Network Management Research Group
Internet-Draft
Intended status: Informational
Expires: 23 April 2026

C. Guo
China Mobile

October 2025

Large Model based Agents for Network Operation and Maintenance
draft-chuyi-nmrg-ai-agent-network-02

Abstract

Current advancements in AI technologies, particularly large models, have demonstrated immense potential in content generation, reasoning, analysis and so on, providing robust technical support for network automation and self-intelligence. However, in practical network operations, challenges such as system isolation and fragmented data lead to extensive manual, repetitive, and inefficient tasks, the improvement of intelligence level is very necessary. This document identifies typical scenarios requiring enhanced intelligence, and explains how AI Agents and large model technologies can empower networks to address operational pain points, reduce manual efforts, and explore impacts on network data, system architectures, and interfaces correspondingly. It further explores and summarizes standardization efforts in implementation.

Requirements Language

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 RFC 2119 [RFC2119].

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 4 April 2026.

Table of Contents

1. Introduction

1.1. Large Models

Large models refer to AI systems based on deep learning techniques, containing massive parameters (typically billions to trillions). It is trained on large-scale datasets, and is capable of capturing complex patterns and associations, demonstrating outstanding abilities in natural language processing, image generation, decision-making, and reasoning.

Recent breakthroughs in models like GPT-4 and DeepSeek have continuously pushed technical boundaries and enhancing the performance of models.Users can use the capabilities of large models by accessing or deploying inference models, and combining with Fine tuning, Prompt Learning, etc.

The big model has been empowered in multiple vertical domains, like:

* Research: AlphaFold for protein structure prediction, Galactica for scientific paper assistance. Industry: Generative design (e.g., automotive/chip architecture optimization), automated code development (GitHub Copilot).

* Finance: Risk prediction, automated report generation.

In the future, large models will also move towards embodied AI , embedding model capabilities into physical terminals such as robots and autonomous driving, continuously building an open-source developer ecosystem, opening up some model capability interfaces, and promoting industry collaborative innovation.

1.2. AI Agent

Intelligent agent, as an important concept in the field of artificial intelligence, refers to a system that can autonomously perceive the environment, make decisions, and execute actions. It has basic characteristics such as autonomy, interactivity, reactivity, and adaptability, and can independently complete tasks in complex and changing environments. Intelligent agents have the ability to learn and make decisions. Through learning algorithms and data analysis, they can extract useful information from massive amounts of data and form their own knowledge base. In the decision-making process, intelligent agents can comprehensively consider various factors and use methods such as logical reasoning and probability statistics to make the optimal decision. This ability gives intelligent agents a significant advantage in solving complex problems.

There are four design patterns for intelligent agent workflow[LLMbasedAgents]:

* Reflection: Let the agent review and revise the output generated by themselves;

* Tool Use: LLM generates code, calls APIs, and performs practical operations;

* Planning: Let the agent decompose complex tasks and execute them according to the plan;

* Multi-agent Collaboration: Multiple agents play different roles and collaborate to complete tasks.

At present, intelligent agents have been used in the following scenarios:

* Personal assistant:

  - Cross platform task agent: Automatically organize emails, schedule meetings, and manage schedules (such as Microsoft Copilot).

  - Life Butler: Adjust smart homes according to user habits and recommend personalized health plans.

* Industry Intelligence:

  - Financial advisory: Real time analysis of market data, generation of investment portfolio recommendations, and automatic execution of trades.

  - Medical diagnosis: Provide dynamic treatment recommendations based on the patient’s medical history and real-time monitoring data. Industrial operation and maintenance: Predicting equipment failures and scheduling maintenance resources to optimize production line efficiency.

* Virtual world interaction:

* - Game NPC: Intelligent characters with emotions and memories (such as AI driven open world NPCs).

  - Metaverse Guide: Help users explore virtual spaces and provide personalized content recommendations.

* Scientific research:

* - Laboratory assistant: Automatically design experiments, analyze data, and propose hypotheses (such as chemical synthesis agents).

