GTI Network Agent and NetMCP Technology White Paper

https://gtigroup.org/

GTI Network Agent and NetMCP Technology

White Paper

Version:	v1.0
Deliverable Type	□ **Procedural Document** □ **Working Document**
Confidential Level	☑ **Open to GTI Operator Members** ☑ **Open to GTI Partners** □ **Open to Public**
Program	5G-AxAI
Working Group	N/A
Project	Project 1: Network Intelligence
Task	N/A
Source members	China Mobile, CAICT, Huawei, ZTE, vivo, OPPO, CICT Mobile, Noki, Innomix
Support members	
Editor	
Last Edit Date	
Approval Date	

Confidentiality: This document may contain information that is confidential and access to this document is restricted to the persons listed in the Confidential Level. This document may not be used, disclosed or reproduced, in whole or in part, without the prior written authorization of GTI, and those so authorized may only use this document for the purpose consistent with the authorization. GTI disclaims any liability for the accuracy or completeness or timeliness of the information contained in this document. The information contained in this document may be subject to change without prior notice.

Document History

Date	Meeting #	Version #	Revision Contents

Table of Contents

Introduction

With the widespread deployment of 5G and the evolution toward 6G, traditional network operation can no longer keep pace with dynamic service requirements. The development of Artificial Intelligence (AI) technologies provide new opportunities to address this challenge. By introducing AI agents, network operation evolve from passive response to an autonomous closed loop of perception, decision-making, and execution. This evolution significantly enhances efficiency, ensures the high-quality delivery of diversified services, and advances network intelligence.

This white paper draws upon GTI's exploration and practice in the in-depth convergence of network and AI. The white paper is organized into four parts. First, it highlights four concepts of network running agents: native intent, endogenous intelligence, autonomous policy, and symbiotic ecosystem. Second, centered on mobile communication network operation, it proposes a three-layer system architecture for network running agents. The architecture consists of the atomic capability layer, intelligent execution layer, and orchestration & coordination layer. The layers feature capability toolization, Model Context Protocol (MCP) telecom-transformation, workflow orchestration, multi-agent collaboration, and an open intent portal. Third, it provides a detailed description of the Network-Aware Model Context Protocol (NetMCP), which enables interaction between network running agents and Network Functions (NFs). Finally, it outlines typical application scenarios and specifies the industry progression path, offering practical reference and guidance for technology implementation.

GTI hopes to work closely with partners to explore scenarios, technologies, products, and applications, and to jointly promote the development of the network running agent industry and the prosperity of its ecosystem, building a solid foundation for the transformation toward network intelligence.

1 Background and Driving Forces

Digitalization and intelligence represent the hallmarks of the new wave of technological revolution and serve as the primary driving forces behind next-generation information technologies. In recent years, the world has been undergoing a new wave of technological revolution and industrial transformation. As a cornerstone supporting the digital, networked, and intelligent transformation of the economy and society, 5G has become a key driver of global digital economy growth. With the large-scale deployment and deepening application of 5G networks, 5G not only technically drives the transformation and upgrading of traditional industries, but also creates broad opportunities for the convergence and innovation of emerging technologies such as AI, the Internet of Things (IoT), and the industrial Internet. In doing so, it continuously injects strong momentum into digital transformation across diverse fields.

By the end of 2025, China had deployed 4.83 million 5G base stations covering 1.204 billion 5G mobile users, achieving a penetration rate of more than 80%. At the same time, massive IoT terminals are increasingly being deployed in fields such as industry, transportation, and healthcare, while mobile communication devices continue to grow at an explosive pace. However, the rapid expansion of network scale reveals the limitations of traditional architectures in meeting increasingly diverse service requirements. This poses risks of Quality of Service (QoS) fluctuation and deterioration, while also driving up Operating Expenses (OPEX) due to mounting resource scheduling pressures.

