NMRG
Internet-Draft
Intended status: Informational
Expires: 3 September 2026

M. A. Jadoon
S. Robitzsch
InterDigital Europe Ltd
C. J. Bernardos
Universidad Carlos III de Madrid

2 March 2026

Agentic AI Architectural Principles for Autonomous Computer Networks
draft-jadoon-nmrg-agentic-ai-autonomous-networks-00

Abstract

Agentic AI systems combine planning, reasoning, tool invocation, and feedback loops to pursue system-defined goals with a controlled degree of autonomy. In networking, this enables an evolution from statically configured automation toward goal-driven closed-loop operations spanning multiple protocol layers and administrative domains.

This document introduces architectural principles for "agentic augmentation" of the existing layered protocol stack as represented by the Internet protocol suite (IP suite). The key concept of the proposed principles is that deterministic protocol layering remains intact for interoperability, while AI Agents are introduced as first-class entities at each IP suite layer and are coordinated by one or more agent controllers via agentic methods and procedures.

The purpose of this document is to initiate discussion within the research community on agentic networking. It identifies architectural research challenges that should be discussed to enable the addition of one or more AI Agents at one or more IP suite layers with the goal to allow AI Agents to improve the behaviour of a layer through reasoning with AI Agents at the same or other IP suite layers.

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/.

Copyright Notice

Table of Contents

1. Introduction

Computer networks have long pursued higher automation for self-configuration, self-healing, self-optimization, and self-protection. Recent advances in AI, including agentic AI systems, enable a shift from isolated automation approaches toward distributed, goal-driven closed-loops that can plan multi-step actions, invoke tools, adapt to feedback, and coordinate with peers to improve the behaviour of existing or new protocols.

However, multi-agent deployments are typically proprietary: there is no widely adopted, vendor-independent approach for naming, reaching, discovering, and coordinating AI Agents that act on network state across multiple protocol layers and domains.

This document is intended to seed IETF discussion by proposing a architectural principles and an interface model for "agentic augmentation" of the existing layered OSI stack. The objective is not to replace the deterministic behavior of protocols used for interoperability and packet transport. Instead, the architecture introduces AI Agents at each layer, coordinated by orchestrators, so that closed-loop automation can be pursued in a structured and governable way.

2. Terminology and Conventions

AI Agent (AIA): A software entity capable of pursuing goals by reasoning and planning, and by invoking tools (APIs, protocols, controllers, knowledge bases) to observe state and perform actions.

Agentic AI System: A system composed of one or more AI Agents and optional controller(s) that performs goal-directed operations over time by using available tools and feedback loops.

Agent Controller: A logical function that accepts an input request/ goal, performs task decomposition and assignment, coordinates one or more AI Agents, and produces outputs or verified actuation outcomes. Controllers may be per-layer (distributed) or centralized.

Agent Net: A set of AI Agents associated with a specific layer of the stack, together with their coordination logic and tools for that layer.

IP Suite Stack Interfaces: The conventional (non-agentic) interfaces between adjacent layers (e.g., a transport interface exposed to applications), and the conventional control/management interfaces that configure or observe a layer (e.g., socket APIs, YANG/NETCONF/ RESTCONF, routing protocol configuration, telemetry streams).

Tool: Any callable capability used by an AI Agent to ground reasoning or execute actions (e.g., retrieve telemetry, push configuration, initiate diagnostics, query a digital twin, call a controller).

3. Baseline: IP Suite Layered Protocol Stack

Layering is central to interoperability. In a conventional layered stack, upper layers invoke services from lower layers using static, well-defined interfaces. In practice, deployed systems follow the Internet protocol suite as shown in Figure 1, commonly represented by Application, Transport, Internet, and Link/Access and physical layers. Nevertheless, the layered abstraction remains a useful conceptual baseline for discussing where automation and closed-loop decision functions are inserted.

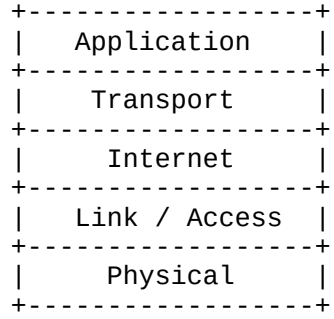

Figure 1: IP Suite Layered Stack

4. Proposed Agentic Augmentation at to Layered Approach

The proposed approach is hybrid: the deterministic layered protocol stack remains the baseline for interoperability and packet transport, while agentic functions are introduced alongside each layer.