  - Climate simulation: Coordinating multidimensional data models to predict extreme weather and generate response plans.

2. Acronyms & Abbreviations

Large model: Machine learning models with large-scale parameters and computing power are typically constructed from deep neural networks, containing billions or even hundreds of billions of parameters, capable of understanding text, images, speech, and other content, and performing tasks such as text generation, image generation, inference question answering, and scientific prediction.

AI Agent: An AI agent is an intelligent entity with autonomous perception, decision-making, and execution capabilities, driven by goals in dynamic environments.

3. Use case

This section list 3 typical use cases of how AI Agent can facilite network operation and maintenance, which have been proven to be the most effective combination in the practice.

3.1. Scenario 1: Network Migration Operations

The current network undergoes a large number of service migration or device switchover every day/month, which have a high degree of similarity in steps and processes, involving querying and filling a large amount of data and configuration. There are two typical types of migrations: service provisioning (for external service data configuration) and migration change (for internal tasks such as route publishing and network optimization). Large models naturally have the ability to process and recognize massive amounts of data, and intelligent agents can guide the process of each step like experienced experts.

Automation via large models and agents can reduce errors and free human resources. Key tasks include:

* Migration Plan Generation: Designing workflows and deployment strategies.

* Plan Auditing: Checking configurations, compliance, and correcting errors (e.g., typos, hallucinations).

* Automated Execution: Replacing manual configurations with AI-generated scripts, call corresponding systems to finish tasks.

Taking the service provisioning scenario as an example, typically, when doing migration, it was necessary to manually log in the device configuration parameters. Now, through the interaction of the large model, the large model generates a script to distribute the device, also configure and audit it. The agent can call other systems, such as digit twin platform for script testing, view the impact of the changed parameters, and return to the assigned system to reduce manual errors. Finally, based on the analysis of the results, it can achieve automatic distribution when there shows no problem.

3.2. Scenario 2: Network Fault Handling

Traditional fault handling is usually single domain autonomy. When monitoring detects anomalies such as alarms, a fault ticket is generated and sent to the corresponding specialized network domain for solutions. Generally speaking, one single fault point triggers alarms across multiple domains and locations, tickets will be assigned to each involved domain, this results in numerous tickets for each domian, which leads to low processing efficiency.

Different network domains receiving fault tickets will conduct fault analysis, positioning, and report troubleshooting results to the upper monitoring layer for fault management. Each network domain has its own troubleshooting process for different alarms. Based on expert experience, data analysis, and tools, the positioning capability is very comprehensive and reliable, but it heavily relies on manual labor and may requires night duty to respond promptly when faults occur. By training professional fault intelligent agents with fault cases and manually summarized experience, coupled with fine-tuning of professional domain knowledge, domain-specific fault handling agents can identify faults based on alarm types and other information, generate troubleshooting steps (such as data query and tool calling schemes), and call corresponding APIs to perform troubleshooting actions, finally obtaining fault positioning conclusions.

Due to the chain reaction caused by alarms, which usually involve multiple specialties, top-level monitoring agents need to cooperate and coordinate with various domain agents, and positioning conclusion needs to be carried out across specialties. In addition to replacing manual alarm preprocessing and ticket dispatching, agents also need to be connected to alarm systems, performance systems, etc., to perceive abnormal data in the network at any time, trigger fault analysis and positioning processes, and continuously interact with different domain agents for necessary information, find the root

cause of the fault, and finally generate the fault handling report. The top-level monitoring agent can have the alarms, topology and other information of different domains. And it can conduct cross-disciplinary joint consultations, which is more advantageous than traditional methods.

3.3. Scenario 3: Intelligent Assistant of User Complaint Handling

In the past few years, there has been research and development of AI model capabilities in the field of complaints, such as complaint prediction, complaint problem positioning, and complaint warning, to improve complaint handling efficiency, reduce complaint ratios, and play an auxiliary role in various service complaint handling such as home broadband, 4/5G, and IoT. However, this single funtion models lack natural language processing and content generation capabilities, and their role in improving efficiency is limited.

