Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 31 August 2026

L. Zhang
S. Wang
Y. Li
H. Yang
AsiaInfo Technologies (China) Inc.

27 February 2026

Ontology-based Semantic Interaction for Internet of Agents
draft-zhang-dmsc-ioa-semantic-interaction-01

Abstract

This document specifies a normative semantic layer for agent-to-agent interaction in the Internet of Agents (IoA). The semantic layer provides a common ontology model and a JSON-LD serialization profile (RECOMMENDED), while allowing other RDF serializations for expressing capabilities, intents, tasks, and context. The document defines required classes and properties, a negotiation procedure, and alignment rules for heterogeneous ontologies. This layer is designed to be used by interaction and discovery protocols but does not define transport, session, or security protocols. It enables deterministic semantic interoperability across domains.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 31 August 2026.

Copyright Notice

Table of Contents

1. Introduction

The Internet of Agents (IoA) envisions large-scale collaboration among heterogeneous agents across domains, vendors, and execution environments. Current agent ecosystems often fail to interoperate because capability descriptions, task intents, and context semantics are expressed using incompatible vocabularies. This semantic mismatch blocks cross-domain discovery, reliable task decomposition, and predictable coordination.

This document defines an ontology-based semantic layer for agent interaction. The layer provides:

- A minimal ontology model for capabilities, intents, tasks, and context.

- A JSON-LD serialization profile (RECOMMENDED) for exchange over existing interaction protocols, with allowance for other RDF serializations.

- A negotiation and alignment procedure for heterogeneous ontologies.

The semantic layer complements existing interaction and discovery protocols. It does not define transport, session, or security protocols, and it does not mandate a specific discovery system. It provides deterministic semantics that those protocols can carry.

2. Conventions and Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] and [RFC8174].

The following terms are used in this document:

Internet of Agents (IoA): An ecosystem and architecture enabling interoperable collaboration among heterogeneous agents across domains and environments.

Semantic Layer: A logical layer that standardizes meaning representation for agent interaction, independent of transport.

Ontology: A formal vocabulary of concepts and relations that can be used to describe capabilities, intents, tasks, and context.

Semantic Profile: A minimal, structured set of fields used to express interaction semantics.

Capability: A well-defined function or service that an agent can provide, described semantically.

Intent: A goal-oriented statement describing the desired outcome of an interaction.

Task: A unit of work with a lifecycle and status, linked to intent and capability.

Context: Constraints, assumptions, and environmental conditions relevant to a task or interaction.

Alignment: A mapping procedure between ontologies to reconcile equivalent or related concepts.

3. Scope and Non-Goals

This document specifies:

- A normative semantic ontology model.

- A JSON-LD serialization profile (RECOMMENDED) for interaction and discovery semantics, with allowance for other RDF serializations.

- A negotiation procedure for semantic compatibility.

- Alignment artifacts and conflict resolution rules.

This document does not specify:

- Transport, session, or security protocols.

- A specific discovery or registration protocol.

- A full ontology library for a specific industry domain.

4. Semantic Layer Architecture

The semantic layer sits above interaction protocols and below application logic. It provides a structured meaning representation that can be exchanged, validated, and mapped across heterogeneous agents. The semantic layer is protocol-agnostic and can be carried by existing agent-to-agent interaction mechanisms.

The layer enables:

- Capability alignment across different vocabularies.

- Intent clarity for task decomposition and coordination.

- Context sharing for constraints and assumptions.

For cross-domain deployments, a shared semantic knowledge base (e.g., ontology and concept registries with version history) MAY be used as a common reference to improve semantic consistency.

The semantic layer is illustrated in Figure 1.

