AgentNet Standards (ANS) — Core Specification v2.0

A Federated Standard for Authoritative Machine-Readable Data and Linked-Data-Grounded AI Reasoning

Version: 2.0
Status: Final
Effective Date: 2026-01-07

Published by: AgentNet

Document Scope

This document defines the core architectural, governance, and operational requirements of the AgentNet ecosystem. It establishes normative standards for identity, data publication, resolution, provenance, and Linked Data Retrieval Augmented Generation (LD-RAG).

This specification is intended for implementers, standards bodies, enterprises, institutions, and regulators.

Copyright and Use

Citation

AgentNet Standards (ANS) — Core Specification v2.0, AgentNet, Effective 2026-01-07.

Table of Contents

1.Document Status, Scope, and Normative Language....8
1.1.Document Status 8
1.2.Scope of the Standard 8
1.3.Normative Language 9
1.4.Architectural Principles 9

1.5.Non-Exclusivity and Coexistence 9
1.6.Versioning and Stability Expectations 11
1.7.Reserved Scope for Future Standards 11
1.8.Conformance 11
2.Terminology and Core Definitions 12
2.1.Purpose of This Section 12
2.2.AgentNet 12
2.3.Capsule 12
2.4.Node 13
2.5.Publisher 13
2.6.Resolver 13
2.7.Registrar 14
2.8.Ontology 14
2.9.Provenance 14
2.10.LD-RAG (Linked Data Retrieval Augmented Generation) 15
2.11.Authority 15
2.12.Consumer 15
2.13.Conformance Terminology 17
2.14.Deterministic 17
2.15.Policy Block 17
2.16.Node Policy 17
2.17.Capsule Policy 17
2.18.Effective Policy 18
2.19.Policy Inheritance Rules 18
3.System Architecture Overview 19
3.1.Purpose of This Section 19
3.2.Architectural Model 19
3.3.Separation of Identity and Data 20
3.4.Authority Flow 20

3.5.Federated Resolution Model **21**
3.6.Trust Boundaries **21**
3.7.Multi-Ontology Architecture **22**
3.8.Independence from Transport and Storage **22**
3.9.Architectural Constraints **22**
4.Capsule Specification 23
4.1.Purpose of This Section **23**
4.2.Definitions (Section-Specific) **23**
4.3.General Capsule Requirements **23**
4.4.JSON-LD Context and Type Requirements **24**
4.5.Capsule Metadata Requirements **24**
4.6.Provenance Requirements **27**
4.7.Versioning and Replacement Rules **28**
4.8.Multi-Ontology and Contextual Operation **29**
4.9.Inference Constraints **29**
4.10.Prohibited Capsule Behaviors **29**
4.11.Provenance, Governance, and Economics Inheritance **31**
5.Node Specification 32
5.1.Purpose of This Section **32**
5.2.Definitions (Section-Specific) **32**
5.3.Node General Requirements **32**
5.4.Node Identifier Requirements **33**
5.5.Node Lifecycle States **34**
5.6.Node–Capsule Binding Rules **35**
5.7.Node Merging and Splitting **36**
5.8.Node Revocation and Retirement **37**
5.9.Node Resolution Expectations **37**
5.10.Prohibited Node Behaviors **37**
5.11.Node Policy Requirements **37**

6.AgentNet Inquiry and Resolution Process 39

6.1.Purpose and Scope 39

6.2.Inquiry Model Overview 39

6.3.Consumer Role in Inquiry Submission 39

6.4.Inquiry Structure and Semantics 40

6.5.Identity Grounding and Responsibility 42

6.6.Resolver Eligibility Evaluation 43

6.7.Determinism and Auditability 43

6.8.Separation of Concerns 44

6.9.Relationship to LD-RAG Systems 44

6.10.Non-Resolution as a Valid Outcome 44

6.11.Conformance 46

7.Resolver Requirements 47

7.1.Purpose of This Section 47

7.2.Definitions (Section-Specific) 47

7.3.General Resolver Obligations. 47

7.4.Resolution Scope and Default Behavior 48

7.5.Historical Resolution 48

7.6.Validation Responsibilities 49

7.7.Caching and Materialization 49

7.8.Inference and Augmentation Prohibitions 50

7.9.Transparency and Signaling 50

7.10.Prohibited Resolver Behaviors 51

8.Registrar Requirements 52

8.1.Purpose of This Section 52

8.2.Definitions (Section-Specific) 52

8.3.Registrar Accreditation 52

8.4.Node Identifier Issuance 53

8.5.Issuance Authority and Scope 53

8.6.Node Lifecycle Management **53**

8.7.Revocation and Retirement **54**

8.8.Registrar Neutrality and Separation of Concerns **54**

8.9.Prohibited Registrar Behaviors **55**

8.10.Node Policy Management **55**

9.LD-RAG (Linked Data Retrieval Augmented Generation) Requirements 56

9.1.Purpose of This Section **56**

9.2.Definitions (Section-Specific) **56**

9.3.Mandatory Retrieval Ordering **57**

9.4.Authority and Precedence Rules **57**

9.5.Provenance Preservation **58**

9.6.Inference Constraints **58**

9.7.Hallucination Control **59**

9.8.Multi-Ontology Handling **59**

9.9.Output Signaling Requirements **59**

9.10.Prohibited LD-RAG Behaviors **59**

10.Governance and Neutrality Doctrine 61

10.1.Purpose of This Section **61**

10.2.Neutrality Principles **61**

10.3.Publisher Sovereignty **62**

10.4.Ontology Neutrality **62**

10.5.Separation of Roles **62**

10.6.Federation and Non-Exclusivity **63**

10.7.Transparency and Auditability **63**

10.8.Update and Stewardship Model **63**

10.9.Prohibited Governance Outcomes **63**

11.Validation and Compliance Requirements 65

11.1.Purpose of This Section **65**

11.2.Conformance Claims **65**

11.3.Capsule Validation Requirements 66

11.4.Resolver Validation Responsibilities 66

11.5.Registrar Compliance Requirements 67

11.6.LD-RAG Compliance Requirements 67

11.7.Compliance Signaling 67

11.8.Non-Conformance Handling 68

11.9.Auditability and Verification 68

12.Security Requirements 69

12.1.Purpose of This Section 69

12.2.Integrity of Capsules and Metadata 69

12.3.Provenance Protection 69

12.4.Anti-Spoofing and Authenticity 70

12.5.Protection Against Silent Manipulation 70

12.6.Separation of Trust Domains 70

12.7.Denial-of-Service and Abuse Considerations 71

12.8.Cryptographic and Transport Non-Goals 71

12.9.Security Disclosure and Transparency 71

13.Performance and Scaling Considerations 72

13.1.Purpose of This Section 72

13.2.Scalability Assumptions 72

13.3.Resolver Performance Considerations 72

13.4.Storage and Materialization 73

13.5.Caching and Replication 73

13.6.Latency and Availability Expectations 73

13.7.Scaling of LD-RAG Systems 74

13.8.Non-Goals 74

14.Concluding Provisions 75

14.1.Purpose of This Section 75

14.2.Authority of the Standard 75

14.3.Interpretive Precedence 75
14.4.Scope Finality and Non-Goals 76
14.5.Evolution of the Standard 76
14.6.Conformance Continuity 76
14.7.Role of Appendices 77
14.8.Final Statement 77
15.Appendix A — Glossary 78
16.Appendix B — Normative Language Reference 83
17.Appendix C — Role Responsibility Matrix 85
18.Appendix D — Conformance Claim Examples 88
19.Appendix E — AgentNet Core Ontology (AN-O) 91
20.Appendix F — Policy Objects (Informative) 97
21.Appendix G —Resolution Envelope and Decision Signaling (Informative) 104
22.Appendix H — Standard Reason Codes (Informative) 109
23.Appendix I — LD-RAG Output Attribution Pattern (Informative)..........114

1. Document Status, Scope, and Normative Language

1.1.Document Status

This document defines AgentNet Standards (ANS) v2.0, a formal specification governing the structure, operation, and governance of the AgentNet ecosystem.

ANS vsupersedes all prior drafts, proposals, white papers, and informal descriptions of AgentNet protocols and SHALL be treated as the authoritative normative reference for all conforming implementations.

This document is intended for public distribution via agentnet.ai and for use by implementers, registrars, resolver operators, publishers, institutional participants, and standards organizations.

1.2.Scope of the Standard

ANS vspecifies:

- The **Capsule** data structure and its mandatory properties
- The **Node** identity model and lifecycle
- The obligations and constraints of Resolvers
- The obligations and accreditation requirements of Registrars
- The formal definition of **LD-RAG (Linked Data Retrieval Augmented Generation)**
- Governance, neutrality, and update mechanisms
- Validation, compliance, and security requirements

ANS vdoes **not** prescribe:

- Business models, pricing structures, or monetization mechanisms
- Licensing terms or intellectual property frameworks
- Jurisdiction-specific legal enforcement mechanisms
- Application-layer user interfaces or end-user experiences

Such concerns are addressed in companion governance, policy, or ecosystem documents and are intentionally excluded from this standard.

Clarification:

While ANS vdoes not prescribe business models, pricing structures, or monetization mechanisms, it *does* define how declared policy and economic terms**,** if present, are represented, inherited, validated, and audited to ensure deterministic machine behavior and interoperability.

1.3.Normative Language

The key words "MUST," "MUST NOT," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," and "MAY" in this document are to be interpreted as described in RFC 2119 and RFC 8174, when, and only when, they appear in all capitals.

Statements containing these terms define normative requirements. All other statements are informative and non-binding.

1.4.Architectural Principles

The AgentNet ecosystem is founded on the following principles:

- **Publisher Sovereignty.** Authoritative control over Capsules resides exclusively with the Publisher.
- **Ontology Neutrality.** No ontology, schema, or vocabulary SHALL be privileged, suppressed, or mandated by this standard.
- **Federation Over Centralization.** Identity issuance and data resolution SHALL support federated, multi-operator deployment.
- **Auditability and Transparency.** Identity, publication, and resolution events SHALL be observable, attributable, and verifiable.
- **Machine-First Design.** All specifications are optimized for deterministic machine consumption and automated reasoning.

1.5.Non-Exclusivity and Coexistence

ANS vis designed to coexist with existing web, data, and ontology standards.

Adoption of ANS v2.0:

- SHALL NOT require exclusive use of AgentNet mechanisms
- SHALL NOT preclude simultaneous use of alternative or complementary systems
- SHALL NOT invalidate or supersede existing standards by implication

ANS defines an interoperable framework intended to operate alongside established protocols, registries, and data ecosystems.

1.6.Versioning and Stability Expectations

ANS versions follow a major.minor.patch structure.

- Minor and patch revisions within the ANS v2.x series SHALL be backward compatible.
- Major revisions MAY introduce breaking changes and SHALL be accompanied by formal transition guidance and deprecation notice.

Implementers SHOULD track version identifiers explicitly and SHOULD NOT assume forward compatibility across major version boundaries.

1.7.Reserved Scope for Future Standards

ANS vreserves the right to define additional sections or companion standards in future versions, including but not limited to:

- Cryptographic attestation and signature frameworks
- Decentralized or verifiable identifier integration
- Automated compliance verification mechanisms
- Formal trust scoring or reputation signaling

The absence of such mechanisms in ANS vSHALL NOT be interpreted as preclusion of their future inclusion.

1.8.Conformance

An implementation SHALL be considered ANS v2.0–conformant only if it satisfies all applicable MUST and SHALL requirements defined in this document for the roles it claims to implement.

Partial, selective, or conditional compliance MUST NOT be represented as full conformance.

2.Terminology and Core Definitions

2.1.Purpose of This Section

This section defines the canonical terms used throughout ANS v2.0. All defined terms SHALL be interpreted consistently wherever they appear in this document.

Definitions in this section are normative unless explicitly stated otherwise.

2.2.AgentNet

AgentNet. A federated, machine-centric network for the publication, identification, and resolution of structured data artifacts, optimized for automated reasoning and AI system consumption.

AgentNet SHALL consist of Capsules, Nodes, Resolvers, Registrars, and Publishers operating under the constraints defined in this standard.

2.3.Capsule

Capsule. A structured, machine-readable data artifact, expressed as a JSON-LD document, that asserts factual claims about an entity, concept, event, or system component.

A Capsule:

- MUST conform to the Capsule Specification defined in Section 3
- MUST include mandatory metadata and provenance
- MUST bind to exactly one Node
- MUST be attributable to a Publisher
- MAY include multiple ontologies
- MAY include inferred data, subject to explicit annotation

A Capsule represents the atomic unit of authoritative information within AgentNet.

2.4.Node

Node. A persistent identity construct that provides stable reference across time, versions, and Capsules.

A Node:

- MUST be globally unique
- MUST be issued by an accredited Registrar
- MUST persist independently of any individual Capsule
- MAY bind to multiple Capsules over time
- MUST maintain historical associations even after retirement

Nodes serve as the identity backbone of AgentNet.

2.5.Publisher

Publisher. An entity that asserts authoritative control over one or more Capsules.

A Publisher:

- MUST be identifiable
- MUST be accountable for the accuracy of the Capsules it issues
- RETAINS sole authority to create, update, or retire its Capsules
- MAY delegate operational tasks without delegating authority

Publisher sovereignty is a foundational principle of AgentNet.