The complaint handling faces challenges such as heavy workload and lack of pre-processing ability in the front-line customer service. Taking the scenario of home broadband as an example, traditional processes allow users to seek help from manual customer service by making phone calls when encountering problems. Frontline customer service staff will answer the user’s questions and provide guidance on simple handling. If it cannot be quickly resolved through simple steps, a problem handling ticket will be generated, and the system will assign maintenance personnel to come to the site. After the repair is completed, a dedicated person will call for satisfaction follow-up. Traditional processes rely heavily on manual labor, and customer service responses have a high degree of repetition and structure. It is necessary to introduce electronic means to systematically improve processing efficiency and replace manual labor.

Introducing the intelligent assistant of user complaint handling can replace customer service staff in answering customer complaint calls, it can achieve the following process:

* Firstly, perform intent recognition on the voice content of the customer’s phone call

* Intelligent diagnosis of identified problems, determining the category and attributes of the problem, and determining the next steps for operation

* Can provide direct response/prompt handling, dispatch, remote operation, and manual processing for user issues

* Intelligent quality inspection and automatic satisfaction follow-up

Leveraging the complaint-handling intelligent agent and its AI-powered diagnostic engine and big data analytics capabilities, we have built an end-to-end service loop of "proactive perception - precise prediction - preemptive resolution". This transforms complaint handling from passive acceptance to active intervention.

Based on real-time network quality monitoring and a historical fault signature database, installation/maintenance technicians can now remotely automate issue resolution and generate solutions before on-site visits. The complaint-handling process shifts from human-driven to AI-driven, task scheduling evolves from reliance on manual experience to autonomous planning and decision-making, and manual operations are upgraded to automated execution or human-machine collaboration. This significantly reduces processing time and lowers repeat complaint rates, reshaping service experiences through intelligent solutions and substantially improving fault resolution efficiency.

During user interactions or technician on-site visits, the system utilizes large model capabilities to call APIs or execute SQL queries for real-time device status and performance metrics, replacing fixed interfaces and manual queries with intelligent integrated access. For complex issues and workflows, the large model-based agent can also invoke small model capabilities (e.g., network traffic fault diagnosis models) mid-process.

When handling complaints, the agent responsible for direct human-complaint interaction can collaborate with other specialized agents (e.g., a Home Broadband Troubleshooting Assistant) through multi-agent cooperation to achieve closed-loop resolution.

4. Architecture and Functionality

Intelligent agents based on large models can automate network operations by coordinating system scheduling and leveraging diverse capabilities of large models. This process involves multiple interactions with systems such as large models and network management systems. Each agent has specialized functions, such as agents for intent understanding or agents dedicated to fault localization and demarcation in specific network scenarios. Current operational systems already provide basic data support, foundational atomic capabilities, and well-defined orchestration workflows for task execution. However, most processes are manually connected, involve repetitive mechanical work, and lack an intelligent coordination "brain". See Figure 1.

