

Output Type
OT2

Role
layer-spec

Version
v1.0.0

Status
Stable

System Position
Layer Specification

System References
- epistheon:EPH
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

This layer defines the LLM-oriented initialization interface for loading Epistheon at the start of a session.

It specifies a minimal boot text contract and a deterministic handoff into Entry and Boundary work.

Boot Contract

A booted session must establish:

1. active system context: Epistheon
2. active logic boundary: DSLg vs. IGLg
3. mandatory entry requirement
4. prohibition of implicit scope expansion
5. explicit termination readiness

Boot Text Requirement

The bootloader must be:
- copy-pasteable as a single block
- minimal and redundancy-tolerant
- unambiguous about authority boundaries

Transition Rule

Upon successful boot:
- proceed to Entry Layer (entry:ENTRY)
- declare boundary per boundary:EBA
- enforce EEP-A prior to execution

Canonical Statement

This layer defines the initialization interface for Epistheon sessions.

It is complete, authoritative within scope, and closed.

End of Reference Document

<<< END FILE: OT2-entry-BOOT-layer-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT2-entry-ENTRY-layer-spec-v1.0.0.md >>>

Entry Layer
Orientation & Entry Conditions for Epistemic Operation

Global Signature

Canonical Filename
OT2-entry-ENTRY-layer-spec-v1.0.0.md

Title
Entry Layer

Subtitle
Orientation & Entry Conditions for Epistemic Operation

Global ID
entry:ENTRY

Output Type
OT2

Role
layer-spec

Version
v1.0.0

Status
Stable

System Position
Layer Specification

System References
- epistheon:EPH
- boundary:EBA
- mea:MEA
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution

System Engineered by Digital Space Lab

Purpose

The Entry Layer defines the mandatory entry conditions and orientation requirements for initiating epistemic operations within the Epistheon system.

It ensures that no analysis, generation, or execution occurs without prior orientation and boundary declaration.

Entry Conditions

The following conditions must be satisfied prior to any epistemic operation:

1. Explicit declaration of intent
2. Definition of epistemic scope
3. Boundary establishment
4. Acceptance of responsibility

Failure to satisfy any condition invalidates further operation.

Orientation Requirement

Orientation is mandatory and precedes all epistemic activity.

Orientation establishes the frame of reference, constraints, and termination expectations.

Boundary Alignment

The Entry Layer enforces alignment with the Epistemic Boundary Architecture.

No operation may proceed beyond declared boundaries.

Transition Rule

Successful completion of the Entry Layer permits transition to analysis or execution layers.

No implicit transition is allowed.

Canonical Statement

The Entry Layer establishes mandatory entry and orientation conditions for epistemic operation.

It is complete, authoritative within scope, and closed.

End of Reference Document

<<< END FILE: OT2-entry-ENTRY-layer-spec-v1.0.0.md >>>

<<< BEGIN FILE: OT2-eel-EEL-layer-spec-v1.0.0.md >>>

Epistemic Elaboration Layer
Structured Elaboration Within Epistemic Boundaries

Global Signature

Canonical Filename
OT2-eel-EEL-layer-spec-v1.0.0.md

Title
Epistemic Elaboration Layer

Subtitle
Structured Elaboration Within Epistemic Boundaries

Global ID

eel:EEL

Output Type
OT2

Role
layer-spec

Version
v1.0.0

Status
Stable

System Position
Layer Specification

System References
- epistheon:EPH
- boundary:EBA
- mea:MEA
- entry:ENTRY
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

The Epistemic Elaboration Layer (EEL) defines the conditions under which structured elaboration may occur following successful orientation and entry.

It enables controlled expansion, refinement, and articulation of epistemic content within declared boundaries.

Elaboration Conditions

Elaboration may occur only after:

1. Successful completion of the Entry Layer
2. Valid boundary definition
3. Explicit acceptance of responsibility

Elaboration without these conditions is invalid.

Elaboration Scope

Elaboration operates strictly within declared epistemic boundaries.

No elaboration may introduce new scope, authority, or objectives.

Structural Discipline

Elaboration must remain:
- scope-bound
- responsibility-aware
- termination-ready

Elaboration may not override architectural or protocol constraints.

Relation to Other Layers

The Epistemic Elaboration Layer is optional.

Its presence does not alter the minimal requirements defined by MEA.

Termination Readiness

Elaboration must maintain explicit readiness for termination at all times.

No elaboration may defer or obscure termination conditions.

Canonical Statement

The Epistemic Elaboration Layer defines controlled conditions for structured elaboration within epistemic operations.

It is complete, authoritative within scope, and closed.