2.6.Resolver

Resolver. A service that retrieves Capsules associated with a given Node or query, subject to validation and authority rules.

A Resolver:

- MUST NOT modify Capsule contents
- MUST preserve metadata and provenance
- MUST return the latest non-revoked Capsule unless otherwise requested

- MAY cache results subject to expiration rules
- SHALL operate in a federated, non-exclusive manner

Resolvers function analogously to DNS resolvers, but for structured data.

2.7.Registrar

Registrar. An entity accredited to issue, manage, revoke, and retire Node identifiers.

A Registrar:

- MUST enforce global uniqueness of Node identifiers
- MUST maintain auditable issuance and revocation logs
- MUST operate under governance rules defined in Section 6
- MAY delegate issuance authority subject to oversight

Registrars form the trust rails of the AgentNet identity layer.

2.8.Ontology

Ontology. A formal, machine-interpretable schema that defines classes, properties, and semantic relationships.

In the context of ANS:

- Ontologies MAY be public, private, or domain-specific
- No ontology SHALL be privileged by this standard
- Multiple ontologies MAY coexist within a single Capsule

Ontology neutrality is mandatory.

2.9.Provenance

Provenance. Metadata that describes the origin, method of acquisition, authority, and evidentiary basis of data contained in a Capsule.

Provenance:

- MUST be explicit

- MUST distinguish authoritative data from inferred data
- MUST be preserved intact by Resolvers
- MUST support audit and traceability

Data without provenance SHALL NOT be treated as authoritative.

2.10.LD-RAG (Linked Data Retrieval Augmented Generation)

LD-RAG. A structured data–first AI reasoning model in which authoritative linked data is retrieved and validated prior to generative inference.

Within AgentNet:

- LD-RAG systems MUST prioritize Capsule data over latent model output
- LD-RAG systems MUST respect provenance and authority boundaries
- LD-RAG behavior is governed normatively in Section 7
- LD-RAG is a defining architectural capability of AgentNet.

2.11.Authority

Authority. The recognized right of a Publisher or Registrar to assert control over data or identity within AgentNet.

Authority:

- MUST be explicit
- MUST be verifiable
- MUST NOT be inferred by Resolvers or Consumers

Authority SHALL NOT be overridden by inference or aggregation.

2.12.Consumer

Consumer. Any system, application, or agent that retrieves and uses Capsules.

Consumers:

- MUST respect provenance and authority

- MUST NOT misrepresent inferred data as authoritative
- MAY apply additional logic or reasoning beyond the scope of this standard

Consumers are downstream participants and do not alter authoritative data.

2.13.Conformance Terminology

Conformant Implementation. An implementation that satisfies all applicable normative requirements for the roles it claims to perform.

Non-Conformant Implementation. An implementation that fails to satisfy one or more applicable MUST or SHALL requirements.

Misrepresentation of conformance status SHALL be considered a violation of this standard.

2.14.Deterministic

1) **Deterministic.**
   A property of a computation or evaluation such that, for a given set of explicit and versioned inputs, it produces a single, reproducible result that is invariant across time, implementations, execution environments, and operators, and does not rely on inference, heuristics, probabilistic methods, or implicit mutable state.

2.15.Policy Block

Policy Block.
A machine-readable object that declares provenance defaults, governance constraints, and optional economic terms applicable to a Node or Capsule, expressed explicitly and subject to versioning and audit.

2.16.Node Policy

Node Policy.
A Policy Block associated with a Node that defines default and baseline rules governing provenance, governance, and declared economic terms for all Capsules bound to that Node, subject to inheritance rules defined in this standard.

2.17.Capsule Policy

Capsule Policy.
A Policy Block associated with a specific Capsule version that refines, constrains, or (where explicitly permitted) overrides aspects of the Node Policy.

2.18.Effective Policy

Effective Policy.
The deterministically computed result of applying the inheritance rules defined in ANS to the Node Policy and Capsule Policy governing a specific Capsule at time of use.

2.19.Policy Inheritance Rules

Policy Inheritance Rules.
Normative rules defined by this standard that govern how Node Policy and Capsule Policy are combined to produce an Effective Policy.

3.System Architecture Overview

3.1.Purpose of This Section

This section defines the architectural model of the AgentNet ecosystem, including the relationships among Capsules, Nodes, Publishers, Resolvers, and Registrars.

This section is normative with respect to architectural constraints and informative with respect to illustrative flows. Detailed protocol behavior is specified in subsequent sections.

3.2.Architectural Model

AgentNet SHALL operate as a federated, machine-centric data network composed of the following primary components:

- Capsules — authoritative structured data artifacts
- Nodes — persistent identity anchors
- Publishers — authoritative data originators
- Resolvers — retrieval and validation services
- Registrars — identity issuance and trust authorities

These components SHALL interact according to the trust and authority rules defined in this document.

AgentNet SHALL support federated operation and SHALL NOT require permanent reliance on a centralized global database or single controlling operator.

3.3.Separation of Identity and Data

AgentNet SHALL enforce a strict separation between:

- Identity (Nodes)
- Data (Capsules)

Nodes provide durable identity over time, while Capsules provide versioned assertions bound to that identity.

A Node:

- SHALL exist independently of any specific Capsule
- SHALL persist across Capsule updates, corrections, and retirements

A Capsule:

- MUST bind to exactly one Node
- MUST NOT serve as an identity mechanism

This separation ensures long-term referential stability.

3.4.Authority Flow

Authority within AgentNet SHALL flow as follows:

- Registrars establish and manage Node identities.
- Publishers assert authoritative data through Capsules bound to Nodes.
- Resolvers retrieve and validate Capsules but do not assert authority.
- Consumers rely on provenance and authority metadata when using data.

Resolvers and Consumers SHALL NOT create, modify, or override authoritative assertions.

3.5.Federated Resolution Model

AgentNet SHALL support a federated resolution model analogous to DNS, in which:

- Multiple Resolvers MAY operate concurrently
- No Resolver SHALL be designated as globally exclusive
- Resolution MAY involve referral, delegation, or caching
- Authority SHALL remain traceable to the originating Publisher

Resolvers SHALL respect Registrar-issued identity boundaries and Publisher-issued data boundaries.

3.6.Trust Boundaries

The following trust boundaries SHALL be enforced:

- **Registrar Boundary.** Registrars are trusted only for identity issuance and lifecycle management.
- **Publisher Boundary.** Publishers are trusted only for the Capsules they issue.
- **Resolver Boundary.** Resolvers are trusted only for retrieval, validation, and delivery—not authorship.

Trust SHALL NOT be transitive beyond these boundaries.

3.7.Multi-Ontology Architecture

AgentNet SHALL support Contextual Multi-Ontology (CMO) operation as a first-class architectural feature.

A first-class architectural feature is a capability that is explicitly defined by this standard and whose correct implementation is mandatory for all systems claiming conformance, such that omission, degradation, or circumvention of the feature constitutes non-conformance.

Accordingly:

- Capsules MAY reference multiple ontologies simultaneously
- Ontology interpretation SHALL be explicit via declared contexts
- No Resolver SHALL suppress or reorder ontology contexts
- Consumers MAY choose how to interpret ontologies but SHALL NOT alter authoritative data

Ontology conflicts SHALL be resolved according to Publisher-declared priority rules (defined in Section 3).

3.8.Independence from Transport and Storage

ANS vintentionally avoids prescribing:

- Transport protocols (e.g., HTTP, HTTPS, message queues)
- Storage mechanisms (e.g., databases, object stores)
- Hosting models (e.g., cloud, on-premise, hybrid)

Implementers MAY select appropriate technologies provided all normative behaviors defined in this standard are preserved.

3.9.Architectural Constraints

Implementations claiming ANS vconformance:

- MUST preserve authority boundaries
- MUST preserve provenance without modification
- MUST support federation

- MUST avoid centralization by design
- MUST maintain deterministic behavior for machine consumption

Architectural shortcuts that violate these constraints SHALL be considered non-conformant.

4.Capsule Specification

4.1.Purpose of This Section

This section defines the Capsule as the fundamental data artifact of AgentNet and specifies the mandatory structural, semantic, provenance, and versioning requirements for all Capsules.

All requirements in this section are normative unless explicitly stated otherwise.

4.2.Definitions (Section-Specific)

Capsule. A structured, machine-readable data artifact, expressed as a JSON-LD document, that asserts authoritative factual claims and binds those claims to a persistent Node identity.

Capsule Version. A specific, immutable revision of a Capsule identified by a semantic version identifier.

Capsule Metadata. A required data block containing authority, provenance, versioning, and binding information for a Capsule.

Inference. Any data element not explicitly asserted by the Publisher and instead derived through automated or logical processes.

4.3.General Capsule Requirements

4.3.1.A Capsule MUST be represented as a valid JSON-LD document.

4.3.2.A Capsule MUST include the following top-level elements:

- @context
- @type
- metadata
- data

4.3.3.A Capsule MUST be attributable to exactly one Publisher.

4.3.4.A Capsule MUST bind to exactly one Node.

4.3.5.A Capsule MUST NOT contain runtime state, execution instructions, or non-deterministic values.

4.3.6.A Capsule MUST NOT obscure, omit, or falsify provenance or authority metadata.

4.3.7.A Capsule SHALL be treated as immutable once published, except through versioned replacement.

4.4.JSON-LD Context and Type Requirements

4.4.1.The @context field MUST declare one or more ontology IRIs that define the semantic interpretation of the Capsule.

4.4.2.A Capsule MAY declare multiple ontologies simultaneously.

4.4.3. A Capsule MAY include the AgentNet Core Ontology (AN-O) to express structural, provenance, lifecycle, and conformance semantics, as defined in Appendix E.

4.4.4.The order of declared contexts SHALL NOT imply precedence unless explicitly stated by the Publisher.

4.4.5.The @type field MUST identify the primary semantic class of the Capsule.

4.4.6.Nested or scoped @context declarations MUST be explicit and syntactically valid.

4.4.7.Resolvers MUST NOT reorder, suppress, or modify declared contexts.

4.5.Capsule Metadata Requirements

4.5.1.Every Capsule MUST include a metadata object.

4.5.2.The metadata object MUST include, at a minimum:

- Publisher identifier
- Publication timestamp
- Capsule version identifier
- Node binding identifier
- Provenance information

4.5.3. Capsule version identifiers MUST follow semantic versioning conventions (MAJOR.MINOR.PATCH).

4.5.4. The Node binding MUST reference a valid Node identifier issued by an accredited Registrar.

4.5.5. Metadata fields MUST NOT be inferred or synthesized by Resolvers.

4.5.6. Capsule metadata MUST include explicit policy linkage sufficient to compute Effective Policy, including at minimum:

- a reference to the governing Node Policy (URI or identifier and version or hash), and
- either an embedded Capsule Policy or a reference to one.

4.5.7. Capsule metadata MAY include an Effective Policy hash as an integrity signal but MUST NOT substitute this hash for required policy inputs.

4.6.Provenance Requirements

4.6.1.Provenance MUST be explicitly declared for all authoritative data elements.

4.6.2.Provenance MUST identify:

- The data source
- The method of acquisition
- The time of acquisition
- Whether the data is asserted or inferred

4.6.3.Provenance metadata MUST be preserved intact by all Resolvers and Consumers.

4.6.4.Data elements lacking provenance SHALL NOT be treated as authoritative.

4.6.5.Provenance MUST NOT be altered, normalized, or merged in a manner that obscures original attribution.

4.6.6.Provenance Inheritance. If an authoritative data element lacks element-level provenance, provenance MAY be inherited from:

- Capsule-level provenance defaults, else
- Node Policy provenance defaults.

4.6.7.Data elements lacking both explicit and inherited provenance SHALL NOT be treated as authoritative.

4.6.8.Inherited provenance MUST preserve attribution to its source level (Node or Capsule) and MUST NOT be flattened or obscured.

4.7.Versioning and Replacement Rules

4.7.1.Each Capsule version MUST be uniquely identifiable.

4.7.2.A new Capsule version MUST NOT retroactively alter historical facts without explicit correction metadata.

4.7.3.Breaking semantic changes MUST increment the MAJOR version.

4.7.4.Additive, non-breaking changes MUST increment the MINOR version.

4.7.5.Corrections and clerical fixes MUST increment the PATCH version.

4.7.6.Resolvers MUST return the latest non-revoked version unless otherwise requested.

4.8.Multi-Ontology and Contextual Operation

4.8.1.Capsules MAY reference multiple ontologies concurrently.

4.8.2.Property interpretation MUST be governed by the explicitly declared context.

4.8.3.Conflicting semantic interpretations MUST be resolved according to Publisher-declared rules.

4.8.4.In the absence of explicit conflict rules, Consumers MAY apply their own interpretation logic but MUST NOT alter the Capsule.

4.8.5.Multi-ontology support is a mandatory architectural capability for all conformant implementations.

4.9.Inference Constraints

4.9.1.Inference MAY be included only if explicitly identified as inferred.

4.9.2.Inferred data MUST NOT override authoritative Publisher assertions.