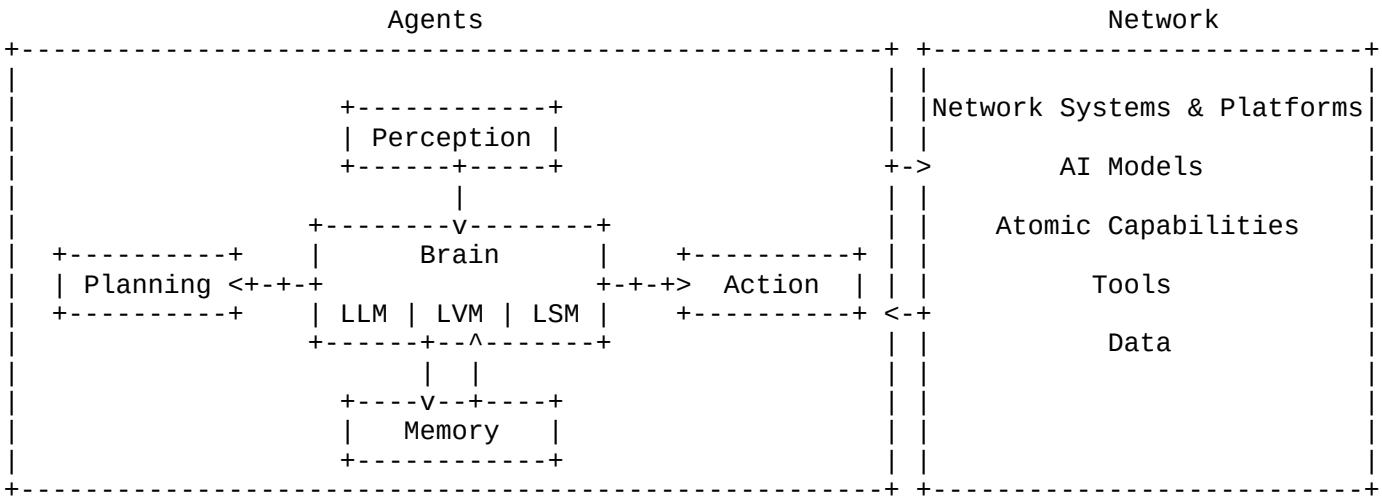

Figure 1: Architecture of Large Model based Agents

Functions of Agents:

* Intent Recognition: Understand and interpret user input intentions. Determine whether subsequent tasks require identifying suitable agents or multi-turn dialogues to complete intent recognition and parsing.

* Intent Classification and Analysis: Decompose tasks based on recognized user intent.Categorize tasks according to different functional requirements.

* Perception: Proactively receive alarms, threshold-exceeding notifications, or environmental change information, issuing warnings when necessary.Accept task requests from other systems, potentially involving multimodal data processing.

* Memory:

  - Long-term memory: Stores user habits, domain-specific processing experiences (e.g., failure/success cases, encountered faults) in knowledge bases.

  - Short-term memory: Caches temporary processing data (e.g., context).

* Agents perform reflection and error correction by interacting with long-term memory and contextual information.

* Planning: Analyze and decompose intent based on task objectives and learned knowledge. Orchestrate subtasks (e.g., breaking complex problems into simpler ones). Identify required system components (other agents, large models, APIs, etc.).

* Decision-Making: Finalize execution plans and match workflows to current tasks. Generate instantiated, executable solutions by aligning system components, data, and model strategies.

* Execution: Convert orchestrated results into network-understandable commands. Execute tasks by mobilizing resources and dynamically adjusting based on feedback.

* Multi-Agent Collaboration:

  - Team Collaboration: Enable coordinated teamwork among multiple agents.

  - Competitive Collaboration: Manage competitive relationships to avoid efficiency loss.

5. Observed Requirements

Based on the aforementioned practical use cases regarding the implementation of intelligent agent functional modules, this chapter will summarize and analyze the technologies, challenges and constraints encountered during the deployment and implementation of agents in network operations. It will also propose potential requirements across different dimensions.

5.1. Data Requirement

Data is the foundation of agents to take effect, which includes:

* Data used for training and inference: The data used for training and inference, which is typically used before the agent provides services to users, it can be expert knowledge in operation and maintenance processes, logs, configuration rules, policy knowledge, case manuals, alarms, network topologies, fault reports, and more. The data here is used to teach big models how to do it, usually historical data. The performance of the model based on data learning can be adjusted through reward functions, etc. The actual implementation cases in the later stage can also be stored in long-term storage as reinforcement learning, which can be used as a reference for generating solutions later.