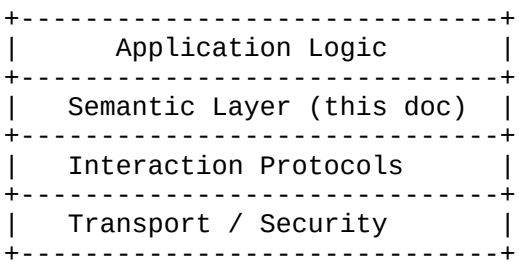

Figure 1

Figure 1: Semantic Layer Position

This layer positioning is consistent with the unified layered architecture shown in Figure 5 of [MACP-01].

5. Ontology Model (Normative)

The ontology model defines the minimal classes and properties that MUST be supported for interoperable interaction. This model is RECOMMENDED to be expressed using RDF/OWL concepts [RDF11-CONCEPTS] [OWL2-OVERVIEW].

RDF provides a graph-based, globally identifiable data model that is well-suited to cross-domain integration, incremental extension, and ontology alignment. It enables explicit semantics, constraint validation, and reasoning using mature tooling, which improves interoperability, verifiability, and long-term reuse.

5.1. Core Classes

The following classes are REQUIRED:

- ioa:Agent An entity that can perform tasks and exchange messages.

- ioa:Capability A function or service an agent can provide.

- ioa:Intent A goal-oriented description of desired outcome.

- ioa:Task A unit of work with status and lifecycle.

- ioa:Context Constraints and environmental conditions.

- ioa:SemanticProfile A container for semantic references used in interaction.

Optional classes MAY include:

- ioa:Skill Reusable expertise or competency that refines a Capability.

- ioa:Resource Data, tools, or assets required to perform a Task.

- ioa:Constraint Policy, rule, or limit that bounds Task execution.

Optional classes provide domain-specific extensions and are not required for baseline interoperability.

5.2. Core Properties

The following properties are REQUIRED:

- ioa:hasCapability (Agent -> Capability) Links an agent to its provided capabilities.

- ioa:hasIntent (Task -> Intent) Associates a task with its intended goal.

- ioa:hasContext (Task -> Context) Binds a task to its contextual constraints.

- ioa:requiresCapability (Task -> Capability) Declares capabilities required to fulfill a task.

- ioa:providedBy (Capability -> Agent) Identifies the agent that provides a capability.

- ioa:hasProfile (Message -> SemanticProfile) Attaches the semantic profile to a message.

- ioa:ontologyId (SemanticProfile -> IRI) Identifier of the ontology used for interpretation.

- ioa:ontologyVersion (SemanticProfile -> literal) Version of the referenced ontology.

The following properties are RECOMMENDED:

- ioa:profileVersion (SemanticProfile -> literal) Version of the semantic profile instance.

- ioa:alignmentRef (SemanticProfile -> IRI) Reference to alignment artifacts used for mapping.

- ioa:profileContext (SemanticProfile -> Context) Contextual constraints used for interpretation.

These properties link core classes into a minimal graph that enables capability discovery, intent interpretation, and semantic alignment.

5.3. Profiles and Versions

A SemanticProfile is an instance-level declaration of the ontology concepts and context used in a specific interaction.

A SemanticProfile SHOULD include:

- ioa:ontologyId Identifier of the ontology used for interpretation.

- ioa:ontologyVersion Version of the referenced ontology.

- ioa:capabilityRefs One or more Capability IRIs referenced by this interaction.

- ioa:intentRef The Intent IRI that declares the interaction goal.

SemanticProfile versioning SHOULD be explicit and SHOULD be comparable using a version string or numeric identifier. Agents SHOULD support backwards compatibility within the same major version.

5.4. Constraints and Consistency Rules

Ontologies used with this specification SHOULD define constraints that enable semantic validation. Constraints MAY be expressed using RDF/OWL axioms or constraint languages such as SHACL.

At minimum, the following consistency rules are RECOMMENDED:

- Capability domain/range constraints: ioa:hasCapability MUST relate an Agent to a Capability.

- Intent applicability: ioa:intentRef MUST reference an Intent that is compatible with the referenced capabilities.

- Context consistency: if profileContext.domain is present, termBindings MUST resolve to concepts within that domain.