The rapid development of AI technologies offers new opportunities to address these challenges. Traditional network struggles to quickly and accurately respond to diverse service demands in complex and dynamic environments. By introducing AI agents with autonomous decision-making capabilities, networks can achieve real-time state awareness, dynamic environment learning, and adaptive strategy optimization. On the one hand, AI agents can collect and analyze key information in real time (e.g., network traffic, node loads, link quality, resource utilization, user experience, and terminal distribution). On the other hand, they can quickly generate and execute optimal scheduling strategies in complex scenarios. For example, when network congestion occurs, they can promptly adjust data forwarding paths and allocate bandwidth efficiently to ensure the quality of critical services, significantly enhancing network flexibility, reliability, and stability. Moreover, a multi-agent collaborative model can be applied in large-scale network scenarios, where different agents are responsible for various regions or functional modules of the network. Through cross-domain collaboration and information exchange, network-wide optimization can be achieved, which is crucial for efficiently managing billions of device connections and massive data transmissions in 5G and future 6G network. Ultimately, this promotes the network's upgrade from a 'passive response' mode to an autonomous operation model (i.e., active perception, intelligent decision-making and closed-loop optimization), achieving improvements in both service quality and operational efficiency.

Especially over the past year, the application of agent technology has become a major focus across global industries. In general domains, Anthropic and Google have released the MCP and the Agent-to-Agent (A2A) protocol successively. MCP tackles the challenge for agents to invoke tools, while A2A enables interaction between agents, together laying a solid foundation for the standardization of agent technology. The deep integration of agent technology with communication network, however, will require strong standardization efforts. The Internet Engineering Task Force (IETF) has outlined potential directions for advancing agent-based Internet standards and will further clarify scenario requirements and technical standardization pathways. The European Telecommunications Standards Institute (ETSI), through its Experiential Networked Intelligence (ENI) group, has published multi-domain standards that lay the groundwork for integrating agents into communications networks. Likewise, the 3rd Generation Partner Project (3GPP) has formally included the definition of agents in its 6G requirement study report (TR 22.870) under SA1, while SA2 has incorporated agents into research on 6G network architecture. At the same time, more than 80 industry, academic, and research institutions launched the Joint Initiative on Co-construction and Sharing of Agent Protocols at the International Cooperation Forum on Standardization of Artificial

Intelligence, held during the 2025 World Artificial Intelligence Conference (WAIC), promoting standardization and open-source sharing. In China, the China Communications Standards Association (CCSA) established a sub-working group on agent communication networks to consolidate industry efforts, advance research on scenario requirements, system architecture, and key technologies for the convergence of 6G and agents, and foster a collaborative landscape internationally.

In conclusion, the transformation toward network intelligence is being driven by multi-dimensional factors and pressing requirements. From the perspective of technological evolution, the connection of hundreds of millions of devices, massive data transmission, and millisecond-level latency requirements in 5G/6G networks pose significant challenges to traditional network management models. To address the exponential growth in network complexity, more efficient and intelligent management approaches are urgently needed. From the perspective of industry requirements, emerging application scenarios such as the industrial Internet, smart city, autonomous driving, and telemedicine place unprecedentedly stringent demands on network real-time performance, reliability, and flexibility. Consequently, networks must evolve from best-effort delivery to deterministic assurance. From the perspective of operation efficiency, the continuous expansion of network scale has created increasingly pressing challenges, including high manual O&M costs, delayed fault response, and low resource utilization. To overcome these issues, operators are eager to cut costs and boost efficiency through intelligent solutions. All these multi-level and multi-dimensional driving forces converge on a single imperative: communication networks must undergo a fundamental shift from traditional reactive management to intelligent and proactive optimization. The deep integration of AI technologies—particularly agent technology—will be the key technical pathway to realizing this transformation.

2 Vision and Evolution of Network Agents

This chapter systematically presents the vision, concepts, and evolutionary path of network running agents, establishing a theoretical foundation for the subsequent technology framework and application scenarios.