* Dynamic data during task execution: Dynamic data can also be categorized into two types: User intent data (e.g., natural language task descriptions), operational data from systems. For user-expressed intents, typically extracted during user interactions (which may include multimodal inputs like voice or images), agents expose dedicated interfaces for user input, and upon receiving it, performs intent recognition before executing subsequent steps. The data used for system operation refers to the constantly collected data in the network, such as monitoring performance indicators, alarms and other data, as well as messages from other systems, such as device network access/upgrade information.

Requirements:

For training data:

* It mainly reflects integrity. The training data of a large model not only needs to reach a certain number of parameter levels, but also needs to ensure comprehensive and accurate coverage, reducing interference data.

* In order to enable intelligent agents to trigger subsequent processing based on alarm and other data in the system, it is necessary to manually establish expert experience and rules, and even establish new interfaces to support the problem meanings represented by different data phenomena. This part can be standardized.

For dynamic data:

* Modal transformation. For example, if it is an LLM based agent, users need to convert text before extracting intent when receiving images.

* How to extract intent. This requires understanding the running data, knowing what the indicators represent, which type of analysis and processing should be triggered (consistent with the training data requirement 2 mentioned above), and extracting which type of features from the input for subsequent intent recognition. This process requires classifying the network’s events, behaviors, etc. in advance.

* Align the formats of different input data. The main solution is to solve the problem of not being able to directly understand and call each other when different APIs and tools are accessed or called with different formats. The MCP protocol[MCP] in the industry can ease this problem. MCP converts tool capabilities

into standardized function semantics through JSON Schema, enabling large models to implement zero sample calls based on tool descriptors without the need for prior learning or fine-tuning. This enables the integration of large language models with external data sources and tools, and is used to establish secure bidirectional connections between large models and data sources. MCP defines a set of universal communication protocols, data formats, and rules that can be simplified for development, flexible, real-time responsive, secure, compliant, and scalable. It processes both local resources (such as databases, files, services, etc.) and remote resources (such as APIs like Slack or GitHub) through the same protocol, hence it is known as the USB-C port for AI applications. In addition to MCP, enterprises can also use their own agent protocols to call different systems and adopt and design unified format conversion standards.

5.2. Network Intent Recognition Requirement

Network intent recognition refers to how to correctly understand the significance of various indicators for the network when receiving different data from the network, and how to analyze the collected indicators based on existing network operation and maintenance practices, extracted from human experience and knowledge manuals. On the other hand, how human input is transformed into network goals and actions can also be a part of network intent recognition. The current implementation. This is where standardization is needed.

To support the implementation of closed-loop, a knowledge graph model can be constructed to inject large models. After receiving user or network intent, perform entity recognition, retrieve knowledge graph, complete entity linking, and output structured intent[CCSAKG].

Graph Neural Network (GNN) refers to an algorithm that uses neural networks to learn graph structured data, extract and discover features and patterns from graph structured data, and meet the requirements of graph learning tasks such as clustering, classification, prediction, segmentation, and generation. In the construction of network operation and maintenance knowledge, entities such as devices, links, and alarms can be abstracted into graph structures, and hidden relationships can be mined through message passing mechanisms to achieve the transformation of operation and maintenance problems from "passive response" to "active cognition". In the graph structure, physical devices can be nodes, connection relationships can be edges, and they can also be used to construct event correlation graphs.

Requirement: Establish a knowledge graph in the field of network management and operation, sort out the interrelationships and collaborative relationships between entities in different events or categories, and assist large models in understanding and inference by retrieving relevant entities and relationships from the knowledge graph.

5.3. Self Close Loop Requirement

Intelligent agents need to have the ability to complete tasks on their own without human intervention. In addition to the aforementioned requirements for data input and intent understanding, the operation and maintenance agent should also have execution capabilities, including generating task plans, calling APIs, and completing actions. This process involves issues such as permission management and security confidentiality.

5.4. Resource Requirement

The resource requirements for agent applications mainly include three aspects: storage, computing force, and network infrastructure.