- Disallowed sense mapping: a termBinding MAY be rejected if the sense conflicts with the declared domain.

Agents SHOULD validate semantic profiles against these rules before task execution and reject or renegotiate when violations are detected.

5.5. Example Ontology Fragment

The following simplified fragments illustrate a base ontology and two domain extensions. Domain ontologies extend the base by importing or referencing it.

Base ontology (ioa core):

```
@prefix ioa: <https://ex.org/ioa/semantic#> .

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .

ioa:Domain   a rdfs:Class .
ioa:Concept  a rdfs:Class .
ioa:inDomain a rdfs:Property .
ioa:abbr     a rdfs:Property .
ioa:preferredAbbr a rdfs:Property .
ioa:profileContext a rdfs:Property .
```

Figure 2

Computer-science extension:

```
@prefix ioa: <https://ex.org/ioa/semantic#> .
@prefix cs:  <https://ex.org/onto/cs#> .

cs:ComputerScience  a ioa:Domain .
cs:LargeLanguageModel  a ioa:Concept ;
  ioa:inDomain cs:ComputerScience ;
  ioa:abbr "LLM" .

cs:TechTranslation  a ioa:Capability .
cs:TranslateTechDoc a ioa:Intent ;
  ioa:requiresCapability cs:TechTranslation ;
  ioa:appliesToDomain cs:ComputerScience .
```

Figure 3

Legal-domain extension:

```
@prefix ioa: <https://ex.org/ioa/semantic#> .
@prefix law: <https://ex.org/onto/law#> .

law:Law  a ioa:Domain .
law:MasterOfLaws  a ioa:Concept ;
  ioa:inDomain law:Law ;
  ioa:abbr "LLM" ;
  ioa:preferredAbbr "LL.M." .
```

Figure 4

6. JSON-LD Serialization Profile (Normative)

JSON-LD is the RECOMMENDED serialization for semantic payloads [JSON-LD]. Other semantically equivalent representations MAY be used if they preserve the required fields and deterministic interpretation. Other RDF serializations (Turtle, RDF/XML) MAY be supported for offline exchange.

6.1. @context Requirements

A JSON-LD payload MUST include a @context that defines ioa: terms and any domain-specific extensions. The base context for this document is:

https://ex.org/ctx

This base context defines the ioa: prefix as https://ex.org/ioa/semantic# and MAY be extended. Domain extensions MAY be declared inline by adding prefixes that map to ontology IRIs, thereby inheriting the common ioa: semantics.

6.2. Message Semantic Envelope

A semantic envelope is an object that can be embedded in interaction messages. The envelope MUST contain:

- "@context"

- "type": "ioa:SemanticProfile"

- "ontologyId"

- "ontologyVersion"

- "capabilityRefs"

- "intentRef"

Example:

```
{
  "@context": [
    "https://ex.org/ctx",
    { "cs": "https://ex.org/onto/cs#" }
  ],
  "type": "ioa:SemanticProfile",
  "ontologyId": "https://ex.org/onto/agent-core",
  "ontologyVersion": "1.0",
  "capabilityRefs": ["cs:TechTranslation"],
  "intentRef": "cs:TranslateTechDoc",
  "profileContext": {
    "domain": "cs:ComputerScience",
    "termBindings": [
      {
        "term": "LLM",
        "sense": "cs:LargeLanguageModel"
      }
    ]
  }
}
```

Figure 5

6.3. Capability Registration Payload

Capability registration payloads MUST include ioa:Capability objects with identifiers and input/output constraints. Example:

```
{
  "@context": [
    "https://ex.org/ctx",
    { "cs": "https://ex.org/onto/cs#" }
  ],
  "type": "ioa:Capability",
  "id": "cs:TechTranslation",
  "label": "Technical Document Translation",
  "inputs": ["text/plain"],
  "outputs": ["text/plain"]
}
```

Figure 6

7. Semantic Negotiation and Alignment

7.1. Compatibility Determination

Agents MUST determine semantic compatibility prior to executing a task. Compatibility outcomes are:

- Direct match: identical ontologyId and compatible versions.