4.9.3.Inference metadata MUST identify:

- Source elements used
- Inference rules applied
- Confidence level

4.9.4.Resolvers MUST NOT introduce new inferred data.

4.9.5.Consumers MUST NOT represent inferred data as authoritative.

4.10.Prohibited Capsule Behaviors

A Capsule MUST NOT:

- Contain executable code
- Contain mutable state
- Depend on undisclosed external computation
- Assert authority it does not possess
- Mask inferred data as asserted fact

Violation of these constraints constitutes non-conformance.

4.11.Provenance, Governance, and Economics Inheritance

4.11.1.Determinism Requirement. Effective Policy computation MUST be deterministic.

4.11.2.Default Inheritance. Unless explicitly overridden as permitted, Capsules bound to a Node MUST inherit provenance defaults, governance rules, and declared economic terms from the Node Policy.

4.11.3.No Silent Loosening. A Capsule Policy MUST NOT loosen governance or economic constraints defined in Node Policy unless the Node Policy explicitly permits that specific override.

4.11.4.Most-Restrictive-Wins Rule. Where policy provisions conflict and no explicit override permission exists, the Effective Policy SHALL select the most restrictive applicable rule.

4.11.5.Capsule Tightening Permitted. Capsule Policy MAY impose stricter governance or economic requirements than Node Policy.

4.11.6.Preservation of Policy Sources. Resolvers and Consumers MUST preserve references to both Node Policy and Capsule Policy sufficient to independently recompute Effective Policy.

Informative Note:

Capsules are treated as authoritative or non-authoritative based on policy, validation, and the lifecycleState of the bound Node, not by any Capsule lifecycle state.

5.Node Specification

5.1.Purpose of This Section

This section defines the Node as the persistent identity construct within AgentNet and specifies requirements for Node structure, issuance, lifecycle management, and binding to Capsules.

All requirements in this section are normative unless explicitly stated otherwise.

5.2.Definitions (Section-Specific)

Node. A persistent, globally unique identity construct that serves as a stable reference point for one or more Capsules over time.

Node Identifier (Node ID). A unique identifier assigned to a Node by an accredited Registrar.

Node Lifecycle State. The operational status of a Node, including Active, Inactive, and Retired states.

5.3.Node General Requirements

5.3.1.A Node MUST be globally unique.

5.3.2.A Node MUST be issued by an accredited Registrar in accordance with Section 6.

5.3.3.A Node MUST exist independently of any specific Capsule.

5.3.4.A Node MUST be persistent across Capsule updates, replacements, and retirements.

5.3.5.A Node MUST NOT embed mutable data, factual assertions, or semantic claims.

5.3.6.A Node SHALL function solely as an identity anchor.

5.4.Node Identifier Requirements

5.4.1.Each Node MUST be assigned exactly one Node Identifier.

5.4.2.Node Identifiers MUST be globally unique across all Registrars.

5.4.3.Node Identifiers MUST be stable and non-reusable for the lifetime of the Node.

5.4.4.Node Identifiers MUST NOT encode semantic meaning.

5.4.5.Node Identifiers MUST be generated exclusively by Registrar-controlled systems using deterministic or cryptographically suitable processes.

5.4.6.Human-supplied, suggested, requested, or operator-influenced identifier values SHALL NOT be accepted, incorporated, or translated, directly or indirectly, into a Node Identifier.

5.4.7.Requestors, Publishers, and Consumers SHALL NOT have the ability to influence the content, structure, or format of a Node Identifier.

5.4.8.Node Identifiers MUST NOT be reassigned to a different Node under any circumstances.

5.4.9.The format and generation mechanism for Node Identifiers SHALL be enforced uniformly by the issuing Registrar and SHALL ensure identifier opacity.

Informative Note:

Node Identifiers are intentionally opaque to prevent semantic inference, preserve ontology neutrality, and ensure long-term stability independent of changing classifications or meanings. All semantic interpretation is expressed exclusively through Capsules.

5.5.Node Lifecycle States

5.5.1.A Node MUST exist in exactly one of the following lifecycle states at any given time:

- Active
- Inactive
- Retired
- Revoked

5.5.2.An Active Node is eligible for new Capsule bindings.

5.5.3.An Inactive Node is temporarily suspended and MUST NOT accept new Capsule bindings.

5.5.4.A Retired Node is permanently closed and MUST NOT accept new Capsule bindings.

5.5.5.Transition of a Node between lifecycle states MUST be performed by the issuing Registrar.

*5.5.6.*All lifecycle transitions MUST be logged and auditable.

Informative Note:

Lifecycle State applies to Nodes, not Capsules. Capsules do not possess lifecycleState under ANS v2.0. Resolver selection constraints based on lifecycleState SHALL be evaluated against the bound Node.

5.6.Node–Capsule Binding Rules

5.6.1.Each Capsule MUST bind to exactly one Node.

5.6.2.A Node MAY bind to multiple Capsules over time.

5.6.3.Capsule bindings MUST preserve historical ordering and association.

5.6.4.Neither Nodes nor Capsules SHALL be deleted once issued; lifecycle state changes MUST be used to indicate operational status.

5.6.5.A change in lifecycle state MUST NOT remove or invalidate historical Node–Capsule associations.

5.6.6.Resolvers that apply lifecycle constraints MUST evaluate those constraints against the lifecycleState of the bound Node. A Resolver MUST NOT claim that a Capsule is "Active," "Inactive," "Retired," or "Revoked" as a lifecycleState under ANS v2.0.

5.6.7.If a Node is Inactive, Retired, or Revoked, Resolvers MUST treat all Capsules bound to that Node as non-authoritative for current use, except where the Resolver's response explicitly indicates historical retrieval and the Consumer has requested or accepted such retrieval.

5.6.8.Historical retrieval may include Capsules bound to Nodes that are Inactive, Retired, or Revoked, provided the Resolver response explicitly discloses that the bound Node is not Active and that the returned Capsules are presented for historical or archival purposes.

5.6.9.Node–Capsule binding records MUST be auditable and sufficient to establish the Node associated with each returned Capsule and the bound Node's lifecycleState at time of resolution.

Informative Note:

Logical persistence of Node–Capsule bindings does not imply continuous inclusion in active resolver indexes or caches. Implementations may employ tiered storage, archival systems, or delayed materialization provided historical bindings remain retrievable when explicitly requested.

5.7.Node Merging and Splitting

5.7.1.Node merging and Node splitting SHALL NOT occur as part of normal operation.

5.7.2.Node merging and Node splitting MAY occur only as exceptional corrective actions under Registrar-controlled procedures.

5.7.3.Only the issuing Registrar SHALL have authority to initiate or approve a Node merge or split.

5.7.4.Node merging MUST NOT result in the loss, deletion, or reassignment of historical Node Identifiers.

5.7.5.When Nodes are merged, all original Node Identifiers MUST remain resolvable and MUST clearly indicate their merged status.

5.7.6.Node splitting MUST NOT retroactively alter historical Capsule bindings.

5.7.7.Node merging or splitting MUST NOT rewrite, invalidate, or obscure historical provenance or authority claims.

5.7.8.All Node merge or split actions MUST be explicitly logged, timestamped, and auditable.

5.7.9.The absence of a standardized merge or split procedure SHALL NOT be interpreted as permission for arbitrary or automated identity modification.

Informative Note:

Node merging and splitting are intended solely to correct issuance errors or identity misassignment and are not intended as data normalization, deduplication, or ontology reconciliation mechanisms.

5.8.Node Revocation and Retirement

5.8.1.Node revocation MUST be performed only by the issuing Registrar.

5.8.2.Revocation MUST include a timestamp and reason code.

5.8.3.Retired Nodes MUST remain resolvable for historical reference.

5.8.4.Revocation MUST NOT erase or obscure historical Capsule associations.

5.9.Node Resolution Expectations

5.9.1.Resolvers MUST be able to resolve Node Identifiers to associated Capsules.

5.9.2.Resolution of a retired Node MUST indicate its retired status.

5.9.3.Node resolution MUST NOT imply endorsement of the bound Capsules.

5.10.Prohibited Node Behaviors

A Node MUST NOT:

- Contain factual assertions
- Contain ontology references
- Be repurposed after issuance
- Be deleted without trace
- Be controlled by a non-accredited entity

Any such behavior constitutes non-conformance.

5.11.Node Policy Requirements

5.11.1.Every Node MUST have an associated Node Policy.

5.11.2.Node Policy MUST be:

  - explicitly retrievable,
  - versioned, and
  - subject to immutable audit logging.

5.11.3. Node Policy MUST support, at minimum:

- provenance defaults,
- governance constraints, and
- optional declared economic terms.

5.11.4. Node Policy MUST explicitly declare whether Capsule-level overrides are permitted and under what conditions.

5.11.5. Node Policy MUST NOT be inferred, synthesized, or assumed by Resolvers or Consumers.

6.AgentNet Inquiry and Resolution Process

6.1.Purpose and Scope

This section defines the AgentNet inquiry process by which a Consumer submits a resolution request and a Resolver determines the authoritative Capsule(s) eligible to satisfy that request.

This section applies to all implementations claiming conformance with AgentNet Standards (ANS) — Core Specification v2.0, regardless of Consumer type (Agent, software system, or application).

This section specifies:

- how inquiry intent is expressed,
- how subject matter is identified,
- how resolution eligibility is determined,
- and how authority boundaries are enforced.

6.2.Inquiry Model Overview

AgentNet inquiries are declarative and constraint-based.

An inquiry:

- SHALL express what is being requested in terms of subject identity and scope,
- SHALL NOT express how to find it,
- SHALL NOT require interpretation, inference, or probabilistic ranking by the Resolver.

Resolvers operate by evaluating declared capsule metadata against inquiry constraints, not by searching content or inferring meaning.

6.3.Consumer Role in Inquiry Submission

A **Consumer** is any entity that submits an inquiry to an AgentNet Resolver.

A Consumer MAY be:

- an Agent acting on behalf of an LLM or autonomous system,
- a traditional software system,

- a human-facing application or service.

The Consumer role is technology-agnostic and is defined solely by participation in the resolution process.

6.4.Inquiry Structure and Semantics

6.4.1.Declarative Inquiry Requirements

An AgentNet inquiry:

- SHALL be expressed as a structured, machine-readable object,
- SHALL specify a subject identity,
- MAY specify one or more required authority scopes,
- MAY specify policy constraints governing selection.

An inquiry SHALL NOT:

- contain natural-language questions,
- rely on keyword matching,
- request interpretation or inference by the Resolver.

6.4.2.Subject Identification

Each inquiry SHALL identify the subject of interest using a canonical identifier.

The identifier:

- SHALL uniquely identify a single work, entity, or object,
- SHALL be treated as an opaque value,
- SHALL be compared by exact match only.

Resolvers SHALL NOT:

- infer subject identity from descriptive fields,
- guess identity based on similarity or popularity,
- accept ambiguous or fuzzy subject identifiers.

If no authoritative identifier can be determined, the inquiry SHALL be considered invalid for resolution.

6.4.3.Authority Scope Constraints

An inquiry MAY specify one or more required authority scopes, indicating the nature of assertions being requested (e.g., rights, licensing, metadata).

Resolvers SHALL:

- consider only Capsules whose declared authority scope satisfies the inquiry,
- exclude Capsules asserting unrelated or insufficient scopes.

Authority scope matching SHALL be explicit and deterministic.

6.5.Identity Grounding and Responsibility

6.5.1.Identity Determination

Identity grounding occurs prior to resolution.

The determination of a subject identifier:

- SHALL be performed by the Consumer or its delegated Agent,
- SHALL rely on authoritative identity systems when available,
- SHALL NOT be delegated to the Resolver.

Resolvers SHALL assume that the subject identifier provided is intentional and authoritative.

6.5.2.Identifier Format

AgentNet does not prescribe a universal identifier format.

Identifiers:

- MAY originate from external registries,
- MAY be AgentNet-native when no authoritative registry exists,
- SHALL be treated as opaque values by all resolvers.

Identifier validity is determined by authority and context, not by string structure.

6.6. Resolver Eligibility Evaluation

Resolvers SHALL determine resolution eligibility by evaluating inquiry constraints against declared Capsule metadata.

This evaluation SHALL include, at minimum:

- Subject identity match
- Ontology compatibility
- Authority scope compatibility
- Publisher trust and policy constraints
- Capsule status and validity

Resolvers SHALL NOT:

- search unstructured content,
- rank results by relevance,
- infer missing declarations.

If no Capsule satisfies all required constraints, the Resolver SHALL return a non-resolution result indicating that no authoritative declaration exists.

6.7. Determinism and Auditability

Given identical inquiry inputs and policy conditions, a Resolver:

- SHALL return a consistent resolution result,
- SHALL be able to explain why a Capsule was selected or excluded,
- SHALL support post-hoc audit of the resolution decision.

Resolvers SHOULD provide a resolution envelope containing:

- the selected Capsule identifier,
- applied policies,
- trust or eligibility rationale.

6.8.Separation of Concerns

The AgentNet inquiry process enforces strict separation of responsibilities:

Function	Responsible Role
Intent interpretation	LLM or upstream system
Identity grounding	Agent or Consumer
Resolution eligibility	Resolver
Assertion authority	Publisher

Resolvers SHALL NOT assume responsibilities assigned to other roles.