2.1 Vision and Concepts

With the widespread deployment of 5G networks and the evolution toward 6G technologies, traditional network operations, which rely on manual configuration and static rules , can no longer meet dynamic and changing service requirements. Breakthroughs in AI technologies have created new opportunities for the transformation toward network intelligence. Within this shift, network running agent technology represents an important direction in the evolution of communication network toward intelligent operation.

The vision for network running agents is to build a network ecosystem with autonomous perception, intelligent decision-making, and closed-loop optimization capabilities. This marks a transformation from passive response to proactive service, enabling human-like cognition and ultimately evolving into a truly intelligent digital infrastructure.

Based on the exploration of network intelligence, four core concepts of network running agents has been introduced. These concepts are mutually reinforcing and organically integrated, collectively forming the theoretical foundation of network running agents:

- **Native Intent**

Network makes a leap from simple instruction execution to intent understanding. Network is natively capable of intent understanding. It can directly receive requirements expressed in natural language and translate those into precise network operations through semantic parsing and intelligent inference. Users can access personalized intelligent services without having to master complex network technologies, thereby achieving truly user-adaptive networks.

- **Endogenous Intelligence**

AI capabilities shift from being external applications layered onto systems to becoming natively converged. AI capabilities are deeply embedded across every layer of the network architecture. As a result, network possesses endogenous intelligence that enables them to independently detect environmental changes, dynamically learns and optimizes policies, and proactively predicts risks. Network is endowed with human-like cognitive decision-making, enabling them to autonomously evolve and continuously optimize within complex environments.

- **Autonomous Policy Generation**

Network policies evolve from static pre-configuration to dynamic generation. Network can generate, evaluate, and optimize policies based on real-time service perception and multi-dimensional data analysis, enabling adaptive decision-making and elastic control. Policies are continuously iterated during service operation, forming a closed loop that drives autonomy from global optimization to local execution. Ultimately, this process produces an intelligent decision-making response system that enables networks to adapt to diverse environments.

- **Symbiotic Ecosystem**

The transition from closed operation to open collaboration is accelerated. A unified standards framework opens up network atomic capabilities, enabling deep integration between network and the agent ecosystem. Internal and external agents collaborate seamlessly through standardized protocols, forming a multi-entity network of value creation and propelling the transformation of network from infrastructure provider to intelligent service ecosystem

2.2 Evolution of the Technology Roadmap

The convergence of communications network and AI agents creates opportunities for transforming production methods and reshaping network architectures. Based on the levels of autonomous task execution and intent perception, this convergence can be structured into three layers, which evolve in parallel and reinforce one another.

- **Agent-empowered Network (Agent4Net)**

Agent technology is deeply integrated into the network architecture and its core operational mechanism. At this layer, agents evolve beyond external applications, embedding themselves as core components of NFs. Agents empower networks with advanced perception, decision-making, and execution capabilities, driving intelligent upgrades including automatic fault diagnosis, smart resource scheduling, predictive maintenance, and personalized service delivery, all of which significantly boost operational efficiency.

- **Network-empowered Agent (Net4Agent)**

As an intelligent infrastructure, the network provides interconnection, identity authentication, orchestration and coordination, and information exchange for external agents. Standardization organizations such as 3GPP have been actively studying AI agent application scenarios as a key requirement of 6G, highlighting the rapid progress at this layer. At this layer, the network primarily provides connectivity and support, ensuring communication and coordination services for diverse agents, while maintaining the stable operation of the agent ecosystem.

- **Agentic Network (AgenticNet)**

The network itself evolves into a distributed autonomous system composed of agents. NFs are reconstructed as agents, and network capabilities can be agentified to enable autonomous, closed-loop operations. Traditional NFs are transformed as specialized agent modules—specifically orchestration, perception, and data agents. Driven by Large Language Models (LLMs) in the network domain, these agents enable autonomous coordination and closed-loop autonomy. The network architecture evolves from function-oriented to agent-oriented, empowering each component to autonomously make decisions and collaborate.