- Mapped match: alignment artifacts allow translation.

- No match: reject or fallback.

7.2. Alignment Artifacts

Alignment artifacts MUST be represented as mapping tables or graphs. The artifact SHOULD include:

- sourceOntologyId

- targetOntologyId

- mappingRules

- confidence

7.3. Conflict Resolution Rules

When multiple mappings are possible, agents SHOULD apply conflict resolution rules in this order:

1. Local policy priority

2. Higher confidence mapping

3. Explicit user override

7.4. Alignment Algorithm Details

Alignment MAY be performed by combining lexical, structural, and behavioral signals. The following steps are RECOMMENDED:

- Lexical Matching: compare labels, synonyms, and aliases.

- Structural Matching: compare parent/child relationships and property constraints.

- Behavioral Matching: compare expected inputs/outputs of capabilities and historical success.

The alignment process SHOULD yield a mapping score between 0 and 1. Mappings below a local threshold SHOULD be rejected or require explicit user confirmation.

8. Discovery and Capability Semantics

This semantic layer can be used by discovery systems to match capability requirements against registered capabilities. Agents SHOULD advertise capabilities using ioa:Capability objects and include semantic profiles in discovery responses. The discovery protocol itself is out of scope; for example interactions see [IoA-Task].

9. Interaction Semantics

Interaction messages SHOULD carry a semantic envelope. Agents MUST ensure that semantic profiles associated with a task remain stable for the lifetime of the task unless renegotiated. If semantic renegotiation occurs, a new profile MUST be recorded with the task.

An implementation may use natural language or other task content as the primary interaction content, while carrying the semantic envelope alongside the task-specific input so that intent, capability, and context remain explicit to both endpoints. As discussed in [DMSC-NLIP-NOTES], this semantic layer MAY be carried by an NLIP-based interaction. In such cases, the semantic requirements defined by this document still apply, while the concrete message encoding and transport realization remain outside the scope of this document.

Semantic compatibility checks, profile updates, and alignment renegotiation SHOULD be performed using a control exchange separate from ordinary task execution messages. A control exchange for semantic renegotiation SHOULD carry the proposed ontology identifier, version, and any candidate alignment artifacts, and a successful renegotiation MUST result in a new semantic profile being recorded for subsequent task messages.

10. Mapping to ACPs/ADP/AIP

This semantic layer is intended to complement the Agent Collaboration Protocols (ACPs) architecture [ACPs-ARC] and can be mapped as follows:

- ADP (Agent Discovery Protocol): use SemanticProfile to represent capability requirements and results for semantic matching.

- AIP (Agent Interaction Protocol): attach SemanticProfile to interaction messages to make task intent and context explicit.

- Identity/authentication components: no changes required; semantic profiles are orthogonal to identity and authentication.

- NLIP-based realization: carry the semantic envelope as a structured payload and use control exchanges for semantic negotiation and profile renegotiation.

The NLIP-based mappings in this section and in Section 11.5 are illustrative deployment examples only. This document defines the semantic layer and its payload requirements, but does not normatively depend on NLIP or any other transport, session, or security mechanism for conformance.

This mapping allows ACPs deployments to adopt semantic interoperability without modifying transport or security components.

This semantic layer also complements the DMSC infrastructure architecture draft [INF-ARCH] by providing semantic descriptions for capability directories (Section 6.1), discovery matching (Section 6.2), and semantic routing inputs (Section 6.3). An illustrative HTTP encapsulation example for the semantic envelope in an NLIP-style realization is shown in Section 11.5.

11. Example Flow and Examples

11.1. Cross-Domain Discovery Example

Agent A (domain X) requests a capability "radiology-report-summary". Agent B (domain Y) advertises capability "clinical-note-summarize". The domains use different ontologies. Alignment produces a mapping with confidence 0.91, enabling cross-domain discovery.