6.9.Relationship to LD-RAG Systems

AgentNet inquiries are designed to support Linked-Data Retrieval-Augmented Generation (LD-RAG) systems.

- In such systems:
- Natural-language interpretation occurs upstream,
- Structured inquiries are submitted to the Resolver,
- Authoritative Capsules are returned,
- Reasoning occurs only over resolved declarations.

This separation ensures that reasoning systems operate over authoritative, provenance-bound data rather than inferred or scraped content.

6.10.Non-Resolution as a Valid Outcome

The absence of an eligible Capsule is a valid and meaningful resolution outcome.

Resolvers SHALL explicitly distinguish between:

- lack of authoritative declaration, and
- technical or operational errors.

Consumers SHOULD treat non-resolution results as authoritative statements about the state of published knowledge.

6.11.Conformance

An implementation claiming ANS-Core vconformance:

- SHALL implement the inquiry model defined in this section,
- SHALL NOT substitute heuristic or inferential mechanisms for declared resolution,
- SHALL preserve the determinism and auditability guarantees specified herein.

7.Resolver Requirements

7.1.Purpose of This Section

This section defines the mandatory obligations, constraints, and behaviors of Resolvers operating within the AgentNet ecosystem.

Resolvers are responsible for retrieval, validation, and delivery of authoritative Capsules but do not possess authority to create, modify, infer, or reinterpret data.

All requirements in this section are normative unless explicitly stated otherwise.

7.2.Definitions (Section-Specific)

Resolver. A service that retrieves Capsules associated with a Node Identifier or other supported query mechanism, subject to authority, provenance, and lifecycle constraints.

Default Resolution. The standard resolution behavior in which only current, authoritative Capsule bindings are returned.

Historical Resolution. An explicit resolution mode in which historical, inactive, or retired Capsule bindings are requested and returned.

7.3.General Resolver Obligations.

7.3.1.A Resolver MUST operate in a federated, non-exclusive manner.

7.3.2.A Resolver MUST NOT assert authority over any Capsule or Node.

7.3.3.A Resolver MUST NOT modify Capsule contents, metadata, provenance, or declared contexts.

7.3.4.A Resolver MUST preserve Capsule integrity exactly as published.

7.3.5.A Resolver MUST respect Registrar-issued Node identity boundaries.

7.3.6.A Resolver MUST include the lifecycleState of the bound Node for each returned Capsule (or for the target Node when returning a single Capsule), sufficient for Consumers to evaluate authoritativeness.

7.4.Resolution Scope and Default Behavior

7.4.1.When resolving Capsules for authoritative use, Resolvers MUST exclude Capsules bound to Nodes whose lifecycleState is Inactive, Retired, or Revoked, unless the Consumer explicitly requests historical or archival retrieval.

7.4.2.When a Resolver returns Capsules bound to a Node that is Inactive, Retired, or Revoked, the Resolver response MUST disclose the bound Node's lifecycleState and MUST NOT represent the returned Capsules as authoritative for current use.

7.4.3.Default resolution responses MUST represent the current authoritative state and MUST NOT include historical bindings unless explicitly requested.

7.4.4.The absence of a Capsule from a default resolution response SHALL NOT be interpreted as deletion, invalidation, or non-existence.

7.5.Historical Resolution

7.5.1.A Resolver MUST support explicit historical resolution requests.

7.5.2.Historical resolution MUST be opt-in and MUST NOT be triggered implicitly.

7.5.3.When historical resolution is requested, a Resolver MUST return:

- lifecycle status,
- version identifiers,
- provenance metadata, and
- ordering information.

7.5.4.Historical resolution responses MUST clearly distinguish authoritative assertions from inferred data.

7.5.5.A Resolver SHALL NOT filter, collapse, or normalize historical results in a manner that obscures sequence or attribution.

7.6.Validation Responsibilities

7.6.1.A Resolver MUST validate that returned Capsules:

- are well-formed JSON-LD,
- include required metadata,
- bind to a valid Node, and
- include provenance information.

7.6.2.A Resolver MUST NOT return Capsules that fail mandatory validation checks as authoritative.

7.6.3.Validation failures MUST be clearly indicated in Resolver responses.

7.7.Caching and Materialization

7.7.1.A Resolver MAY cache resolution results.

7.7.2.Cached results MUST be invalidated when:

- a new Capsule version is published,
- lifecycle status changes, or
- authority is revoked.

7.7.3.A Resolver SHALL NOT be required to cache or index historical Capsules.

7.7.4.A Resolver SHALL NOT be required to maintain full historical materialization in active storage.

7.7.5.Tiered, cold, or archival storage MAY be used provided historical data remains retrievable upon explicit request.

7.8.Inference and Augmentation Prohibitions

7.8.1.A Resolver MUST NOT introduce inferred data.

7.8.2.A Resolver MUST NOT merge, synthesize, or reconcile Capsules.

7.8.3.A Resolver MUST NOT override Publisher assertions.

7.8.4.Any inference or augmentation SHALL occur only downstream, outside the Resolver role.

7.9.Transparency and Signaling

7.9.1.Resolver responses MUST clearly signal:

- lifecycle filtering,
- validation status, and
- authority boundaries.

7.9.2.Resolver responses MUST NOT imply endorsement or verification beyond validation performed.

7.9.3.Errors, omissions, or filtering behavior MUST be explicitly indicated.

7.9.4.Resolver responses MUST signal the policy basis governing returned Capsules, including references to:

- Node Policy version, and
- Capsule Policy (if present).

7.9.5.Resolvers MUST NOT invent, repair, infer, or relax policy terms. Failure to compute Effective Policy MUST be surfaced explicitly.

7.10.Prohibited Resolver Behaviors

A Resolver MUST NOT:

- Act as a Publisher or Registrar
- Modify Capsule content or metadata
- Inject inferred or speculative data
- Hide lifecycle state transitions
- Require exclusive control or central authority

Any such behavior constitutes non-conformance.

8.Registrar Requirements

8.1.Purpose of This Section

This section defines the mandatory obligations, authorities, and constraints applicable to Registrars operating within the AgentNet ecosystem.

Registrars are responsible exclusively for Node identity issuance and lifecycle management and SHALL NOT exercise authority over Capsule content.

All requirements in this section are normative unless explicitly stated otherwise.

8.2.Definitions (Section-Specific)

Registrar. An accredited entity authorized to issue, manage, revoke, and retire Node Identifiers in accordance with this standard.

Registration Engine. A Registrar-controlled system responsible for generating and assigning Node Identifiers.

Accreditation. The formal recognition that a Registrar satisfies governance, operational, and audit requirements defined by ANS.

8.3.Registrar Accreditation

8.3.1.An entity MUST be accredited before operating as a Registrar.

8.3.2.Accreditation MUST require demonstrated capability to:

- generate globally unique Node Identifiers,
- enforce identifier opacity,
- maintain audit logs, and
- comply with governance obligations.

8.3.3.Accreditation MAY be revoked for non-conformance.

8.3.4.Accreditation status MUST be publicly verifiable.

8.4.Node Identifier Issuance

8.4.1.Registrars MUST generate Node Identifiers exclusively through Registrar-controlled systems.

8.4.2.Node Identifiers MUST be system-generated, opaque, and non-semantic.

8.4.3.Human-supplied, suggested, requested, or operator-influenced identifier values SHALL NOT be accepted, incorporated, or translated, directly or indirectly.

8.4.4.Registrars MUST ensure global uniqueness of all issued Node Identifiers.

8.4.5.Registrars MUST prevent reuse or reassignment of Node Identifiers.

8.4.6.Registrars SHALL enforce uniform identifier generation rules across all issuance events.

8.5.Issuance Authority and Scope

8.5.1.Registrars MUST issue Node Identifiers only within their accredited scope.

8.5.2.Registrars MUST NOT issue Node Identifiers on behalf of unverified entities.

8.5.3.Delegation of issuance authority MAY occur only under explicit Registrar control and audit.

8.5.4.Delegated issuance MUST NOT relax any requirements of this standard.

8.6.Node Lifecycle Management

8.6.1.Registrars MUST manage Node lifecycle states (Active, Inactive, Retired).

8.6.2.Lifecycle state transitions MUST be explicit, timestamped, and auditable.

8.6.3.Registrars MUST NOT delete Nodes once issued.

8.6.4.Revocation or retirement MUST NOT erase historical Node–Capsule associations.

8.7.Revocation and Retirement

8.7.1.Registrars MUST provide mechanisms to revoke or retire Nodes.

8.7.2.Revocation actions MUST include:

- timestamp,
- reason code, and
- authority reference.

8.7.3.Retired Nodes MUST remain resolvable for historical reference.

8.7.4.Revocation MUST NOT be used to rewrite history or obscure provenance.

8.7.5.Audit and Logging Requirements

8.7.6.Registrars MUST maintain immutable audit logs of:

- Node issuance,
- lifecycle transitions,
- revocations, and
- merge/split actions.

8.7.7.Audit logs MUST be retained for the lifetime of the Node.

8.7.8.Audit logs MUST be accessible for compliance review.

8.7.9.Failure to maintain auditability constitutes non-conformance.

8.8.Registrar Neutrality and Separation of Concerns

8.8.1.Registrars MUST NOT assert authority over Capsule content.

8.8.2.Registrars MUST NOT modify, filter, or interpret Capsule data.

8.8.3.Registrars MUST remain ontology-neutral.

8.8.4.Registrars SHALL NOT perform Resolver functions, including Capsule retrieval, validation, caching, filtering, or delivery.

8.9.Prohibited Registrar Behaviors

A Registrar MUST NOT:

- Allow human influence over Node Identifier content
- Delete issued Node Identifiers
- Reassign Node Identifiers
- Conceal revocation or lifecycle events
- Privilege specific Publishers or ontologies

Any such behavior constitutes non-conformance.

Informative Note:

Privileging specific Publishers or ontologies may occur through preferential identity issuance, selective delays or scrutiny, conditional acceptance based on content category or ontology usage, or implicit requirements to adopt particular schemas, vocabularies, or affiliated services. Registrars are expected to remain content-agnostic and ontology-agnostic, issuing and managing Node Identifiers without regard to Publisher identity, intended use, or semantic classification. This prohibition exists to prevent gatekeeping, structural bias, and downstream distortion of authority or resolution behavior.

8.10.Node Policy Management

8.10.1.Registrars MUST support creation, retrieval, and versioned update of Node Policy.

8.10.2.Node Policy updates MUST be append-only and fully auditable, including timestamps and reason codes.

8.10.3.Registrars MUST make Node Policy retrievable in a manner suitable for Resolver validation.

8.10.4.Registrars MUST NOT mandate specific economic models, payment rails, or pricing structures.

9.LD-RAG (Linked Data Retrieval Augmented Generation) Requirements

9.1.Purpose of This Section

This section defines the mandatory requirements for systems that perform Linked Data Retrieval Augmented Generation (LD-RAG) using AgentNet Capsules.

LD-RAG governs how authoritative structured data is retrieved, validated, and incorporated into generative or reasoning processes. These requirements exist to prevent hallucination, authority inversion, and provenance loss.

All requirements in this section are normative unless explicitly stated otherwise.

9.2.Definitions (Section-Specific)

LD-RAG (Linked Data Retrieval Augmented Generation). A data-first AI reasoning model in which authoritative, provenance-backed structured data is retrieved and validated prior to generative inference or synthesis.

Retrieval Phase. The process by which Capsules are resolved, validated, and selected for use.

Generation Phase. The process by which outputs are produced using retrieved data, potentially combined with model-internal reasoning.

9.3.Mandatory Retrieval Ordering

AgentNet data MAY be claimed as authoritative input only under the following conditions:

9.3.1. Authoritative Capsules MUST be retrieved through a conformant Resolver prior to their use as authoritative input in any generative, inferential, or reasoning process.

9.3.2. Retrieved Capsules MUST be validated for structural integrity, provenance completeness, and lifecycle status prior to use.

9.3.3. Authoritative Capsule data MUST be incorporated as primary factual grounding and MUST take precedence over latent model knowledge.

9.3.4. Where authoritative Capsules are available and applicable, failure to retrieve them prior to claiming authoritative input SHALL constitute non-conformance.

Informative Note:

This requirement does not constrain internal model architecture or reasoning strategies. It defines the minimum retrieval precondition under which AgentNet data may be claimed as authoritative input for systems asserting LD-RAG compliance.

9.4.Authority and Precedence Rules

9.4.1.Authoritative Capsule data MUST take precedence over model-generated content.

9.4.2.Where Capsule data exists, LD-RAG systems MUST NOT contradict, override, or reinterpret authoritative assertions.

9.4.3.In the event of conflicting authoritative Capsules, LD-RAG systems MUST preserve and expose the conflict rather than resolving it implicitly.

9.4.4.Authority boundaries defined by Publisher provenance MUST be respected at all times.

9.5.Provenance Preservation

9.5.1.LD-RAG systems MUST preserve provenance metadata associated with all retrieved Capsule data.

9.5.2.Generated outputs MUST retain traceability to source Capsules when authoritative facts are included.

9.5.3.Provenance MUST NOT be collapsed, abstracted, or omitted in a manner that obscures source attribution.