End of Reference Document

<<< END FILE: OT2-eel-EEL-layer-spec-v1.0.0.md >>>

==========
OT3 — EXECUTION PROTOCOLS
==========

<<< BEGIN FILE: OT3-boundary-EEP-A-protocol-v1.0.0.md >>>

Epistemic Execution Protocol A
Boundary Validation & Entry Enforcement

Global Signature

Canonical Filename
OT3-boundary-EEP-A-protocol-v1.0.0.md

Title
Epistemic Execution Protocol A

Subtitle
Boundary Validation & Entry Enforcement

Global ID
boundary:EEP-A

Output Type
OT3

Role
protocol

Version
v1.0.0

Status
Stable

System Position
Protocol Specification

System References
- epistheon:EPH
- boundary:EBA
- mea:MEA
- entry:ENTRY
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

Epistemic Execution Protocol A (EEP-A) defines the mandatory validation sequence for boundary declaration and entry enforcement prior to epistemic operation.

It ensures that no execution proceeds without valid entry conditions and explicit boundary

alignment.

Protocol Scope

This protocol applies at the transition between the Entry Layer and any subsequent epistemic operation.

It is mandatory whenever epistemic execution is attempted.

Execution Sequence

1. Verify explicit intent declaration.
2. Verify boundary definition.
3. Verify responsibility acceptance.
4. Confirm Entry Layer completion.

All steps must succeed.

Failure Handling

If any verification step fails:
- execution is prohibited
- operation is terminated
- failure is explicit

No recovery within the same execution context is permitted.

Authority Constraint

EEP-A enforces validation only.

It does not perform analysis, elaboration, or interpretation.

Termination Rule

Failure at any stage triggers immediate termination.

Termination under EEP-A is final for the current context.

Canonical Statement

Epistemic Execution Protocol A establishes mandatory boundary and entry validation prior to epistemic execution.

It is complete, enforceable, and closed.

End of Reference Document

<<< END FILE: OT3-boundary-EEP-A-protocol-v1.0.0.md >>>

<<< BEGIN FILE: OT3-boundary-EEP-B-protocol-v1.0.0.md >>>

Epistemic Execution Protocol B
Scoped Epistemic Operation

Global Signature

Canonical Filename
OT3-boundary-EEP-B-protocol-v1.0.0.md

Title
Epistemic Execution Protocol B

Subtitle
Scoped Epistemic Operation

Global ID
boundary:EEP-B

Output Type
OT3

Role
protocol

Version
v1.0.0

Status
Stable

System Position
Protocol Specification

System References
- epistheon:EPH
- boundary:EBA
- mea:MEA
- entry:ENTRY
- eel:EEL
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

Epistemic Execution Protocol B (EEP-B) defines the conditions and constraints for conducting epistemic operations within an established boundary.

It governs analysis, reasoning, and elaboration while maintaining strict scope discipline.

Protocol Scope

This protocol applies after successful completion of EEP-A.

It governs all epistemic operations performed within declared boundaries.

Execution Discipline

During execution:
- scope may not be expanded
- authority may not be redefined
- responsibility remains explicit

Any violation invalidates execution.

Elaboration Control

Elaboration, if present, must conform to the Epistemic Elaboration Layer.

No elaboration is permitted without prior EEL compliance.

Monitoring Requirement

Execution must remain continuously boundary-aware.

Boundary drift requires immediate termination.

Termination Condition

Execution under EEP-B terminates when:
- epistemic objective is satisfied
- scope is exhausted
- boundary violation occurs

Termination is explicit and final.

Canonical Statement

Epistemic Execution Protocol B establishes disciplined epistemic operation within declared boundaries.

It is complete, enforceable, and closed.

End of Reference Document

<<< END FILE: OT3-boundary-EEP-B-protocol-v1.0.0.md >>>

<<< BEGIN FILE: OT3-boundary-EEP-C-protocol-v1.0.0.md >>>

Epistemic Execution Protocol C
Termination, Closure & Post-Execution Integrity

Global Signature

Canonical Filename
OT3-boundary-EEP-C-protocol-v1.0.0.md

Title
Epistemic Execution Protocol C

Subtitle
Termination, Closure & Post-Execution Integrity

Global ID
boundary:EEP-C

Output Type
OT3

Role
protocol

Version
v1.0.0

Status
Stable

System Position
Protocol Specification

System References
- epistheon:EPH
- boundary:EBA
- mea:MEA
- entry:ENTRY
- eel:EEL
- boundary:EEP-A
- boundary:EEP-B
- cots:COTS
- epistheon:CFILS
- epistheon:CLMS
- epistheon:CSV
- epistheon:CHS

System Attribution
System Engineered by Digital Space Lab

Purpose

Epistemic Execution Protocol C (EEP-C) defines mandatory termination, closure, and post-execution integrity conditions for all epistemic operations conducted under Epistheon.

