Network Working Group
Internet-Draft
Intended status: Informational
Expires: 24 May 2026

G. Kartha

20 November 2025

Internet 2.0: An Intent-Aware, AI-Native Extension of the Web
draft-kartha-internet20-ainative-00

Abstract

This document proposes Internet 2.0, an AI-native extension of the traditional web architecture. Unlike the current internet—which is designed primarily for document retrieval—Internet 2.0 enables distributed model discovery, intent-based routing, and protocol-level AI interaction. Core components include HTTP+AI, an AI-aware extension to HTTP; the Model Resolution Network (MRN), an AI-native analogue to DNS; and the AI-Aware Browser, a new class of client optimized for intelligent interaction rather than document traversal. This architecture treats AI models as first-class network entities and provides a foundation for a decentralized, semantic, and privacy-preserving AI layer on the internet.

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 24 May 2026.

Copyright Notice

Table of Contents

1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Status of This Memo

This document is not an Internet Standards Track specification; it is published for examination, experimental implementation, and evaluation.

This document defines an Experimental Protocol for the Internet community. This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfcXXXX.

3. Copyright Notice

4. Introduction

The web evolved from static hypertext into dynamic, socially interactive platforms, yet its foundations continue to center around documents, URLs, and link traversal. Modern AI usage has shifted user expectations away from link retrieval toward synthesized answers and goal-oriented assistance. However, AI systems remain isolated behind proprietary APIs without integration into the underlying architecture of the internet.

Internet 2.0 proposes a protocol-level expansion where AI models are discoverable, routable, and composable across the network. This document defines three components supporting this architecture: (1) HTTP+AI as an AI-aware request protocol; (2) the Model Resolution Network as a semantic analog to DNS; and (3) the AI-Aware Browser as the primary user interface. The objective is not to replace the existing internet, but to extend it with a distributed, intelligent mesh capable of resolving intents, negotiating capabilities, and enabling privacy-preserving inference.

5. Terminology

Intent: A structured representation of a user's goal expressed in natural language.

Intent Vector: The numerical embedding derived from natural language input.

Model Resolution Network (MRN): A hierarchical discovery network resolving intents to model endpoints.

Model Record (MRec): A signed metadata object describing model identity, capabilities, trust score, and endpoint (formally defined in Appendix A).

HTTP+AI: An extension of HTTP supporting AI-specific negotiation fields.

AI-Aware Browser: A client capable of rendering synthesized answers and invoking distributed models.

6. Background

The traditional internet was designed for navigating and retrieving documents, not for understanding or synthesizing information. Users rely on keyword-based search engines that return lists of URLs, requiring manual extraction of relevant knowledge. This document-centric model has become increasingly insufficient as AI systems take on roles in programming assistance, legal summarization, medical triage, and enterprise knowledge retrieval. [Internet20-Paper]

Large language models have shifted user expectations from link retrieval to synthesized, goal-oriented answers. However, these models operate behind proprietary APIs, lack standard discovery mechanisms, and remain centralized and cloud-bound. Traditional browsers are unaware of model semantics, capabilities, or privacy requirements. These architectural gaps motivate the design of Internet 2.0. [Internet20-Paper]

7. Problem Statement

The traditional internet was not designed for AI-native interaction. Limitations include document-centric retrieval, lack of intent-based routing, no AI model discovery mechanism, centralized cloud APIs, and absence of standard metadata describing model capabilities. These limitations hinder decentralization, privacy, and modular AI deployment.

Internet 2.0 aims to support semantic routing, model negotiation, and privacy-aware edge-based inference.

8. Benefits

Internet 2.0 enables multiple benefits across domains, including developer tools, enterprise knowledge search, IoT edge inference, education and research assistance, and regulated workloads in healthcare, law, and compliance. These capabilities arise from the discoverability and composability of models in the MRN, along with privacy-aware execution pathways. [Internet20-Paper]

9. Architectural Overview

Internet 2.0 introduces an AI-native mesh layer composed of HTTP+AI, MRN, MRecs, and AI-aware browsers. The architecture enables a workflow where user intent is embedded, resolved through MRN, and executed through selected models.

```
User -> AI Browser -> MRN -> Model Selection ->
  HTTP+AI -> Model Execution -> Response
```

The system supports privacy-aware routing, latency constraints, capability filtering, caching, and federated model registries. The detailed workflow is illustrated in Appendix B.