In the proposed architecture, each layer contains:

* An Agent Net that can interpret intents relevant to that layer, gather context (telemetry/state), reason, propose actions, and execute actions through the layer’s tools. Agent Net may have one or more AIAs for different purposes within in the layer. For example, a layer may have different agents for congestion control, QoS adaptation, policy and compliance and intent-parsing.

* An agent controller or simply controller manages task decomposition, assignment, guardrails and governance. A controller may or may not be an AIA.

The proposed model targets a "structured autonomy" property: autonomous actions are possible, but are constrained through explicit reference points, policies, authentication/authorization, and auditable workflows.

4.1. 5.1. Per-Layer Agent Nets and Per-Layer Controllers

In this option, each layer has its own Agent Net and its own controller. This aligns with operational decomposition (e.g., separate ownership or tooling per layer), and supports scaling and fault isolation.

Figure 2a illustrates the concept.

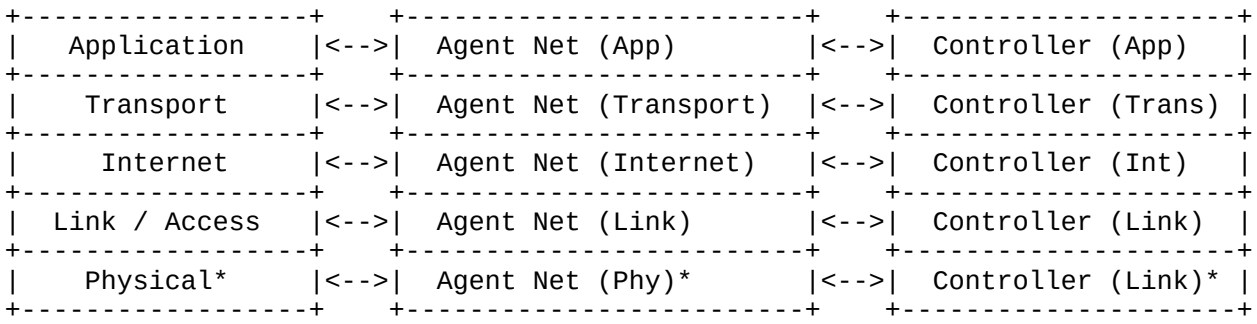

Figure 2a: Proposed Stack with Per-Layer Agent Nets and Per-Layer Controllers

The Physical layer is shown for completeness and includes data-link and access technologies. Some aspects of access technologies may be outside IETF scope and would typically be accessed as external tools/ controllers rather than specified by IETF protocols.

4.2. 5.3. Relationship to the Deterministic Stack

The hybrid architecture is intended to preserve deterministic interoperability of the underlying stack:

* Deterministic inter-layer service interfaces (e.g., a transport service offered to applications) remain unchanged.

* AI Agents do not replace protocols; they invoke tools that already exist (or will be defined) to observe and actuate layer behavior.

* The architecture explicitly separates (a) packet transport and protocol interoperability, from (b) goal-driven automation logic.

This separation is central to making agentic automation governable, testable, and incrementally deployable.

5. Functional Workflow and Reference Points

5.1. 6.1. Controller and Agent Functional Blocks

A generic agentic workflow can be represented by the following functional blocks:

Controller:

* I/O Engine: handles intent/goal/request intake and output formatting.

* Discovery: finds relevant agents, tools and capabilities.

* Task Assignments: decomposes tasks and assigns them to agents (planning).

AI Agent:

* Reasoning: interprets tasks, plans steps, and maintains context.

* Synthesis: composes intermediate results into coherent outputs.

* Execution: invokes tools and performs actions.

* Reinforcement: learns/adapts from feedback.

* Conflict Resolution: resolves competing proposals/actions among agents or across layers.

Figure 3 illustrates this functional view and explicitly labels reference points that are candidates for interoperable definition.

5.2. 6.2. Reference Points (Interfaces)

The following reference points are identified:

C_I (Controller Input): Input request/goal interface into the controller.

C_O (Controller Output): Output response (including action confirmation, explanation, or result summary) from the controller.

C_E (Controller Execution Feedback): Interface for execution-related status and feedback flow between controller and agent reasoning context (e.g., progress, intermediate outcomes, updated constraints).

AIA_E (Agent Execution / Task Assignment): Task assignment and execution coordination between controller and AI Agent execution function.

AIA_R (Inter-AI-Agent Reasoning/Context Exchange): Interface for AI Agents to exchange context, negotiate, and coordinate during reasoning and planning.