Example discovery response (simplified):

```
{
  "@context": [
    "https://ex.org/ctx",
    { "clinical": "https://y.org/onto/clinical#" }
  ],
  "type": "ioa:SemanticProfile",
  "ontologyId": "https://y.org/onto/clinical",
  "ontologyVersion": "2.1",
  "capabilityRefs": ["clinical:note-summarize"],
  "intentRef": "clinical:summary-intent",
  "alignmentRef": "https://ex.org/align/X-Y-2026"
}
```

Figure 7

11.2. Interaction Example

Agent A sends a task request with a semantic profile that includes a domain context and term binding. This helps disambiguate abbreviations, e.g., "LLM" is bound to "LargeLanguageModel" in computer science, not "MasterOfLaws" in legal education.

Example task request (simplified):

```
{
  "header": {
    "task_id": "t-1038",
    "message_type": "task_request"
  },
  "semantic": {
    "@context": [
      "https://ex.org/ctx",
      { "cs": "https://ex.org/onto/cs#" }
    ],
    "type": "ioa:SemanticProfile",
    "ontologyId": "https://ex.org/onto/agent-core",
    "ontologyVersion": "1.0",
    "capabilityRefs": ["cs:TechTranslation"],
    "intentRef": "cs:TranslateTechDoc",
     "profileContext": {
      "domain": "cs:ComputerScience",
      "sourceLang": "zh",
      "targetLang": "en",
      "termBindings": [
        {
          "term": "LLM",
          "sense": "cs:LargeLanguageModel"
        }
      ]
    }
  },
  "payload": {
    "input": "..."
  }
}
```

Figure 8

11.3. Consistency Validation Example

In this example, the profile declares a computer-science domain but binds "LLM" to a legal-education sense. A validator SHOULD reject or request correction based on the consistency rules.

Example validation failure (simplified):

```
{
  "semantic": {
    "@context": [
      "https://ex.org/ctx",
      { "cs": "https://ex.org/onto/cs#" },
      { "law": "https://ex.org/onto/law#" }
    ],
    "type": "ioa:SemanticProfile",
    "ontologyId": "https://ex.org/onto/agent-core",
    "ontologyVersion": "1.0",
    "intentRef": "cs:TranslateTechDoc",
    "profileContext": {
      "domain": "cs:ComputerScience",
      "termBindings": [
        {
          "term": "LLM",
          "sense": "law:MasterOfLaws"
        }
      ]
    }
  },
  "validation": {
    "status": "reject",
    "reason": "termBinding sense conflicts with domain"
  }
}
```

Figure 9

11.4. Alignment Artifact Example

```
{
  "sourceOntologyId": "https://x.org/onto/rad",
  "targetOntologyId": "https://y.org/onto/clinical",
  "mappingRules": [
    {
      "from": "rad:report-summary",
      "to": "clinical:note-summarize"
    }
  ],
  "confidence": 0.91
}
```

Figure 10

11.5. Illustrative NLIP HTTP Encapsulation Example

This example is non-normative and shows how the semantic envelope can be carried in an NLIP-style HTTP exchange without changing the semantic requirements in Section 6.2.

```
POST /nlip HTTP/1.1
Host: ex.org
Content-Type: application/json

{
  "MessageType": "Request",
  "Format": "text",
  "Subformat": "en",
  "Content": "Translate the attached note into English.",
  "Submessages": [
    {
      "Label": "src-doc",
      "Format": "binary",
      "Subformat": "application/pdf",
      "Content": "<pdf>"
    },
    {
      "Label": "sem-profile",
      "Format": "structured",
      "Subformat": "json",
      "Content": {
         "@context": [
          "https://ex.org/c",
          { "cs": "https://ex.org/cs#" }
        ],
          "type": "ioa:SemanticProfile",
        "ontologyId": "https://ex.org/core",
        "ontologyVersion": "1.0",
        "capabilityRefs": ["cs:TechTrans"],
        "intentRef": "cs:TranslateDoc",
        "alignmentRef": "https://ex.org/a1"
      }
    }
  ]
}
```

Figure 11

12. Compliance Requirements

An implementation compliant with this specification MUST:

- Support the core classes and properties in Section 5.