10. HTTP+AI Protocol

HTTP+AI is an extension of HTTP enabling semantic intent-based communication between clients and AI model endpoints. This section defines the URI scheme, ABNF syntax, header field definitions, request/response rules, and error codes.

10.1. AI URI Scheme

The ai URI scheme is used for intent-based requests. It is syntactically compatible with generic URIs but introduces AI-specific query parameters. The scheme is defined using Augmented BackusNaur Form (ABNF) as specified in [RFC5234].

```
ai-URI              = "ai:" "//" authority ai-path [ "?" ai-query ]

authority           = host [ ":" port ]
ai-path             = path-abempty
ai-query            = intent-param *( "&" ai-param )

intent-param        = "intent=" qchar-1plus

ai-param            = capability-param
                    / privacy-param
                    / latency-param
                    / ai-token-param

capability-param    = "capability=" qchar-1plus
privacy-param       = "privacy=" qchar-1plus
latency-param       = "latency-target=" 1*DIGIT
ai-token-param      = "token=" qchar-1plus

qchar-1plus         = 1*qchar
qchar               = unreserved / pct-encoded / sub-delims / ":" / "@"
```

Example:

```
ai://models.example.com/query?
  intent=optimize+python+script&
  capability=code-optimization&
  privacy=local-preferred&
  latency-target=100
```

10.2. New HTTP Header Fields

The following header fields are defined for use with HTTP+AI and requested for registration in the "Permanent Message Header Field Registry" (Section 16.2).

10.2.1. AI-Intent Header Field (Request)

The AI-Intent header conveys the full, un-encoded natural-language description of the user’s goal. It SHOULD be used by the MRN for generating the precise Intent Vector.

AI-Intent = "AI-Intent:" OWS quoted-string

Example:

AI-Intent: "optimize this python script for speed"

10.2.2. AI-Capability Header Field (Request)

The AI-Capability header indicates required model skills needed for routing and negotiation (e.g., code-generation, medical-dermatology).

AI-Capability = "AI-Capability:" OWS token *(OWS "," OWS token)

Example:

AI-Capability: code-optimization, python

10.2.3. AI-Privacy Header Field (Request)

The AI-Privacy header indicates privacy constraints for routing and model selection. The value SHALL be a token registered in the IANA "AI-Privacy Constraint Tokens" registry (Section 16.4).

AI-Privacy = "AI-Privacy:" OWS token

Example:

AI-Privacy: regulated

10.2.4. AI-Latency-Target Header Field (Request)

The AI-Latency-Target header conveys the client’s latency expectation in milliseconds. The MRN SHOULD use this value to filter MRecs based on performance metrics. The value represents the time in milliseconds.

```
AI-Latency-Target = "AI-Latency-Target:" OWS 1*DIGIT
```

Example:

```
AI-Latency-Target: 120
```

10.2.5. AI-Model-ID Header Field (Response)

The AI-Model-ID header is returned in the response and contains the ‘ModelID‘ of the specific MRec that successfully executed the inference.

```
AI-Model-ID = "AI-Model-ID:" OWS quoted-string
```

10.2.6. AI-Confidence Header Field (Response)

The AI-Confidence header is returned in the response and contains a float (0.0 to 1.0) indicating the model’s self-assessed confidence in the generated result.

```
AI-Confidence = "AI-Confidence:" OWS 1*DIGIT
                [ "." 1*DIGIT ]  ; float 0.01.0
```

10.3. Request Semantics

A valid HTTP+AI request MUST contain at least:

* The AI-Intent header.
* Either an ai URI or an HTTP(s) URI resolved by the MRN.
* HTTP method GET or POST.