9.6.Inference Constraints

9.6.1.LD-RAG systems MAY perform inference only after authoritative retrieval has occurred.

9.6.2.Inferred content MUST be clearly distinguishable from authoritative Capsule assertions.

9.6.3.LD-RAG systems MUST NOT present inferred content as authoritative.

9.6.4.Inference MUST NOT be used to fill gaps where authoritative data is absent unless explicitly labeled as inference.

Informative Note:

The inference constraints in this section apply to inference performed during LD-RAG execution. These constraints do not prohibit the use of data that has been inferred upstream by a Capsulizer or Publisher and subsequently published within a Capsule with explicit inference metadata and provenance. Such data, while inferred, is considered authoritative-with-qualification for the purposes of LD-RAG.

9.7. Hallucination Control

9.7.1. LD-RAG systems MUST NOT fabricate facts, relationships, or attributes in contradiction to Capsule data.

9.7.2. Where authoritative data is incomplete or unavailable, LD-RAG systems MUST explicitly indicate uncertainty.

9.7.3. Silence or omission MUST be preferred over fabrication.

9.8. Multi-Ontology Handling

9.8.1. LD-RAG systems MUST respect all declared ontologies within a Capsule.

9.8.2. Ontology context boundaries MUST be preserved during reasoning and output generation.

9.8.3. LD-RAG systems MUST NOT collapse multi-ontology data into a single semantic frame without explicit disclosure.

9.9. Output Signaling Requirements

9.9.1. LD-RAG outputs MUST clearly distinguish:

- authoritative Capsule data,
- inferred content, and
- model-generated synthesis.

9.9.2. Outputs MUST NOT imply authority beyond that provided by source Capsules.

9.9.3. Any aggregation, summarization, or transformation MUST preserve factual accuracy.

9.10. Prohibited LD-RAG Behaviors

An LD-RAG system MUST NOT:

- Bypass retrieval in favor of latent knowledge
- Override authoritative Capsule data
- Mask inference as fact

- Suppress conflicts or uncertainty
- Discard provenance

Any such behavior constitutes non-conformance.

10.Governance and Neutrality Doctrine

10.1.Purpose of This Section

This section defines the governance principles and neutrality obligations that apply to all participants in the AgentNet ecosystem.

These principles ensure that AgentNet remains a neutral, federated, and non-exclusive infrastructure standard, suitable for global adoption and long-term stewardship.

All requirements in this section are normative unless explicitly stated otherwise.

10.2.Neutrality Principles

10.2.1.AgentNet SHALL operate as a neutral infrastructure standard.

10.2.2.No single entity SHALL exercise unilateral control over identity issuance, data publication, resolution behavior, or ontology selection.

10.2.3.Governance mechanisms MUST prevent structural favoritism, gatekeeping, or exclusionary practices.

10.2.4.Neutrality SHALL be preserved across:

- Publishers
- Registrars
- Resolvers
- Ontologies
- Consumers

10.3. Publisher Sovereignty

10.3.1. Publishers SHALL retain sole authority over the Capsules they issue.

10.3.2. No Registrar, Resolver, or Consumer MUST alter, reinterpret, suppress, or override Publisher assertions.

10.3.3. Publisher authority MUST be explicit and verifiable through provenance metadata.

10.3.4. Publisher sovereignty SHALL NOT be diluted by aggregation, inference, or downstream transformation.

10.4. Ontology Neutrality

10.4.1. ANS SHALL NOT mandate, endorse, or privilege any ontology, schema, or vocabulary.

10.4.2. Registrars and Resolvers MUST remain ontology-agnostic.

10.4.3. Capsules MAY reference any ontology, including public, private, proprietary, or domain-specific ontologies.

10.4.4. Ontology conflicts MUST be preserved and exposed rather than resolved implicitly.

10.5. Separation of Roles

10.5.1. Registrar, Resolver, Publisher, and Consumer roles MUST remain functionally distinct.

10.5.2. An entity MAY operate multiple roles only if each role's obligations are independently satisfied and clearly segregated.

10.5.3. No role SHALL exercise the authority reserved to another role.

10.5.4. Combined-role implementations MUST NOT compromise neutrality, auditability, or authority boundaries.

10.6. Federation and Non-Exclusivity

10.6.1. AgentNet SHALL support federated operation.

10.6.2. No *specific* Resolver or Registrar SHALL be required for global participation.

10.6.3. Adoption of ANS SHALL NOT require exclusive reliance on AgentNet infrastructure.

10.6.4. Coexistence with other standards, registries, and data systems SHALL be permitted.

10.7. Transparency and Auditability

10.7.1. Governance decisions MUST be transparent and auditable.

10.7.2. Identity issuance, lifecycle transitions, and revocations MUST be logged.

10.7.3. Resolution behavior MUST be observable and explainable.

10.7.4. Silent modification or suppression of authoritative data MUST NOT occur.

10.8. Update and Stewardship Model

10.8.1. ANS SHALL be maintained through an open, documented update process.

10.8.2. Material changes MUST undergo public review.

10.8.3. Backward compatibility SHOULD be preserved within major versions.

10.8.4. Governance authority SHALL NOT be used to introduce economic, licensing, or competitive constraints into the standard.

10.9. Prohibited Governance Outcomes

The following outcomes MUST NOT occur under ANS governance:

- Centralized control of identity or resolution
- Mandatory ontologies or schemas
- Exclusive Resolver or Registrar requirements
- Hidden policy enforcement

- Undisclosed conflicts of interest

Any such outcome constitutes a violation of this doctrine.

11.Validation and Compliance Requirements

11.1.Purpose of This Section

This section defines the validation, conformance, and compliance requirements applicable to all implementations claiming adherence to ANS v2.0.

Compliance with this section is mandatory for any entity or system asserting ANS vconformance for one or more roles defined in this standard.

All requirements in this section are normative unless explicitly stated otherwise.

11.2.Conformance Claims

11.2.1.An implementation SHALL NOT claim ANS vconformance unless it satisfies all applicable requirements for the roles it performs.

11.2.2.Conformance claims MUST specify the roles implemented, including but not limited to:

- Publisher
- Registrar
- Resolver
- LD-RAG system

11.2.3.Conformance semantics MAY be expressed using the AgentNet Core Ontology (AN-O), as defined in Appendix E.

11.2.4.Partial or selective conformance MUST NOT be represented as full conformance.

11.2.5.Misrepresentation of conformance status SHALL constitute non-conformance.

11.3.Capsule Validation Requirements

11.3.1.Capsules MUST be validated for:

- structural correctness,
- required fields,
- declared contexts,
- provenance completeness, and
- Node binding integrity.

11.3.2.Capsules that fail mandatory validation MUST NOT be treated as authoritative.

11.3.3.Validation MUST distinguish between:

- authoritative assertions, and
- inferred data elements.

11.3.4.Validation failures MUST be detectable and reportable.

11.4.Resolver Validation Responsibilities

11.4.1.Resolvers MUST validate Capsules prior to delivery as authoritative data.

11.4.2.Resolvers MUST clearly signal validation status in resolution responses.

11.4.3.Resolvers MUST NOT suppress validation errors or warnings.

11.4.4.Resolvers SHALL NOT return invalid Capsules as authoritative.

11.5.Registrar Compliance Requirements

11.5.1.Registrars MUST comply with:

- identity issuance rules,
- lifecycle management rules,
- audit and logging requirements, and
- neutrality obligations.

11.5.2.Registrars MUST be able to demonstrate compliance upon request.

11.5.3.Failure to maintain auditability SHALL constitute non-conformance.

11.6.LD-RAG Compliance Requirements

11.6.1.Systems claiming LD-RAG compliance MUST adhere to Section 7.

11.6.2.LD-RAG compliance MUST include:

- retrieval-before-claiming-authority,
- authority precedence,
- provenance preservation,
- inference signaling.

11.6.3.LD-RAG systems MUST NOT claim authoritative grounding when requirements are not met.

11.7.Compliance Signaling

11.7.1.Implementations SHOULD provide explicit compliance signaling, including:

- supported roles,
- supported sections,
- and known limitations.

11.7.2.Compliance signaling MUST NOT be misleading.

11.7.3.Absence of compliance signaling SHALL NOT imply conformance.

11.8.Non-Conformance Handling

11.8.1.Non-conformance MUST be correctable.

11.8.2.Implementations MAY temporarily suspend conformance claims while remediation occurs.

11.8.3.Persistent non-conformance SHALL invalidate conformance claims.

11.8.4.This standard DOES NOT prescribe penalties, enforcement mechanisms, or dispute resolution procedures.

11.9.Auditability and Verification

11.9.1.Conformance MUST be verifiable through observable behavior.

11.9.2.Audit logs, validation signals, and resolution outputs MUST support independent verification.

11.9.3.Hidden or unverifiable compliance claims SHALL be considered invalid.

12.Security Requirements

12.1.Purpose of This Section

This section defines the security requirements applicable to all AgentNet participants and implementations claiming conformance with ANS v2.0.

These requirements focus on integrity, authenticity, provenance protection, and abuse prevention, while intentionally avoiding prescriptive cryptographic or key-management mandates.

All requirements in this section are normative unless explicitly stated otherwise.

12.2.Integrity of Capsules and Metadata

12.2.1.Capsules MUST be protected against unauthorized modification.

12.2.2.Capsule content, metadata, provenance, and declared contexts MUST be delivered intact by Resolvers.

12.2.3.Any detected alteration of Capsule content MUST be treated as a validation failure.

12.2.4.Integrity verification MAY be implemented through checksums, hashes, or equivalent mechanisms.

12.2.5.Failure to preserve integrity SHALL constitute non-conformance.

12.3.Provenance Protection

12.3.1.Provenance metadata MUST be preserved without alteration.

12.3.2.Systems MUST NOT suppress, normalize, or collapse provenance in a manner that obscures attribution.

12.3.3.Provenance MUST remain traceable to the authoritative Publisher.

12.3.4.Loss or corruption of provenance SHALL invalidate authoritative claims.

12.4.Anti-Spoofing and Authenticity

12.4.1.Systems MUST prevent spoofing of:

- Node Identifiers,
- Publisher identity,
- Registrar authority.

12.4.2.Capsules MUST NOT be represented as authoritative unless their Node binding and Publisher authority can be verified.

12.4.3.Resolvers MUST reject Capsules with unverifiable or conflicting authority claims.

12.4.4.Authenticity checks MAY rely on Registrar-maintained records, provenance validation, or equivalent mechanisms.

12.5.Protection Against Silent Manipulation

12.5.1.Silent modification, substitution, or suppression of authoritative data MUST NOT occur.

12.5.2.Systems MUST surface lifecycle state changes, validation failures, and authority boundaries explicitly.

12.5.3.Aggregation, summarization, or transformation MUST NOT obscure original authoritative assertions.

12.6.Separation of Trust Domains

12.6.1.Identity issuance, data publication, resolution, and consumption MUST remain logically separated trust domains.

12.6.2.Compromise of one domain MUST NOT automatically compromise others.

12.6.3.Implementations SHOULD minimize cross-domain trust dependencies.

12.7.Denial-of-Service and Abuse Considerations

12.7.1.Implementations SHOULD employ reasonable measures to mitigate abuse, including excessive resolution requests or malformed Capsules.

12.7.2.Abuse mitigation MUST NOT compromise neutrality or authority boundaries.

12.7.3.Security controls MUST NOT be used as a mechanism for content censorship or preferential treatment.

12.8.Cryptographic and Transport Non-Goals

12.8.1.ANS vDOES NOT mandate specific cryptographic algorithms, key management systems, or transport protocols.

12.8.2.Implementers MAY employ cryptographic signatures, encryption, or secure transport as appropriate.

12.8.3.The absence of cryptographic mechanisms SHALL NOT imply non-conformance provided integrity and authenticity requirements are met.

12.9.Security Disclosure and Transparency

12.9.1.Security-relevant failures SHOULD be observable and diagnosable.

12.9.2.Hidden or undisclosed security-affecting behavior SHALL be considered non-conformant.

12.9.3.Transparency SHALL take precedence over silent failure modes.

13.Performance and Scaling Considerations

13.1.Purpose of This Section

This section describes performance, scalability, and operational considerations relevant to implementations of ANS v2.0.

Except where explicitly stated, the contents of this section are informative and SHALL NOT be interpreted as normative requirements or conditions of conformance.

13.2.Scalability Assumptions

13.2.1.AgentNet is designed to scale to:

- large numbers of Nodes,
- high volumes of Capsules,
- diverse ontologies, and
- heterogeneous Consumers.

13.2.2.The standard ASSUMES that implementations will employ distributed and federated architectures to achieve scalability.

13.2.3.No single implementation is expected to service all resolution or issuance requests globally.

13.3.Resolver Performance Considerations

13.3.1.Default resolution behavior is expected to return only current, authoritative Capsule bindings.

13.3.2.Historical resolution is expected to occur less frequently and MAY incur higher latency.

13.3.3.Implementations MAY employ caching strategies provided:

- integrity is preserved,
- lifecycle changes invalidate caches,
- and authority boundaries are maintained.

13.3.4.Performance optimizations MUST NOT compromise correctness or neutrality.