3 Architecture for Network Running Agents

Network agents apply to network operation, O&M, and service scenarios. As the core for network capability implementation, the network operation layer is critical for ensuring user experience and service quality. It faces stringent real-time performance requirements, complex and changeable scenarios, and strong timeliness of decision-making. As such, the network operation layer is a vital landscape for the real-world application of agent technologies.

This white paper focuses on the network operation layer and describes how to implement autonomous network perception, intelligent decision-making, and efficient execution based on the agent architecture.

3.1 Overall Architecture

The technical framework of network running agents adopts a hierarchical and decoupled three-layer architecture, which consists of the atomic capability layer, intelligent execution layer, and orchestration & coordination layer from bottom to top. While decoupled, these layers coordinate closely to form a complete closed-loop of perception, decision-making, and execution. This loop represents the five core features, including capability toolization, MCP telecom-transformation, workflow orchestration, multi-agent collaboration, and an open intent portal.

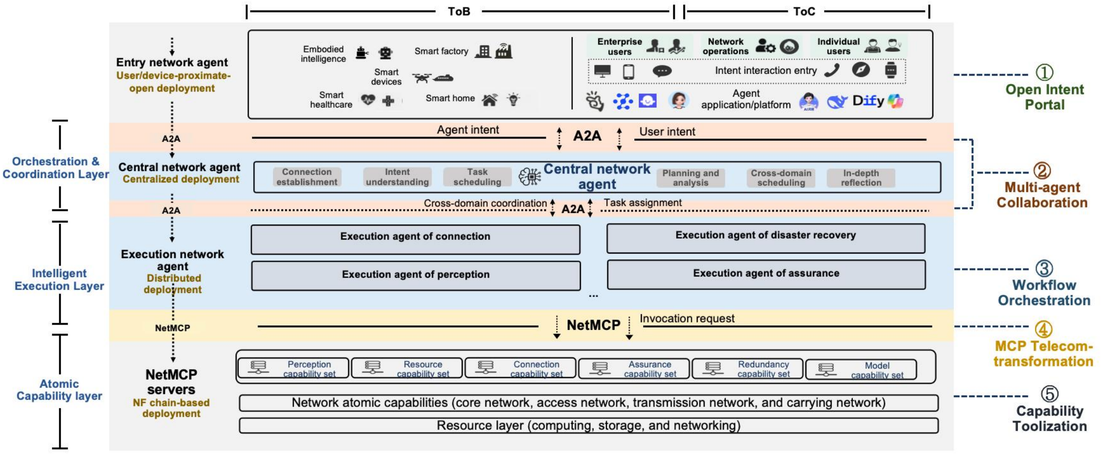


Figure 1 Network running agents using the architecture with three layers and five features

3.1.1 Atomic Capability Layer

As the foundation of the entire technical framework, the atomic capability layer is where the concepts of capability toolization and MCP telecom-transformation are realized and all network operations are ultimately executed.

Capability Abstraction and Encapsulation

Complex physical and virtual network functions (such as core network, base station, transport, computing power, and storage) at the underlying layer are uniformly abstracted and encapsulated as standardized atomic capabilities. Each atomic capability has deterministic input and output APIs, execution logic, and error handling mechanisms. Various atomic capabilities compose a capability set covering data collection, resource management, connection control, performance assurance, status perception, and redundancy recovery.

Unified Capability Invoking

A NetMCP server is introduced. It provides a unified capability invoking API for external systems. NetMCP is a protocol extended from the standard MCP to adapt to telecom network scenarios. It supports standardized interconnection between AI agents and communications network capabilities. In this way, upper-layer agents can invoke required capabilities through the standard NetMCP without considering the implementation complexity at the underlying layer. The NetMCP server also provides service governance functions, such as capability registration and discovery, load balancing, and failover.