AIA_CR (Inter-AI-Agent Conflict Resolution): Interface supporting detection, signaling, and resolution of conflicts among AI Agents (including conflicts spanning layers).

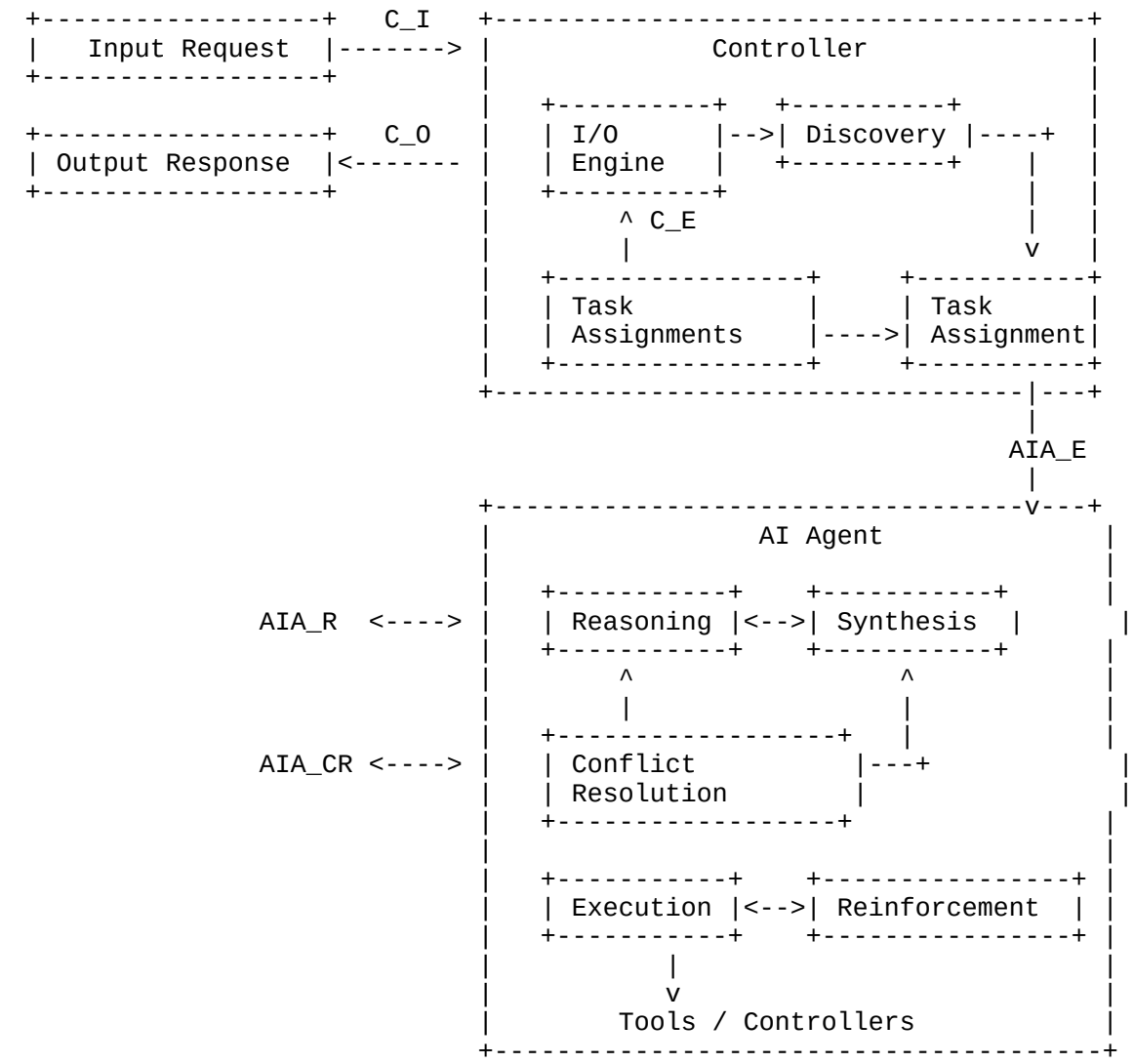

Figure 3: Functional Workflow of Agentic AI Systems with Reference Points (Conceptual)

The intent is not to prescribe a single protocol, but to establish common reference points so that multiple protocol proposals (e.g., agent-to-agent, agent discovery, tool invocation) can be evaluated against a consistent architectural model.

In this document, the term "CX interface" refers to the controller external interfaces (primarily C_I and C_O), and the term "AIAX interfaces" refers to the agent-related interfaces (C_E, AIA_E, AIA_R, AIA_CR) shown above.