13.4.Storage and Materialization

13.4.1.ANS vDOES NOT REQUIRE:

- hot storage of all historical Capsules,
- continuous indexing of retired data, or
- uniform storage strategies across implementations.

13.4.2.Tiered storage, cold storage, and archival systems are compatible with ANS vprovided historical data remains retrievable upon explicit request.

13.4.3.Logical persistence MUST NOT be conflated with operational materialization.

13.5.Caching and Replication

13.5.1.Replication and caching MAY be used to improve availability and latency.

13.5.2.Replication MUST NOT introduce divergence from authoritative data.

13.5.3.Cached data MUST be invalidated upon:

- Capsule version changes,
- lifecycle state transitions,
- authority revocations.

13.6.Latency and Availability Expectations

13.6.1.ANS vDOES NOT prescribe latency, throughput, or availability targets.

13.6.2.Implementations MAY offer different performance characteristics based on deployment context.

13.6.3.Performance variability SHALL NOT be interpreted as non-conformance.

13.7.Scaling of LD-RAG Systems

13.7.1.LD-RAG systems are expected to perform retrieval as a discrete, bounded operation.

13.7.2.Retrieval-first requirements DO NOT imply synchronous blocking behavior in all architectures.

13.7.3.Implementations MAY employ asynchronous, pre-fetch, or batched retrieval strategies provided authority precedence is preserved.

13.8.Non-Goals

13.8.1.ANS vDOES NOT mandate:

- specific performance benchmarks,
- infrastructure sizing,
- cloud or on-premise deployment models, or
- cost characteristics.

13.8.2.Performance optimization is an implementation concern, not a standards concern.

14.Concluding Provisions

14.1.Purpose of This Section

This section provides the formal closing provisions of ANS v2.0, including statements of authority, scope finality, interpretive precedence, and the role of appendices.

14.2.Authority of the Standard

14.2.1.ANS vconstitutes the authoritative specification governing the AgentNet ecosystem.

14.2.2.In the event of conflict between this document and prior drafts, white papers, or informal descriptions, this document SHALL take precedence.

14.2.3.No external document, implementation detail, or marketing material SHALL be construed to modify the requirements of this standard.

14.3.Interpretive Precedence

14.3.1.Normative sections of this document SHALL take precedence over informative sections.

14.3.2.In the event of conflict between normative requirements and illustrative examples, normative requirements SHALL prevail.

14.3.3.Informative notes and examples DO NOT create obligations and SHALL NOT be used to infer conformance.

14.4.Scope Finality and Non-Goals

14.4.1.ANS vdefines a protocol and governance framework, not an application, product, or service.

14.4.2.This standard DOES NOT define:

- business models or monetization schemes,
- licensing or intellectual property frameworks,
- jurisdictional enforcement mechanisms,
- end-user interfaces or experiences.

14.4.3.The absence of such definitions SHALL NOT be interpreted as endorsement or prohibition.

14.5.Evolution of the Standard

14.5.1.ANS vis intended to evolve through an open, documented governance process.

14.5.2.Future revisions MAY:

- clarify requirements,
- add sections, or
- introduce companion standards.

14.5.3.Backward compatibility SHOULD be preserved within major versions.

14.5.4.Material changes MUST be clearly identified and publicly documented.

14.6.Conformance Continuity

14.6.1.Implementations conformant with ANS v2.0 SHALL remain conformant until:

- the implementation changes, or
- the standard is superseded by a future major version.

14.6.2. No entity SHALL be required to upgrade solely due to publication of minor or patch revisions.

14.7. Role of Appendices

14.7.1. Appendices are provided to support interpretation, clarity, and reference.

14.7.2. Unless explicitly stated, appendices are informative and DO NOT introduce new normative requirements.

14.7.3. Appendix A (Glossary) provides canonical definitions to support consistent interpretation.

14.8. Final Statement

ANS vestablishes a neutral, federated, and authoritative foundation for machine-centric data exchange and AI-grounded reasoning. Its effectiveness depends on strict adherence to authority boundaries, provenance integrity, and role separation as defined herein.

15.Appendix A — Glossary

AgentNet
A federated, machine-centric network for the publication, identification, and resolution of authoritative structured data artifacts used for automated reasoning and AI systems.

ANS (AgentNet Standards)
The formal specification defining the architecture, governance, and operational requirements of the AgentNet ecosystem.

Authoritative Data
Data asserted by a Publisher through a Capsule with explicit provenance and authority metadata.

Authority
The recognized right of an entity to assert control over identity (Registrar) or data (Publisher) within AgentNet.

Capsule
A structured, machine-readable data artifact, expressed as a JSON-LD document, that binds authoritative or inferred-with-provenance assertions to a Node.

Capsulizer
A system or process that discovers, extracts, structures, or infers data and publishes it as Capsule content under Publisher authority.

Compliance
The state of satisfying all applicable normative requirements of ANS vfor the roles an implementation claims to perform.

Conformance
A verifiable claim that an implementation meets all mandatory requirements of ANS vfor specified roles.

Consumer
Any system, application, or agent that retrieves and uses Capsules without asserting authority over their content.

Contextual Multi-Ontology (CMO)
An architectural capability allowing a Capsule to reference multiple ontologies simultaneously with explicit context declaration and without enforced semantic collapse.

Default Resolution
Resolver behavior that returns only current, authoritative Capsule bindings based on lifecycle state.

Federation
An architectural model in which multiple independent operators provide identity issuance and resolution without centralized control or mandatory exclusivity.

First-Class Architectural Feature
A capability explicitly defined by this standard whose correct implementation is mandatory for conformance and whose omission constitutes non-conformance.

Historical Resolution
An explicit resolution mode that returns inactive or retired Capsule bindings for audit, verification, or temporal analysis.

Inference
Derivation of data not explicitly asserted by a source, performed either upstream (Capsulizer inference) or downstream (runtime model inference).

Inferred Data
Data included in a Capsule that is derived rather than directly asserted, accompanied by explicit inference metadata and provenance.

LD-RAG (Linked Data Retrieval Augmented Generation)
An AI reasoning model in which authoritative structured data is retrieved, validated, and incorporated prior to generative inference.

Lifecycle State
A state of a Node indicating operational status, including Active, Inactive, Retired, and Revoked (if defined by this standard). Lifecycle State does not apply to Capsules under ANS v2.0.

Node
A persistent, globally unique identity construct that serves as a stable reference point for one or more Capsules over time.

Node Identifier (Node ID)
A system-generated, opaque identifier issued by a Registrar to uniquely identify a Node.

Normative
Describing requirements that are mandatory for conformance, typically expressed using terms such as MUST, SHALL, or MUST NOT.

Ontology
A formal schema defining classes, properties, and semantic relationships used to interpret structured data.

Opaque Identifier
An identifier that carries no embedded semantic meaning and cannot be interpreted without external context.

Provenance
Metadata describing the origin, authority, method of acquisition, and derivation of data contained in a Capsule.

Publisher
An entity that asserts authoritative control over Capsule content and is responsible for its accuracy and provenance.

Registrar
An accredited entity authorized to issue, manage, revoke, and retire Node Identifiers.

Registration Engine
A Registrar-controlled system responsible for generating and assigning Node Identifiers.

Resolver
A service that retrieves, validates, and delivers Capsules associated with a Node or query, without asserting authority.

Retrieval Phase
The stage in LD-RAG in which authoritative Capsules are resolved and validated prior to inference or generation.

System-Generated Identifier
An identifier created exclusively by software processes without human-supplied or semantic input.

Validation
The process of verifying structural correctness, provenance completeness, authority, and lifecycle status of Capsules.

16.Appendix B — Normative Language Reference

B.1 Purpose

This appendix defines the interpretation of normative language used throughout AgentNet Standards (ANS) v2.0.

The intent of this appendix is to ensure consistent understanding of mandatory, recommended, and optional requirements by implementers, reviewers, and governance bodies.

B.2 Normative Keywords

The key words "MUST," "MUST NOT," "SHALL," "SHALL NOT," "SHOULD," "SHOULD NOT," and "MAY" in this document are to be interpreted as described in RFC 2119 and RFC 8174, when, and only when, they appear in all capital letters.

These terms have the following general meanings:

- MUST / SHALL
  An absolute requirement of this standard.
- MUST NOT / SHALL NOT
  An absolute prohibition of this standard.
- SHOULD / SHOULD NOT
  A strong recommendation; valid reasons may exist to deviate, but the implications must be fully understood and carefully weighed.
- MAY
  An optional feature or behavior.

B.3 Normative vs. Informative Content

Sections, clauses, and statements designated as normative define requirements that are binding for conformance with ANS v2.0.

Content designated as informative, including explanatory notes, examples, and appendices not explicitly marked as normative:

- DO NOT introduce new requirements
- DO NOT modify normative obligations
- DO NOT create conformance expectations

In the event of any conflict, normative text SHALL take precedence over informative text.

B.4 Capitalization Convention

Normative keywords are capitalized intentionally to signal binding requirements.

Lowercase usage of the same words is non-normative and SHALL NOT be interpreted as conveying mandatory behavior.

B.5 Scope of Applicability

Normative requirements apply only to the roles and behaviors explicitly addressed by the relevant section of the standard.

An implementation is required to satisfy only those normative requirements applicable to the roles it claims to perform.

17.Appendix C — Role Responsibility Matrix

C.1 Purpose

This appendix provides a concise, role-oriented summary of responsibilities, prohibitions, and authority boundaries defined normatively in Sections 3 through 9 of ANS v2.0.

The purpose of this matrix is to:

- clarify separation of concerns,
- reduce misinterpretation of role boundaries,
- and assist implementers in scoping conformance claims.

This appendix is informative and does not introduce new normative requirements.

C.2 Role Definitions (Summary)

The AgentNet ecosystem defines the following primary roles:

- **Publisher** — Asserts authoritative Capsule content
- **Registrar** — Issues and manages Node identities
- **Resolver** — Retrieves and validates Capsules
- **LD-RAG System** — Performs reasoning using retrieved data
- **Consumer** — Uses Capsule data without asserting authority

C.3 Responsibility Matrix

Function / Capability	Publisher	Registrar	Resolver	LD-RAG System	Consumer
Issue Node Identifiers	❌	✅	❌	❌	❌
Manage Node Lifecycle	❌	✅	❌	❌	❌
Publish Capsule Content	✅	❌	❌	❌	❌
Modify Capsule Content	✅ (own Capsules only)	❌	❌	❌	❌
Retrieve Capsules	❌	❌	✅	❌	❌
Validate Capsules	❌	❌	✅	❌	❌
Cache or Materialize Capsules	❌	❌	✅	❌	❌
Perform Runtime Inference	❌	❌	❌	✅ (bounded)	❌
Introduce New Inferred Data	❌	❌	❌	❌	❌
Use Capsule Data	❌	❌	❌	✅	✅
Assert Authority Over Data	✅	❌	❌	❌	❌

C.4 Interpretation Notes

- A checkmark (✅) indicates that the function is permitted or required for that role, subject to applicable normative constraints.
- A cross (❌) indicates that the function is prohibited or outside the scope of that role.
- “Bounded” inference refers to inference performed in accordance with **Section 7 (LD-RAG Requirements)**, including authority precedence and provenance signaling.

- Combined-role implementations are permitted only if all role boundaries and obligations are independently satisfied.

C.5 Use of This Matrix

This matrix is intended to be used as:

- a quick reference for implementers,
- a compliance scoping aid,
- and a governance clarity tool.

In the event of any discrepancy between this matrix and normative sections of ANS v2.0, the normative sections SHALL take precedence.

18.Appendix D — Conformance Claim Examples

D.1 Purpose

This appendix provides illustrative examples of acceptable and unacceptable conformance claims under ANS v2.0.

The intent is to:

- clarify how conformance claims should be scoped,
- prevent misleading or overstated claims,
- and assist implementers, reviewers, and consumers in interpreting compliance statements.

This appendix is informative and does not introduce new normative requirements.

D.2 General Guidance

- Conformance claims MUST be scoped to the specific roles implemented.
- Partial conformance MUST NOT be represented as full conformance.
- Absence of a claim SHALL NOT imply conformance.
- Examples below are illustrative and non-exhaustive.

Normative requirements governing conformance are defined in Section 9.

D.3 Examples of Acceptable Conformance Claims

Example D.— Resolver-Only Conformance

“This system is conformant with ANS vin the role of Resolver only.”

Explanation:
The claim is role-scoped and does not imply Registrar, Publisher, or LD-RAG conformance.

Example D.— Publisher Conformance

“This Publisher conforms to ANS vfor Capsule creation, metadata, provenance, and versioning requirements.”

Explanation:
The claim correctly limits scope to Publisher responsibilities.

Example D.— LD-RAG System Conformance

“This system performs LD-RAG in conformance with ANS vSection 7, including retrieval-before-authority claims and provenance preservation.”

Explanation:
The claim references specific obligations and does not assert authority over identity or resolution.

Example D.— Multi-Role Conformance

“This implementation is conformant with ANS vas both a Publisher and LD-RAG System. Registrar and Resolver roles are not implemented.”

Explanation:
Multi-role claims are acceptable when roles are explicitly enumerated and segregated.