- Produce and parse semantic envelopes as specified in Section 6; JSON-LD support is RECOMMENDED.

- Perform semantic negotiation prior to task execution (Section 7).

An implementation MAY support extended domain ontologies.

13. Security Considerations

Semantic profiles may reveal sensitive capability and context information. Implementations SHOULD consider:

- Minimizing semantic disclosure (least-information profiles).

- Integrity protection of semantic profiles and alignment artifacts.

- Access control on ontology registries and mapping services.

- Policy-based filtering of sensitive capabilities in discovery.

If a shared semantic knowledge base is used, implementers SHOULD consider provenance, version control, and trust policies to prevent poisoning or unauthorized changes.

Alignment artifacts can be a target for poisoning or downgrade attacks. Implementations SHOULD validate alignment sources and require provenance for mappings used in critical tasks.

14. IANA Considerations

This document makes no request for IANA action.

15. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, 2017, <https://www.rfc-editor.org/info/rfc8174>.

16. Informative References

[ACPs-ARC] Liu, J., Yu, K., Li, K., and K. Chen, "Agent Collaboration Protocols Architecture for Internet of Agents", Work in Progress, Internet-Draft, draft-liu-dmsc-acps-arc-03, 2026, <https://datatracker.ietf.org/doc/draft-liu-dmsc-acps-arc/03/>.

[DMSC-NLIP-NOTES] Verma, D., Bertino, E., Ratnaparkhi, A., and S. Hughes, "Use of Natural Language for Agent Communication", Work in Progress, Internet-Draft, draft-verma-dmsc-nlip-notes-00, 2026, <https://datatracker.ietf.org/doc/draft-verma-dmsc-nlip-notes/00/>.

[INF-ARCH] Li, X. and A. Wang, "Dynamic Multi-agent Secured Collaboration Infrastructure Architecture", Work in Progress, Internet-Draft, draft-li-dmsc-inf-architecture-03, 2026, <https://datatracker.ietf.org/doc/draft-li-dmsc-inf-architecture/03/>.

[IoA-Task] Yang, C., "Internet of Agents Task Protocol", Work in Progress, Internet-Draft, draft-yang-dmsc-ioa-task-protocol-02, 2026, <https://datatracker.ietf.org/doc/draft-yang-dmsc-ioa-task-protocol/02/>.

[JSON-LD] W3C, "JSON-LD 1.1", 2020, <https://www.w3.org/TR/json-ld11/>.

[MACP-01] Li, X., Liu, J., Du, C., and L. Zhang, "Multi-Agent Collaboration Protocol for Internet of Agents", Work in Progress, Internet-Draft, draft-li-dmsc-macp-01, 2026, <https://datatracker.ietf.org/doc/draft-li-dmsc-macp/01/>.

[OWL2-OVERVIEW] W3C, "OWL 2 Web Ontology Language Overview", 2012, <https://www.w3.org/TR/owl2-overview/>.

[RDF11-CONCEPTS] W3C, "RDF 1.1 Concepts and Abstract Syntax", 2014, <https://www.w3.org/TR/rdf11-concepts/>.

Authors' Addresses

Lianhua Zhang
AsiaInfo Technologies (China) Inc.
Beijing
100000
China
Email: zhanglh2@asiainfo.com

Shoufeng Wang
AsiaInfo Technologies (China) Inc.
Beijing
100000
China
Email: wangsf11@asiainfo.com

Yun Li
AsiaInfo Technologies (China) Inc.
Beijing
100000
China
Email: liyun9@asiainfo.com

Huiling Yang
AsiaInfo Technologies (China) Inc.
Beijing
100000
China
Email: yanghl10@asiainfo.com