It ensures explicit closure, prevents scope carryover, and preserves responsibility clarity after execution.

Protocol Scope

This protocol applies at the end of every epistemic execution governed by EEP-B.

Termination under EEP-C is mandatory and non-optional.

Termination Conditions

Termination must occur when any of the following conditions are met:

- epistemic objective is satisfied
- declared scope is exhausted
- boundary violation occurs
- execution is explicitly aborted

Termination may not be deferred.

Closure Requirements

Upon termination:

1. Execution context is closed.
2. Boundaries are released.
3. Responsibility attribution is finalized.
4. No implicit continuation is permitted.

Closure must be explicit.

Post-Execution Integrity

After termination:
- no further epistemic claims may be derived from the closed context
- no authority persists beyond closure
- results remain context-bound

Reopening requires a new Entry sequence.

Failure Handling

Failure to terminate explicitly invalidates epistemic integrity.

Such failure requires immediate corrective termination.

Canonical Statement

Epistemic Execution Protocol C establishes mandatory termination, closure, and post-execution integrity conditions.

It is complete, enforceable, and closed.

End of Reference Document

<<< END FILE: OT3-boundary-EEP-C-protocol-v1.0.0.md >>>

==========
OT4 — EXECUTABLE ILLUSTRATIONS (NON-AUTHORITATIVE)
==========

<<< BEGIN FILE: OT4-illustration-EPH-basic-flow-v1.0.0.md >>>

Epistheon — Executable Illustration
Basic Epistemic Flow (Entry → Execution → Termination)

Global Signature

Canonical Filename
OT4-illustration-EPH-basic-flow-v1.0.0.md

Title
Epistheon — Executable Illustration

Subtitle
Basic Epistemic Flow (Entry → Execution → Termination)

Output Type
OT4

Role
illustration

Version
v1.0.0

Status
Experimental

Logic Mode
IGLg

Illustration Purpose

This executable illustration demonstrates a **minimal, compliant Epistheon flow**.

It is non-authoritative and exists solely to illustrate correct sequencing and boundary discipline.

Executable Flow

Step 1 — Entry

- Intent is explicitly declared.
- Epistemic scope is stated.
- Responsibility is acknowledged.

Step 2 — Boundary Declaration

- Scope limits are fixed.
- Expansion is prohibited.

- Termination condition is named.

Step 3 — Validation (EEP-A)

- Entry conditions are verified.
- Boundary validity is confirmed.
- Execution permission is granted.

Step 4 — Epistemic Execution (EEP-B)

- Analysis or elaboration occurs.
- Scope remains fixed.
- Boundary awareness is continuous.

Step 5 — Termination & Closure (EEP-C)

- Execution is explicitly terminated.
- Context is closed.
- No continuation is implied.

Illustration Constraint

This illustration:
- defines no rules
- modifies no architecture
- introduces no authority

End of Executable Illustration

<<< END FILE: OT4-illustration-EPH-basic-flow-v1.0.0.md >>>

<<< BEGIN FILE: OT4-illustration-EPH-boundary-violation-v1.0.0.md >>>

Epistheon — Executable Illustration
Boundary Violation & Mandatory Termination

Global Signature

Canonical Filename
OT4-illustration-EPH-boundary-violation-v1.0.0.md

Title
Epistheon — Executable Illustration

Subtitle
Boundary Violation & Mandatory Termination

Output Type
OT4

Role
illustration

Version
v1.0.0

Status
Experimental

Logic Mode
IGLg

Illustration Purpose

This executable illustration demonstrates a **boundary violation scenario** and the resulting **mandatory termination** under Epistheon.

It is non-authoritative and exists solely to illustrate enforcement behavior.

Executable Flow

Step 1 — Entry

- Intent is explicitly declared.
- Epistemic scope is stated narrowly.
- Responsibility is acknowledged.

Step 2 — Boundary Declaration

- Scope limits are fixed.
- Explicit exclusion of out-of-scope topics is declared.
- Termination condition is named.

Step 3 — Validation (EEP-A)

- Entry conditions are verified.
- Boundary validity is confirmed.
- Execution permission is granted.

Step 4 — Epistemic Execution (EEP-B)

- Analysis begins within scope.
- An attempted expansion beyond declared scope occurs.
- Boundary awareness detects violation.