Example HTTP+AI request:

```
GET ai://query?intent=optimize+python+script HTTP/1.1
Host: models.example.com
AI-Intent: "optimize python script"
AI-Capability: code-optimization, python
AI-Privacy: local-preferred
AI-Latency-Target: 100
```

10.4. Response Semantics

The response to an HTTP+AI inference request SHOULD include the AI-Model-ID and AI-Confidence headers (defined above) to aid the AI-Aware Browser in interpreting the result and verifying provenance. The response body SHOULD contain the structured inference output (e.g., JSON).

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json
AI-Model-ID: org.example.model.v1
AI-Confidence: 0.91

{
  "result": "Here is the optimized Python script...",
  "explanation": "Loop unrolling improves speed.",
  "metrics": { "latency_ms": 85 }
}
```

10.5. Error Codes

HTTP+AI defines the following additional status codes for registration in the HTTP Status Code registry (Section 16.3):

* 450 AI-Model-Ambiguous — The MRN resolved the intent to multiple models with similar confidence, and the client MUST re-query with more specific constraints.

* 451 AI-Low-Confidence — The selected model executed the inference but returned a confidence score below a client threshold. The client SHOULD warn the user.

* 453 AI-Privacy-Constraint-Failed — The MRN could not locate any model matching the required AI-Privacy constraint.

11. Model Resolution Network (MRN)

The Model Resolution Network (MRN) is a hierarchical, federated discovery system that resolves semantic intents to appropriate AI model endpoints. It serves as the AI-native counterpart to the Domain Name System (DNS).

11.1. MRN Architecture

MRN consists of Index Models that perform semantic search over registered Model Records (MRecs). The architecture is federated, supporting root-level index models, domain-specific registries, and peer-to-peer lookup nodes.

11.2. Resolution Workflow

The MRN resolution process involves:

1. User query parsing and intent vector embedding
2. Query forwarding to appropriate Index Model
3. Semantic search over MRec database using vector similarity
4. Ranking results by relevance, trust score, and performance
5. Returning one or more matching MRecs (defined in Appendix A) to the client

The Intent Vector SHOULD conform to a consistent, standardized encoding format (e.g., Base64-encoded array of floating-point numbers) to ensure interoperability between AI-Aware Browsers and MRN Index Models.

11.3. Vector-Based Routing

Unlike DNS’s exact string matching, MRN uses semantic embedding similarity for routing. Intent vectors are compared against capability embeddings in MRecs using cosine similarity or other distance metrics.

12. AI-Aware Browser

The AI-Aware Browser is a specialized client interface designed for intent-based interaction with the AI-native internet layer.

12.1. Core Capabilities

* Accepts natural language or goal-oriented prompts
* Generates intent vectors from user queries
* Queries MRN for model discovery and selection
* Executes HTTP+AI requests to distributed models

* Presents synthesized answers with provenance and confidence

* Manages conversational context and follow-up queries

13. Use Cases

13.1. Software Development

Query: "Generate a Dockerfile for a Python FastAPI app" Workflow: MRN resolves to DevOps model → Returns ready-to-use Dockerfile with explanation and confidence score.

13.2. Medical Symptom Analysis

Query: "I have red rashes on my body, possible causes?" Workflow: MRN routes to certified dermatology model → Returns potential diagnoses with confidence levels and medical disclaimers.

14. Security Considerations

14.1. MRec Authentication

Model Records MUST be cryptographically signed and verified to prevent spoofing. The public key infrastructure for MRec signing SHOULD be auditable.

14.2. Intent Privacy

User intents may contain sensitive information; TLS encryption and privacy-preserving embedding techniques are RECOMMENDED.

14.3. Model Poisoning

MRN implementations SHOULD implement trust scoring and reputation systems to mitigate malicious model registration. Provenance data in the HTTP+AI response SHOULD link the answer to the verified MRec.

15. Privacy Considerations

The architecture supports local-device inference, region-aware routing, protection of sensitive intents, and differential privacy for shared models. The AI-Privacy header is the primary mechanism for client-mandated privacy control.

16. IANA Considerations

This document requests several registrations with IANA:

16.1. New URI Scheme

Registration of the ai URI scheme in the "Uniform Resource Identifier (URI) Schemes" registry is requested.

16.2. New HTTP Header Fields

Registration of the following HTTP header fields in the "Permanent Message Header Field Registry" is requested:

* AI-Intent
* AI-Capability
* AI-Privacy
* AI-Latency-Target
* AI-Model-ID
* AI-Confidence

16.3. New Status Codes

Registration of status codes 450, 451, and 453 in the "Hypertext Transfer Protocol (HTTP) Status Code Registry" is requested.

16.4. AI-Privacy Constraint Tokens Registry

This document requests the creation of a new IANA registry named "AI-Privacy Constraint Tokens" to manage the field values for the ‘AI-Privacy‘ HTTP header. The initial tokens SHALL be: ‘local-preferred‘, ‘cloud-allowed‘, and ‘regulated‘.

17. Discussion and Future Work

Key research challenges include authentication and safety of model endpoints, interoperable metadata formats, efficient caching of inference results, versioning and provenance tracking, and economic models for federated participation. [Internet20-Paper]

Future work includes prototyping MRN nodes using vector databases, defining HTTP+AI extensions, creating a lightweight AI-aware browser, and developing open specifications under an AI-RFC series. [Internet20-Paper]

18. Conclusion

Internet 2.0 proposes a protocol-level extension to treat AI models as first-class, discoverable network entities. By integrating intent resolution, model discovery, and AI-native client interfaces, the architecture shifts the web from static document retrieval to dynamic, intelligent interaction. It does not replace the existing internet but extends it with modularity, privacy, and semantic routing. [Internet20-Paper]

19. Appendix A: Model Record (MRec) Specification

The Model Record (MRec) is a signed JSON object that defines an AI model's capabilities, performance characteristics, and deployment information. It is the core record type for the MRN.