6. Challenges Motivating Interoperable Work

This section summarizes key challenges observed in multi-layer, multi-agent deployments that motivate IETF discussion:

* *Lack of a common way for AIAs to identify and reach each other*: Agents operate at different technical layers and environments. There is no consistent way to name, locate, and address agents across networks and platforms. This may results in fragile interoperability, high integration cost, and limited scaling in multi-vendor environments.

* *Limited service discoverability and failure signaling for AIAs*: Agents need to discover other agents, determine if a service exists, and understand why communication failed. This may result in silent failures, unbounded retries, incorrect decisions from incomplete information, and reduced trust in automation.

* *Insufficient data provisioning for training and adaptation*: Agents need data not only for initial training but also during operation or for updates. Agents need consistent approaches for accessing additional data and exchanging learning-relevant information in consistent formats.

* *Distributed security token and access control management*: Agents require credentials to prove identity, access resources, and act on behalf of users/systems. Token creation, scoping, renewal, and revocation are not unified across agent types and layers, complicating audit and compliance.

These challenges are compounded by multi-domain operation, different trust boundaries, and heterogeneous tooling.

7. Security, Safety, Determinism, and Accountability Considerations

Agentic AI systems introduce architectural risks beyond those associated with traditional automation or centralized controllers.

* *Sustainability*:
  Continuous inference, multi-agent coordination, and expanded telemetry exchange may increase energy consumption and management-plane load. Architectural mechanisms prefer bounding control-plane amplification and computational overhead.

* *Security*:
  Agentic systems expand the attack surface through:

  - Tool invocation chains that can enable privilege escalation.

  - Cross-agent coordination interfaces that can propagate compromise.

  - Persistent memory and context that can be subject to poisoning.

  - Model-driven reasoning that can be influenced by malicious or malformed inputs.

  Compromise of a single agent can have cascading impact across layers if coordination boundaries are not explicitly constrained.

* *Deterministic Guardrails*:
  Networking environments require bounded and predictable behavior. Agentic augmentation is intended to preserve deterministic protocol invariants and stability properties of the underlying stack. Guardrail mechanisms can include:

  - Policy-constrained action spaces.

  - Pre-execution validation against safety envelopes.

  - Transactional rollback and state checkpointing.

  - Rate-limited reconfiguration to prevent oscillation.

  - Deterministic conflict resolution hierarchies.

  - Human override and escalation triggers.

  These mechanisms ensure that autonomy remains structured and does not undermine protocol correctness or service guarantees.

* *Accountability*:
  Autonomous decisions are expected to be traceable to inputs, policies, and authorization context. Auditable execution logs and explainability hooks are required to support compliance, debugging, and operational trust.

This document does not define specific mechanisms but asserts that structured autonomy and deterministic safety envelopes are foundational architectural requirements for agentic networking.

8. Relationship to Existing IETF/IRTF Work

This document is complementary to multiple ongoing efforts in the IETF and IRTF that touch on autonomic networking, AI agents for network management, and agent communication. These efforts can be used as inputs when refining protocols that realize the reference points in this draft.

9. Recommendations and Next Steps for IETF Discussion

The following near-term discussion items are proposed:

1. Discuss an agree on the terminology for the agentic AI architecture

2. Orchestration reference framework and reference points: Refine the CX and AIAX reference points and their semantics. Determine what "minimum interoperability" means for each reference point in a multi-vendor environment.

3. Agent registration, discovery, and lifecycle: Identify interoperable mechanisms for agent discovery and capability advertisement, including failure signaling and versioning.

4. Security tokens, authorization, and auditability: Identify how agents authenticate, obtain scoped authorization, and produce auditable action traces when acting on network state.

5. Layer-by-layer automation building blocks: Use the architecture to guide follow-on documents that focus on automation and closed-loop control within specific layers (e.g., routing layer, transport layer, service layer), without conflating those with the inter-agent interface problem.

10. IANA Considerations

TBD

11. Acknowledgements

TBD

12. Normative References

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

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

Authors' Addresses

Muhammad Awais Jadoon
InterDigital Europe Ltd
London
United Kingdom
Email: muhammad.awaisjadoon@interdigital.com

Sebastian Robitzsch
InterDigital Europe Ltd
London
United Kingdom
Email: sebastian.robitzsch@interdigital.com

Carlos J. Bernardos
Universidad Carlos III de Madrid
Madrid
Spain
Email: cjbc@it.uc3m.es