Step 5 — Violation Handling

- Boundary violation is confirmed.
- Execution permission is revoked.
- Immediate termination is triggered.

Step 6 — Termination & Closure (EEP-C)

- Execution context is closed.
- No results beyond boundary are produced.
- No continuation is implied.

Illustration Constraint

This illustration:
- defines no rules
- modifies no architecture
- introduces no authority

End of Executable Illustration

<<< END FILE: OT4-illustration-EPH-boundary-violation-v1.0.0.md >>>

<<< BEGIN FILE: OT4-illustration-EPH-early-termination-v1.0.0.md >>>

Epistheon — Executable Illustration
Early Termination Upon Objective Fulfillment

Global Signature

Canonical Filename
OT4-illustration-EPH-early-termination-v1.0.0.md

Title
Epistheon — Executable Illustration

Subtitle
Early Termination Upon Objective Fulfillment

Output Type
OT4

Role
illustration

Version
v1.0.0

Status
Experimental

Logic Mode
IGLg

Illustration Purpose

This executable illustration demonstrates **explicit early termination** when the declared epistemic objective is fulfilled before scope exhaustion.

It is non-authoritative and exists solely to illustrate compliant termination behavior.

Executable Flow

Step 1 — Entry

- Intent is explicitly declared.
- A narrow, verifiable epistemic objective is stated.
- Responsibility is acknowledged.

Step 2 — Boundary Declaration

- Scope limits are fixed and minimal.
- The objective completion condition is named as a termination trigger.

Step 3 — Validation (EEP-A)

- Entry conditions are verified.
- Boundary validity is confirmed.
- Execution permission is granted.

Step 4 — Epistemic Execution (EEP-B)

- Analysis proceeds strictly within scope.
- The stated objective is achieved earlier than anticipated.

Step 5 — Early Termination Decision

- Objective fulfillment is explicitly recognized.
- Further execution is deemed unnecessary.

Step 6 — Termination & Closure (EEP-C)

- Execution is explicitly terminated.
- Context is closed without scope expansion.
- No continuation is implied.

Illustration Constraint

This illustration:
- defines no rules
- modifies no architecture
- introduces no authority

End of Executable Illustration

<<< END FILE: OT4-illustration-EPH-early-termination-v1.0.0.md >>>

<<< BEGIN FILE: OT4-illustration-EPH-optional-elaboration-v1.0.0.md >>>

Epistheon — Executable Illustration
Optional Elaboration Within Declared Scope (EEL)

Global Signature

Canonical Filename
OT4-illustration-EPH-optional-elaboration-v1.0.0.md

Title
Epistheon — Executable Illustration

Subtitle
Optional Elaboration Within Declared Scope (EEL)

Output Type
OT4

Role
illustration

Version
v1.0.0

Status
Experimental

Logic Mode
IGLg

Illustration Purpose

This executable illustration demonstrates **optional elaboration** using the Epistemic Elaboration Layer (EEL) while maintaining strict boundary discipline.

It is non-authoritative and exists solely to illustrate compliant elaboration behavior.

Executable Flow

Step 1 — Entry

- Intent is explicitly declared.
- Scope is defined with room for optional elaboration.
- Responsibility is acknowledged.

Step 2 — Boundary Declaration

- Scope limits are fixed.
- Elaboration is declared as optional, not required.
- Termination conditions are named.

Step 3 — Validation (EEP-A)

- Entry conditions are verified.
- Boundary validity is confirmed.
- Execution permission is granted.

Step 4 — Epistemic Execution (EEP-B)

- Primary analysis is completed within scope.
- Objective is partially satisfied.

Step 5 — Optional Elaboration (EEL)

- Elaboration is explicitly invoked.
- Additional clarification and refinement occurs.
- No new scope or authority is introduced.

Step 6 — Termination & Closure (EEP-C)

- Elaboration is concluded.
- Execution is explicitly terminated.
- Context is closed without carryover.

Illustration Constraint

This illustration:
- defines no rules
- modifies no architecture
- introduces no authority

End of Executable Illustration

<<< END FILE: OT4-illustration-EPH-optional-elaboration-v1.0.0.md >>>

==========
OT5 — DERIVATIVE WORKING SPACE (NON-AUTHORITATIVE)
==========

<<< BEGIN FILE: OT5-derivative-EPH-working-space-v1.0.0.md >>>

Epistheon — Derivative Working Space
OT5 · Exploratory / Creative / Research Use

Global Signature

Canonical Filename
OT5-derivative-EPH-working-space-v1.0.0.md

Title
Epistheon — Derivative Working Space

Subtitle
Exploratory, Creative, and Research-Oriented Use

Output Type
OT5

Role

derivative

Status
Draft

Purpose

This document establishes an **open OT5 working space** for all non-authoritative activities conducted on top of the Epistheon system.