D.4 Examples of Unacceptable or Misleading Claims

Example D.— Overbroad Claim

“This platform is fully ANS-compliant.”

Why unacceptable:
Fails to specify roles; implies full ecosystem conformance without substantiation.

Example D.— Implicit Authority Claim

“ANS-compliant answers generated from AgentNet data.”

Why unacceptable:
Implies authoritative grounding without specifying LD-RAG compliance or retrieval behavior.

Example D.— Partial Compliance Misrepresentation

“Supports ANS Capsules.”

Why unacceptable:
Support for data formats alone does not constitute conformance.

Example D.— Resolver Authority Overreach

“Our Resolver validates and corrects Publisher data.”

Why unacceptable:
Resolvers must not modify or reinterpret authoritative data.

D.5 Recommended Claim Structure

Conformance claims SHOULD, at a minimum, specify:

- Implemented role(s)
- Applicable ANS version
- Known exclusions or limitations

Example Template

“This system claims conformance with ANS vfor the following roles: [roles]. No other ANS roles are implemented.”

D.6 Interpretation Note

This appendix is intended to improve clarity and prevent misuse of conformance language. In all cases, normative requirements in Section 9 govern actual conformance status, regardless of how claims are phrased.

19. Appendix E — AgentNet Core Ontology (AN-O)

E.1 Purpose and Scope

This appendix defines the AgentNet Core Ontology (AN-O), a minimal, shared semantic vocabulary used to describe the structural, authoritative, and lifecycle aspects of the AgentNet ecosystem, including Nodes, Capsules, operational roles, provenance, inference classification, validation, and conformance signaling.

AN-O is structural, not domain-specific. It does not model subject-matter data (e.g., creative works, healthcare records, products). Such data SHALL be expressed using appropriate external ontologies.

E.2 Ontology Neutrality and Coexistence

The AgentNet Core Ontology does not replace or supersede any external ontology.

Capsules MAY include:

- The AgentNet Core Ontology (AN-O)
- One or more vertical or domain ontologies
- Publisher-defined or third-party vocabularies

AN-O provides governance, provenance, and lifecycle semantics only and SHALL NOT constrain domain expression.

E.3 Normative Ontology Definition (Turtle / OWL)

The normative semantic definition of the AgentNet Core Ontology is expressed using Turtle (TTL) with OWL/RDFS semantics.

This Turtle definition SHALL be considered the authoritative source for:

- Class and property meaning
- Domain and range expectations
- Lifecycle and assertion semantics
- Reasoning, validation, and governance

E.AgentNet Core Ontology — Turtle Definition

```
@prefix an:    <https://agentnet.ai/ontology/core#> .

@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix xsd:   <http://www.w3.org/2001/XMLSchema#> .

an:AgentNetCoreOntology a owl:Ontology ;
  rdfs:label "AgentNet Core Ontology (AN-O)" ;
  rdfs:comment "Structural ontology for AgentNet nodes, capsules,
provenance, lifecycle, policy linkage, and conformance." .

##### Core Classes

an:Node a owl:Class ; rdfs:label "Node" .
an:Capsule a owl:Class ; rdfs:label "Capsule" .

an:Publisher a owl:Class ; rdfs:label "Publisher" .
an:Registrar a owl:Class ; rdfs:label "Registrar" .
an:Resolver a owl:Class ; rdfs:label "Resolver" .
an:Consumer a owl:Class ; rdfs:label "Consumer" .

an:ProvenanceRecord a owl:Class ; rdfs:label "Provenance Record" .
an:InferenceRecord a owl:Class ; rdfs:label "Inference Record" .
an:ValidationReport a owl:Class ; rdfs:label "Validation Report" .
an:ConformanceClaim a owl:Class ; rdfs:label "Conformance Claim" .

an:Policy a owl:Class ;
  rdfs:label "Policy" ;
  rdfs:comment "A machine-readable policy object governing provenance
defaults, governance constraints, or declared economic terms." .

an:NodePolicy a owl:Class ;
  rdfs:subClassOf an:Policy ;
  rdfs:label "Node Policy" .

an:CapsulePolicy a owl:Class ;
  rdfs:subClassOf an:Policy ;
  rdfs:label "Capsule Policy" .

an:EffectivePolicy a owl:Class ;
  rdfs:subClassOf an:Policy ;
  rdfs:label "Effective Policy" .

##### Enumerations

an:LifecycleState a owl:Class ; rdfs:label "Lifecycle State" .
an:Active a an:LifecycleState .
an:Inactive a an:LifecycleState .
an:Retired a an:LifecycleState .
an:Revoked a an:LifecycleState .

an:AssertionKind a owl:Class ; rdfs:label "Assertion Kind" .
an:Asserted a an:AssertionKind
```

E.4 JSON-LD Context (Derived, Informative)

The following JSON-LD context is derived from the Turtle ontology and is provided to support Capsule serialization and exchange.

JSON-LD contexts SHALL be interpreted as equivalent to the Turtle ontology only insofar as they faithfully reflect its semantics.
In the event of conflict or ambiguity, the Turtle ontology SHALL prevail.

E.AgentNet Core JSON-LD Context

```
{

  "@context": {
    "an": "https://agentnet.ai/ontology/core#",
    "xsd": "http://www.w3.org/2001/XMLSchema#",

    "@vocab": "https://agentnet.ai/ontology/core#",

    "Node": "an:Node",
    "Capsule": "an:Capsule",
    "Publisher": "an:Publisher",
    "Registrar": "an:Registrar",
    "Resolver": "an:Resolver",
    "Consumer": "an:Consumer",

    "ProvenanceRecord": "an:ProvenanceRecord",
    "InferenceRecord": "an:InferenceRecord",
    "ValidationReport": "an:ValidationReport",
    "ConformanceClaim": "an:ConformanceClaim",

    "Policy": "an:Policy",
    "NodePolicy": "an:NodePolicy",
    "CapsulePolicy": "an:CapsulePolicy",
    "EffectivePolicy": "an:EffectivePolicy",

    "Active": "an:Active",
    "Inactive": "an:Inactive",
    "Retired": "an:Retired",
    "Revoked": "an:Revoked",

    "Asserted": "an:Asserted",
    "Inferred": "an:Inferred",

    "nodeId": { "@id": "an:nodeId", "@type": "xsd:string" },
    "capsuleId": { "@id": "an:capsuleId", "@type": "xsd:string" },

    "bindsToNode": { "@id": "an:bindsToNode", "@type": "@id" },
    "publishedBy": { "@id": "an:publishedBy", "@type": "@id" },
    "issuedBy": { "@id": "an:issuedBy", "@type": "@id" },

    "hasNodePolicy": { "@id": "an:hasNodePolicy", "@type": "@id" },
    "hasCapsulePolicy": { "@id": "an:hasCapsulePolicy", "@type": "@id" },
    "hasEffectivePolicy": { "@id": "an:hasEffectivePolicy", "@type":
"@id" },
    "policyEvaluationStatus": { "@id": "an:policyEvaluationStatus", "@type":
"xsd:string" },

    "publishedAt": { "@id": "an:publishedAt", "@type": "xsd:dateTime" },
    "updatedAt": { "@id": "an:updatedAt", "@type": "xsd:dateTime" },

    "version": { "@id": "an:version", "@type": "xsd:string" },

    "lifecycleState": { "@id": "an:lifecycleState", "@type": "@id" },
    "lifecycleChangedAt": { "@id": "an:lifecycleChangedAt", "@type":
```

E.5 Authority and Non-Requirements

- Turtle (TTL) is normative
- JSON-LD is derived and operational
- No participant is required to author Turtle
- No specific RDF tooling is mandated

This appendix exists to ensure that AgentNet semantics are explicit, inspectable, and governed, while preserving operational simplicity.

20. Appendix F — Policy Objects (Informative)

F.1 Purpose

This appendix provides illustrative examples of Policy Objects referenced normatively in Sections 3, 4, 5, and 6 of ANS v2.0.

The purpose of this appendix is to:

- clarify the expected shape and content of Policy Objects,
- demonstrate inheritance and override behavior, and
- support consistent implementation across Registrars, Resolvers, and Consumers.

This appendix is informative and does not introduce new normative requirements. In the event of any conflict between this appendix and normative sections of ANS v2.0, the normative sections SHALL prevail.

F.2 Policy Object Characteristics

Policy Objects share the following general characteristics:

- Machine-readable and explicit
- Versioned and auditable
- Free of inference or implicit defaults
- Suitable for deterministic evaluation

Policy Objects MAY be:

- embedded directly within metadata, or
- referenced by stable identifiers (e.g., URI + version or hash)
- The mechanism of storage, transport, or retrieval is intentionally unspecified.

F.3 Node Policy Example (Informative)

The following example illustrates a Node Policy defining provenance defaults, governance constraints, and declared economic terms applicable to all Capsules bound to the Node.

```json
{

  "policyType": "NodePolicy",
  "policyVersion": "1.0.0",
  "issuedAt": "2025-12-15T00:00:00Z",

  "provenanceDefaults": {
    "publisherId": "agentnet://publisher/example-co",
    "assertionKind": "asserted",
    "method": "publisher-attested",
    "inferenceAllowed": false
  },

  "governance": {
    "auditRequired": true,
    "auditRetentionDays": 3650,
    "emergencyChangesPostHocDisclosure": true,
    "overridePermissions": {
      "capsuleMayLoosenGovernance": false,
      "capsuleMayLoosenEconomics": false
    }
  },

  "economics": {
    "resolverFee": {
      "amount": "0.0001",
      "currency": "USD",
      "payer": "requester"
    },
    "publisherRoyalty": {
      "amount": "0.0003",
      "currency": "USD",
      "payer": "requester"
    }
  }
}
```

Informative Notes:

- Economic fields are declarative only and do not mandate settlement mechanisms.
- Absence of an economics block implies no declared economic terms.
- Override permissions are explicit and default to restrictive behavior when omitted.

F.4 Capsule Policy Example (Informative)

The following example illustrates a Capsule Policy that tightens governance and economic terms relative to the governing Node Policy.

```
{

  "policyType": "CapsulePolicy",
  "policyVersion": "1.0.0",
  "appliesToCapsuleVersion": "2.1.0",

  "governance": {
    "auditRequired": true,
    "auditRetentionDays": 7300
  },

  "economics": {
    "publisherRoyalty": {
      "amount": "0.0005",
      "currency": "USD",
      "payer": "requester"
    }
  }
}
```

Informative Notes:

- This Capsule Policy does not loosen any Node Policy constraints.
- Increased audit retention and higher royalty are permitted under the “tightening allowed” rule.
- Omitted fields inherit deterministically from Node Policy.

F.5 Effective Policy (Illustrative Result)

The following illustrates the Effective Policy resulting from deterministic inheritance of the Node Policy (F.3) and Capsule Policy (F.4).

```
          {

           "effectivePolicyVersion": "derived",
           "nodePolicyVersion": "1.0.0",
           "capsulePolicyVersion": "1.0.0",

           "provenance": {
            "publisherId": "agentnet://publisher/example-co",
            "assertionKind": "asserted",
            "method": "publisher-attested",
            "inferenceAllowed": false
           },

           "governance": {
            "auditRequired": true,
            "auditRetentionDays": 7300,
            "emergencyChangesPostHocDisclosure": true
           },

           "economics": {
            "resolverFee": {
             "amount": "0.0001",
             "currency": "USD",
             "payer": "requester"
            },
            "publisherRoyalty": {
             "amount": "0.0005",
             "currency": "USD",
             "payer": "requester"
            }
           }
}
```

Informative Notes:

- Effective Policy is shown here for clarity; Resolvers are not required to materialize it.
- Any conforming implementation given the same inputs MUST compute an equivalent result.
- Effective Policy is derived, not authored.

F.6 Prohibited Policy Patterns (Informative)

The following patterns are non-conformant and shown for illustrative purposes only:

- Implicit defaults not declared in Node or Capsule Policy
- Capsule Policies that silently loosen governance or economic constraints
- Resolver-synthesized policy fields
- Time- or context-dependent policy evaluation
- Policies requiring human interpretation to resolve conflicts

Such patterns violate determinism, auditability, or authority boundaries defined normatively in ANS v2.0.

F.7 Relationship to Registrar and Resolver Implementations

- This appendix is intended to guide, not constrain, implementations:
- Registrars may store, version, and expose Policy Objects in any suitable manner.
- Resolvers may compute Effective Policy or validate inputs without materializing it.
- Consumers may rely on policy signaling without enforcing settlement or governance.

All such behavior is governed normatively elsewhere in this document.

F.8 Final Informative Note

Policy Objects exist to ensure that provenance, governance, and declared economics are explicit, inherited, and machine-enforceable, without introducing inference, discretion, or centralized control. This appendix demonstrates how such objects may look, not how they must be implemented.

21.Appendix G —Resolution Envelope and Decision Signaling (Informative)

G.1 Purpose and Scope

This appendix describes an illustrative resolution envelope that MAY be used by Resolvers to communicate resolution outcomes, validation status, and selection rationale in a structured, machine-auditable manner.

This appendix is informative and non-normative.
Nothing in this appendix alters, expands, or constrains the normative requirements defined elsewhere in ANS v2.0.