3.1.2 Intelligent Execution Layer

As the "nerve center" that links the underlying capabilities with the upper-layer decisions, the intelligent execution layer manifests the concept of workflow orchestration. It decomposes macro policies at the orchestration layer into a specific sequence of executable tasks.

Intent Translation

This function is used to decompose the high-level policies on the central agent at the upper layer into structured task procedures and parameter settings.

Workflow Engine

It matches the optimal template from the workflow pool based on the task type, orchestrates and generates a strict execution workflow, and supports complex logic, including sequential, parallel, and conditional branching.

Workflow Pool and Template Framework

A comprehensive workflow template library is maintained. It is used to foster mature solutions such as user quality of experience (QoE) evaluation, cell congestion prediction, and experience guarantee policy delivery. The template-based design enables quick replication and standardized execution in complex service scenarios.

Enhanced Telecom-grade Protocol

Agents can efficiently and reliably run in the telecom network environment based on the enhanced telecom-grade protocol NetMCP. This protocol supports key functions such as identity authentication, capability negotiation, execution context sharing, and security authorization.

3.1.3 Orchestration & Coordination Layer

As the "brain" for agents, the orchestration & coordination layer implements the core concepts of intent entry openness and agent coordination. It is responsible for understanding high-level intents, planning and analysis, and cross-domain resource coordination.

Central Intelligent Orchestration

The LLM-based central agent is the core of this layer. It excels at natural language understanding (NLU), reasoning, and planning. After receiving a user's natural language intent or high-level system instruction, the central agent performs in-depth semantic parsing and complex task decomposition to generate executable macro policies and task schedules. The central agent plays a core role in overall planning and coordination. It is responsible for task breakdown, resource scheduling, progress coordination, and result aggregation, implementing global, intelligent coordination of network operations.

Multi-level Agent Coordination

Multi-level coordination is orchestrated among agents via standardized protocols. Externally, the central agent establishes collaboration relationships with external agents, including user-side agents (such as China Mobile's Lingxi, Doubao, and DeepSeek) and embodied agents (such as drones, smart vehicles, and industrial robots) through the A2A protocol. In this way, network resources can be dynamically applied for and network services can be accessed in an intelligent manner. Internally, the central agent coordinates with execution agents in each domain through the A2A protocol, thereby creating a clear hierarchical coordination structure. In addition, the execution agents invoke services at the atomic capability layer through the NetMCP to build an end-to-end (E2E) collaboration path from intent understanding to capability execution. This ensures efficient execution and reliable coordination of cross-domain network operation tasks.

3.2 Key Technologies

3.2.1 NF Capability Toolization

A tool is a general term for various resources, functions, or services that an agent invokes to achieve specific objectives, handle complex tasks, or interact with external environments. Tools are important means for agents to extend their capabilities and compensate for their inherent limitations. They help agents more efficiently complete tasks that are difficult to handle based solely on their own knowledge bases or reasoning capabilities. On communications networks, NF capability toolization aims to abstract, encapsulate, and standardize various functions and capabilities of NFs. Following this, NF capabilities can be regarded as tools and flexibly invoked, combined, and managed by agents. In this way, agents can efficiently schedule network resources and deploy network services.

(1) Key Elements for the NF Tool Invoking API

The following basic elements are mandatory for the tool invoking API of an NF:

Tool name: It identifies a network tool and is unique to the tool provider. The name of a tool should be relevant to its functions to help agents understand the tool's practical purposes.

Tool description: It is a brief description of the tool's purposes, usually expressed in natural language. Compared with the tool name, the tool description includes more details about the tool's functions, helping agents understand its practical purposes.

Tool input schema: It describes the information about all parameters required for invoking a tool. The information about a parameter should include at least the parameter name, parameter type, parameter description, and parameter necessity. Each parameter name should be unique. The parameter description contains the actual meaning of a parameter, usually expressed in natural language. The parameter necessity indicates whether the parameter is mandatory.