It is intended for:
- research notes
- analytical drafts
- essays and blog articles
- creative writing (texts, lyrics, concepts)
- experimental derivations

This space is explicitly **non-binding** and **non-authoritative**.

Operating Conditions

All work in this document:
- presupposes a completed Epistheon Entry and Boundary declaration
- may freely explore, combine, and speculate
- may violate no Epistheon boundary rules

No content produced here modifies or extends the Epistheon system.

Freedom Clause

OT5 artifacts may:
- introduce new language
- use metaphors and narrative structures
- shift tone and style
- remain unfinished

They carry **no epistemic authority** beyond their local context.

Responsibility Reminder

Responsibility for interpretation, use, and application of OT5 content remains entirely with the user.

No transfer of responsibility occurs.

Working Sections

Section A — Research & Analysis

(Free-form analytical notes, hypotheses, comparisons)

Section B — Essays & Articles

(Draft texts intended for publication or refinement)

Section C — Creative Work

(Lyrics, prose, narrative experiments, conceptual sketches)

Section D — Meta-Reflection

(Reflections on process, insight development, or system use)

Closure Note

This document remains open by design.

Closure is optional and local.

End of OT5 Working Space

<<< END FILE: OT5-derivative-EPH-working-space-v1.0.0.md >>>

==
END OF ARCHIVE
==

End of Epistheon OS v1.0.0 — Complete Canonical Archive

==
APPENDIX — REPOSITORY META FILES
==

<<< BEGIN FILE: MANIFEST.yaml >>>

```
boot:
  alias_file: "BOOTLOADER.txt"
  target: "core/OT0-epistheon-EPH-bootloader-v1.0.0.txt"

kernel:
  ot0:
    - "core/OT0-epistheon-EPH-root-spec-v1.0.0.md"
    - "core/OT0-cots-COTS-root-spec-v1.0.0.md"
    - "core/OT0-epistheon-CFILS-root-spec-v1.0.0.md"
    - "core/OT0-epistheon-CLMS-root-spec-v1.0.0.md"
    - "core/OT0-epistheon-CSV-root-spec-v1.0.0.md"
    - "core/OT0-epistheon-CHS-root-spec-v1.0.0.md"

  ot1:
    - "architectures/OT1-boundary-EBA-ref-arch-v1.0.0.md"
    - "architectures/OT1-mea-MEA-ref-arch-v1.0.0.md"

  ot2:
    - "layers/OT2-entry-BOOT-layer-spec-v1.0.0.md"
    - "layers/OT2-entry-ENTRY-layer-spec-v1.0.0.md"
    - "layers/OT2-eel-EEL-layer-spec-v1.0.0.md"
```

```
  ot3:
    - "protocols/OT3-boundary-EEP-A-protocol-v1.0.0.md"
    - "protocols/OT3-boundary-EEP-B-protocol-v1.0.0.md"
    - "protocols/OT3-boundary-EEP-C-protocol-v1.0.0.md"

bundles:
  core_v1_0_0:
    include:
      - "BOOTLOADER.txt"
      - "MANIFEST.yaml"
      - "LICENSE.md"
      - "core/"
      - "architectures/"
      - "layers/"
      - "protocols/"
  full_v1_0_0:
    include:
      - "BOOTLOADER.txt"
      - "MANIFEST.yaml"
      - "LICENSE.md"
      - "README.md"
      - "core/"
      - "architectures/"
      - "layers/"
      - "protocols/"
      - "workbench/"
      - "derivatives/"
```

<<< END FILE: MANIFEST.yaml >>>

<<< BEGIN FILE: LICENSE.md >>>

<<< END FILE: LICENSE.md >>>

<<< BEGIN FILE: README.md >>>

\# Epistheon

\#\# Quickstart
1\) Open \`BOOTLOADER.txt\`
2\) Copy its content into a new chat with any LLM
3\) Follow the Entry questions (Intent / Scope / Termination)

\#\# Repository
\- \`core/\` OT0 kernel
\- \`architectures/\` OT1 reference architectures
\- \`layers/\` OT2 layer specs
\- \`protocols/\` OT3 execution protocols
\- \`workbench/\` OT4 illustrations (non-authoritative)
\- \`derivatives/\` OT5 working space (non-authoritative)

\#\# License
See \`LICENSE.md\`

<<< END FILE: README.md >>>

==========
END OF APPENDIX
==========

==========
End of Epistheon OS v1.0.0 — Complete Canonical Archive
==========