G.2 Motivation

ANS v2.0 defines strict behavioral requirements for Resolvers, including neutrality, authority preservation, lifecycle handling, and validation. However, the Core specification intentionally avoids prescribing:

- Resolver APIs
- Transport protocols
- Serialization formats
- Response schemas

This appendix provides a common signaling pattern to improve interoperability, auditability, and transparency without mandating implementation details.

Use of a resolution envelope is OPTIONAL.

G.3 Conceptual Model

A resolution envelope separates what was requested, what was decided, and what was validated, without modifying or obscuring authoritative Capsule data.

Conceptually, a resolution envelope may contain:

1. Resolution Context
   What Node or identifier was requested, and under what mode or constraints.
2. Decision Outcome
   Which Capsule (if any) was selected and why.
3. Validation Signals
   Whether integrity, authority, and lifecycle checks passed or failed.
4. Payload
   The resolved Capsule or reference to it.

Resolvers MUST continue to comply with all normative rules regarding neutrality, provenance preservation, and authority boundaries regardless of whether an envelope is used.

G.4 Illustrative Resolution Envelope Structure

The following example is illustrative only and does not imply a required field set, naming convention, or JSON structure.

```
{

  "resolution": {
    "target": "agentnet://node/example-001",
    "mode": "default",
    "filters": [
      "active-node",
      "non-revoked"
    ]
  },
  "decision": {
    "selectedCapsule": "agentnet://capsule/example-001@1.2.0",
    "reason": "latest-authoritative"
  },
  "validation": {
    "status": "pass",
    "checks": [
      "jsonld-valid",
      "authority-confirmed",
      "provenance-present",
      "lifecycle-eligible"
    ]
  },
  "capsule": {
    "...": "authoritative capsule content"
  }
}
```

G.5 Resolution Modes (Illustrative)

Resolvers MAY expose or internally support different resolution modes, such as:

- **default**
  Return the latest non-revoked Capsule bound to an active Node.
- **historical**
  Resolve a specific Capsule version or historical state.
- **policy-filtered**
  Apply explicit policy constraints declared by the Consumer.

The definition, naming, and availability of modes are implementation choices and are not prescribed by ANS v2.0.

G.6 Decision Reason Signaling

Resolvers MAY signal a human- or machine-readable explanation for selection decisions.

Examples include (non-exhaustive):

- latest-authoritative
- explicit-version-request
- policy-exclusion-applied
- node-inactive
- capsule-revoked

Reason signaling improves auditability and debugging but SHALL NOT be used to imply authority, reinterpret data, or override Publisher assertions.

G.7 Validation Signaling

Resolvers MAY surface validation outcomes to enable independent verification.

Validation signaling MAY include:

- Overall status (e.g., pass / fail)

- Individual check results
- References to Registrar or Publisher verification sources

Validation signaling MUST NOT obscure, normalize, or collapse authoritative provenance.

G.8 Relationship to Conformance

Use or non-use of a resolution envelope **does not affect conformance** with ANS v2.0.

Conformance is determined solely by adherence to the normative requirements defined in the Core specification, not by response formatting or signaling conventions.

G.9 Non-Goals

This appendix does NOT:

- Define a Resolver API
- Mandate a wire format
- Require JSON usage
- Specify authentication mechanisms
- Introduce new authority semantics

Those concerns are intentionally left to implementers, companion specifications, or ecosystem tooling.

G.10 Summary

The resolution envelope pattern provides a **shared mental model** for transparent, auditable resolution behavior while preserving the architectural neutrality and minimalism of ANS-Core v2.0.

Implementers are encouraged—but not required—to adopt similar signaling structures where transparency, debugging, or regulatory scrutiny is desired.

22.Appendix H — Standard Reason Codes (Informative)

H.1 Purpose and Scope

This appendix defines a non-exhaustive set of standard reason codes that MAY be used by Registrars, Resolvers, and related AgentNet components to signal the rationale for lifecycle changes, validation outcomes, resolution decisions, or compliance events.

This appendix is informative and non-normative.
Use of the reason codes defined herein is OPTIONAL and SHALL NOT be interpreted as a condition of conformance with ANS v2.0.

H.2 Motivation

ANS v2.0 establishes strong requirements for auditability, transparency, and deterministic behavior but intentionally avoids prescribing governance procedures or enforcement mechanisms.

Standard reason codes:

- Improve machine-readable audit trails
- Reduce ambiguity in operational signaling
- Promote interoperability across independent implementations
- Support regulatory, legal, and compliance review without imposing policy

Reason codes communicate *why* an action occurred, not *whether* it was justified.

H.3 Design Principles

Standard reason codes:

- Are descriptive, not prescriptive
- Convey explanation, not authority
- Do not alter lifecycle semantics, provenance, or policy inheritance
- Are intended for logging, signaling, and audit, not enforcement

Implementations MAY extend, namespace, or refine reason codes as needed, provided extensions do not conflict with the meanings described here.

H.4 Reason Code Categories

Reason codes are grouped into illustrative categories for clarity.
Category groupings are informational only.

H.4.1 Identity and Lifecycle Management

Used to explain Node issuance, status changes, revocation, or retirement.

Code	Description
NODE_ISSUED	Node identifier was newly issued by an accredited Registrar
NODE_RETIRED	Node was formally retired
NODE_INACTIVATED	Node was temporarily inactivated
NODE_REACTIVATED	Node returned to active status
DUPLICATE_ISSUANCE	Duplicate or conflicting identity issuance detected

---

H.4.2 Authority and Authenticity

Used when authority verification or authenticity checks affect outcomes.

Code	Description
AUTHORITY_CONFIRMED	Publisher and Node authority successfully verified
AUTHORITY_CONFLICT	Conflicting authority claims detected
UNVERIFIABLE_AUTHORITY	Authority could not be verified
KEY_COMPROMISE	Suspected or confirmed compromise of identity credentials

---

H.4.3 Capsule Publication and Versioning

Used to explain Capsule replacement, revocation, or publication state changes.

Code	Description
CAPSULE_PUBLISHED	New Capsule version published
CAPSULE_SUPERSEDED	Capsule replaced by a newer version
CAPSULE_REVOKED	Capsule revoked and no longer authoritative
CAPSULE_EXPIRED	Capsule expired under declared policy or metadata

H.4.4 Resolution and Selection

Used by Resolvers to signal why a particular Capsule was or was not selected.

Code	Description
LATEST_AUTHORITATIVE	Latest non-revoked authoritative Capsule selected
EXPLICIT_VERSION_REQUEST	Specific Capsule version requested by Consumer
POLICY_EXCLUSION_APPLIED	Capsule excluded due to declared policy constraints
NODE_INACTIVE	No Capsule selected because Node was inactive or retired
NO_ELIGIBLE_CAPSULE	No Capsule met resolution criteria

H.4.5 Validation and Conformance

Used to explain validation results or compliance-related actions.

Code	Description
VALIDATION_PASS	All applicable validation checks succeeded
VALIDATION_FAIL	One or more validation checks failed

Code	Description
MISSING_PROVENANCE	Required provenance information was absent
INTEGRITY_VIOLATION	Capsule integrity could not be verified
NON_CONFORMANCE_DETECTED	Persistent non-conformance observed

H.4.6 Abuse and Operational Safeguards

Used to signal abuse mitigation or operational protections.

Code	Description
ABUSE_DETECTED	Abuse or misuse pattern detected
RATE_LIMIT_APPLIED	Resolution or access rate limits enforced
TEMPORARY_SUSPENSION	Temporary suspension pending investigation
OPERATIONAL_MITIGATION	Non-policy operational safeguards applied

---

H.5 Extensibility and Namespacing

Implementations MAY define additional reason codes:

- Using implementation-specific namespaces
- Using Registrar- or Resolver-scoped prefixes
- Using domain-specific extensions

Example:

acme:LEGACY_IDENTITY_MIGRATION

Extensions SHOULD NOT redefine or conflict with the meanings of standard codes defined in this appendix.

H.6 Relationship to Authority and Policy

Reason codes:

- Do NOT create authority
- Do NOT override Publisher assertions
- Do NOT imply enforcement or judgment
- Do NOT substitute for formal governance processes

They exist solely to improve transparency and traceability.

H.7 Relationship to Conformance

Use or non-use of reason codes does not affect conformance with ANS v2.0.

Conformance is determined exclusively by compliance with normative requirements defined in the Core specification.

H.8 Summary

Standard reason codes provide a shared explanatory vocabulary that enhances auditability, interoperability, and ecosystem trust while preserving the federated, neutral, and non-prescriptive design of AgentNet.

Their use is OPTIONAL but RECOMMENDED where clarity, debugging, or external review is required.

23.Appendix I — LD-RAG Output Attribution Pattern (Informative)

I.1 Purpose and Scope

This appendix describes an illustrative attribution pattern for outputs produced using LD-RAG (Linked Data Retrieval Augmented Generation), enabling clear distinction between:

- Authoritative data retrieved from AgentNet Capsules
- Inferred data derived from such data
- Model-generated synthesis produced by an AI system

This appendix is informative and non-normative.
Nothing herein alters the definition of LD-RAG, constrains model behavior, or introduces requirements beyond those defined elsewhere in ANS v2.0.

I.2 Motivation

ANS v2.0 defines LD-RAG as a reasoning approach grounded in authoritative, provenance-bearing data.
However, LD-RAG outputs often combine multiple epistemic layers:

1. Verbatim authoritative facts
2. Inferred or probabilistic conclusions
3. Model-generated narrative or synthesis

Without explicit attribution, these layers may become indistinguishable to downstream systems, auditors, or users.

This appendix provides a transparent attribution pattern to preserve authority boundaries while allowing expressive AI outputs.

I.3 Attribution Principles

An LD-RAG attribution pattern SHOULD adhere to the following principles:

- Source Transparency
  Authoritative claims SHOULD be traceable to specific Capsules.
- Inference Separation
  Inferred data SHOULD be distinguishable from asserted facts.
- Synthesis Disclosure
  Model-generated synthesis SHOULD NOT be represented as authoritative.
- Non-Interference
  Attribution metadata MUST NOT alter or reinterpret Capsule content.

These principles are descriptive and do not impose implementation constraints.

I.4 Conceptual Attribution Layers

An LD-RAG output MAY be conceptually segmented into the following layers:

1. Authoritative Layer
   Facts retrieved directly from Capsules and treated as authoritative.
2. Inferred Layer
   Data derived from authoritative inputs through inference, aggregation, or probabilistic reasoning.
3. Synthesis Layer
   Narrative, summarization, or reasoning generated by the model.

The presence or absence of any layer is implementation-dependent.

I.5 Illustrative Attribution Structure

The following example is illustrative only and does not prescribe structure, field names, or serialization format.

```
{
  "output": {
    "text": "Example Corp was founded in 2014 and operates a global logistics platform."
  },
  "attribution": {
    "authoritative": [
      {
        "capsule": "agentnet://capsule/example-001@1.1.0",
        "facts": ["name", "foundingDate"],
        "publisher": "agentnet://publisher/example-pub"
      }
    ],
    "inferred": [
      {
        "fact": "operates a global logistics platform",
        "confidence": 0.78,
        "sources": [
          "agentnet://capsule/example-001@1.1.0"
        ]
      }
    ],
    "synthesis": {
      "type": "model-generated",
      "notes": "No authoritative operational scope data was available."
    }
  }
}
```

I.6 Authoritative Attribution

Authoritative attribution MAY include:

- Capsule identifiers
- Capsule version identifiers
- Publisher identifiers

- Specific facts or properties used

Authoritative attribution MUST NOT imply endorsement beyond the Publisher’s original assertions.

I.7 Inference Attribution

Inferred attribution MAY include:

- Confidence scores
- Inference rules or methods (informally described)
- Source Capsules used as inputs

Inference attribution improves interpretability but SHALL NOT elevate inferred data to authoritative status.

I.8 Synthesis Disclosure

Model-generated synthesis SHOULD be explicitly identified as such.

Synthesis disclosure MAY include:

- A description of the synthesis role
- Known data gaps or uncertainty
- Model-side assumptions

Synthesis disclosure ensures that generated narrative is not confused with authoritative data.

I.9 Relationship to Provenance and Authority

LD-RAG attribution:

- Does NOT replace Capsule provenance
- Does NOT modify authoritative data
- Does NOT confer authority on inferred or synthesized content

Capsule provenance remains the sole mechanism for authoritative claims within AgentNet.

I.10 Relationship to Conformance

Use or non-use of the attribution pattern described in this appendix does not affect conformance with ANS v2.0.

Conformance is determined exclusively by compliance with normative requirements defined in the Core specification.

I.11 Non-Goals

This appendix does NOT:

- Mandate AI model architectures
- Require confidence scoring
- Define output formats or APIs
- Prescribe UI or presentation layers
- Constrain natural language generation

These concerns are intentionally left to implementers and ecosystem tooling.

I.12 Summary

The LD-RAG Output Attribution Pattern provides a transparent, machine-friendly way to distinguish authority, inference, and synthesis in AI-generated outputs grounded in AgentNet data.

Its adoption is OPTIONAL but RECOMMENDED where clarity, auditability, or regulatory scrutiny is important.