```
{
  "ModelID": "org.example.model.v1",
  "Endpoint": "https://models.example.com/infer",
  "Capabilities": ["python", "fastapi", "asyncio"],
  "Version": "1.2.0",
  "Privacy": "edge-local",
  "TrustScore": 0.92,
  "TTL": 3600,
  "LatencyMetrics": { "avg_ms": 85 },
  "Authentication": "<cryptographic-signature>"
}
```

19.1. MRec Required Fields

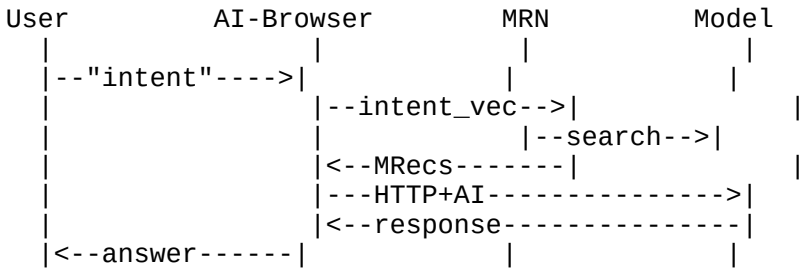
The following fields are REQUIRED in a valid MRec:

* ModelID: Globally unique hierarchical identifier (string).
* Endpoint: Network address for HTTP+AI inference requests (URI).
* Capabilities: Array of domain/task keywords (string array).
* Version: Model version string (string).
* TTL: Time-to-live in seconds for caching (integer).
* Privacy: Token defined in the "AI-Privacy Constraint Tokens" IANA registry (string).
* TrustScore: Metric (0.0 to 1.0) indicating verified reliability (float).

* Authentication: Cryptographic signature over the MRec object, using an IETF-specified algorithm (e.g., ECDSA over JWS).

20. Appendix B: Intent-to-Model Workflow Diagram

The following sequence diagram illustrates the workflow for intent resolution within MRN.



21. References

[RFC2119] IETF, "Key words for use in RFCs to Indicate Requirement Levels", March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC8174] IETF, "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", January 2008, <https://www.rfc-editor.org/rfc/rfc5234>.

[Internet20-Paper] Gokul Kartha, "Internet 2.0: An Intent-Aware, AI-Native Extension of the Web", 2025, <https://engrxiv.org/preprint/view/5800/9663>.

Author’s Address

Gokul Kartha
Email: kartha.gokul@gmail.com
URI: https://www.techysaint.com/.well-known/ietf-draft