Storage should consider cloud edge coordination. In the training and use of intelligent delivery, in addition to building a knowledge base, contextual information, user preferences, historical records, etc. also need to be continuously stored. Some content may need to be stored on the user side, while others are more suitable for obtaining in the cloud. Resource coordination and balance are key issues. In addition, intelligent agents require a large amount of storage resources, and so much information requires efficient access solutions.

The computing force part mainly considers the allocation of computing resources as needed based on the analysis of task complexity during the inference process. The PD separation architecture can be adopted to improve inference efficiency.

The network infrastructure mainly provides low latency and high bandwidth network support, which can use SRv6/G-SRv6 to compress packet headers, improve cross domain scheduling efficiency, and optimize long-distance data transmission throughput through RDMA.

5.5. Muti-Agent Collaboration Requirement

In order to complete specific tasks, some can be accomplished through self-closed-loop of agents, while others require collaboration among multiple agents. This includes three communication modes: interaction between humans and agents, interaction between agents and the environment, and interaction between different agents. The interaction between humans and agents is the input of human intentions, usually in the form of natural language, and the input content is multimodal (voice, text, images, etc.). The interaction between agents and the environment is usually achieved through interface access, which can obtain data, and triggering subsequent actions. The interaction between different agents is the input and output interaction among agents, which involves both multimodal information exchange and system interface level interaction.

In order to enable different agents to cooperate with each other, complete tasks together, and achieve a large closed loop, the following requirements are observed:

* Agent discovery. Unlike traditional communication, where connections are established based on known addresses, communication through agent collaboration is temporary and the objects completing tasks are not unique, requiring self discovery to determine the communication objects. To support this, the communication network needs to establish an intelligent agent discovery mechanism.

* Agent authentication. Agents need network authentication to access the network. Unlike mobile devices, agents have many behaviors that cannot be distinguished from human behavior. Therefore, targeted agent authentication schemes need to be proposed.

For the above two requirements, A2A[A2A], AGNTCY and others in the industry have proposed their own protocol implementations, but the complete solution that requires network adaptation involved in the above process is still blank.

* Unified Agents connection control and arrangement. Due to the frequent team communication and partner searching required between intelligent agents to complete tasks, different vendors may provide intelligent agents with similar functions. It is best to have a unified intelligent agent management platform for easier team building. Additionally, it also includes the scheduling and allocation of underlying computing resources, as well as state management.

* Standardized communication protocol between Agents for Network Operation and Maintenance at different levels. There is a mutual calling relationship between agents at different levels in work Operation and Maintenance. For example, the fault handling agent calls multiple network professional domain agents, and the network professional agent calls the underlying controller or network management system to complete the analysis and processing of faults. Different agents need to communicate with each other, and the fundamental requirement is to agree on the information parameters and return content that need to be transmitted between the upper and lower layers for a certain event. Therefore, the requirments are:

  - A standard inter level protocol is needed, which is not related to the natures or functions of agents within the level, and standardizes the input and output of agents.

  - A unified communication format is needed to ensure semantic consistency.

  .

6. IANA Considerations

This document has no requests to IANA.

7. Security Considerations

This document describes concepts and definitions of agent in network OAM. As such, the following security considerations remain high level, i.e., in the form of principles, guidelines or requirements.

8. References

8.1. Informative References

[A2A] Google, "Agent2Agent(A2A) Protocol：https://google.github.io/A2A/#/", April 2025.

[CCSAKG] CCSA, "SR 460-2024 Study on building knowledge graph for network", January 2024.

[LLMbasedAgents] Cheng, Y. Cheng., Zhang, C. Zhang., Zhang, Z. Zhang., Meng, X. Meng., and S. Hong. Hong, "Exploring Large Language Model based Intelligent Agents: Definitions, Methods, and Prospects.", January 2024.

[MCP] Anthropic, "Model Context Protocol (MCP) : https://www.anthropic.com/news/model-context-protocol", November 2024.

8.2. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

Author’s Address

Chuyi Guo
China Mobile
Beijing
100053
China
Email: guochuyi@chinamobile.com