Tool output schema: It describes the structured return format of a tool's execution results, helping agents better understand and correctly process the output data of a tool. Similar to the input schema, the output schema should include at least the parameter name, parameter type, parameter description, and parameter necessity.

In addition, to ensure secure, controllable, and efficient tool invoking, the following enhancements need to be covered:

Tool permissions: Levels of permissions required to invoke tools. Tool invoking approaches include public invoking, token-based invoking, and invoking by specified objects. Public invoking means that a tool can be invoked by all agents without additional authentication. Token-based invoking means that a tool can be invoked only by agents with access tokens. Invoking by specified objects means that a tool can be invoked only by specified agents.

Constraints on the parameters for tool invoking: Value ranges of parameters and inter-parameter constraints. The parameters are used to help agents generate as reasonable tool invoking behaviors as possible, preventing the network stability from being affected by insecure tool invoking behaviors.

Tool category label: It briefly describes the category information (such as the connectivity, sensing, and data capabilities) about an NF tool. It can be used to help agents quickly sort required tools and centrally register tools of the same type.

(2) Core Features of NF Tools

NF tools are encapsulated with predefined atomic network capabilities and have the following core features:

Function specificity: Each NF tool is responsible for only one clear and specific network task, without redundant functions. For example, one NF tool provides only the function of adjusting the QoS policy for a specified user or collecting specified data.

High reusability: One NF tool can be used in various scenarios. For example, the data collection tool can be used for both user mobility analysis and network congestion analysis.

Atomicity: An NF tool cannot be segmented into smaller capability units with practical significance. It should be noted that the execution of an NF tool may involve the coordinated interaction of multiple NFs, or may involve only a single NF. For a single NF, a tool invoking API represents a novel encapsulation of its service-based interfaces (SBIs). This specific form is more suitable for agents to understand both the API's functions and invoking patterns. For multi-NF interaction, a tool invoking API needs to be configured with all necessary parameters required for SBI-based interaction between NFs in a process, and the execution may involve multiple steps.

Composability: NF tools are basic blocks for building more complex capabilities. By invoking, orchestrating, or combining NF tools, agents can provide powerful, composite capabilities to meet service requirements in more complex scenarios.

Openness: NF tools are not only consumed by agents on the network, but also by trusted and authorized terminal agents and third-party agents. This helps build a brand-new terminal-network synergy paradigm and deeply empower vertical industries.

(3) Dynamic Management of NF Tools

To address the complexity of network dynamics and diverse task demands, a dynamic management mechanism for NF tools has been established.

On one hand, this mechanism continuously monitors telemetry such as tool availability and load. It ensures system resilience by deprecating invalid tools, automating version updates, and performing load balancing across redundant key capabilities, thereby maintaining the operational integrity of the agent.

On the other hand, the mechanism moves beyond static management to handle the shift from basic connectivity to complex service assurance. By aggregating capabilities into functional toolsets, it enables the agent to autonomously discover and match resources through task decomposition. This allows for the on-demand loading of new toolsets and the offloading of redundant ones, ensuring elastic resource allocation and high scalability through seamless dynamic updates.

3.2.2 Workflow Orchestration and Execution System

The workflow orchestration and execution system serves as the core mechanism for network running agents to implement intent-to-action conversion, undertaking the critical mission of automating complex service requirements. Given the challenges of cross-domain coordination, time-sequence sensitivity, and the intricate status of network tasks, traditional manual configuration can no longer meet modern intelligence demands. By leveraging an architecture of layered coordination, intelligent translation, and template-driven orchestration, the system enables E2E automation—from user intent understanding to precise network operations.

Currently, network operation faces two core challenges. One is how to accurately convert high-order intents expressed in natural language into executable operation sequences. The other is how to implement efficient system operation and continuous optimization while ensuring execution reliability. The workflow orchestration and execution system is the intelligent solution specifically designed to address these two challenges.

1. Template-based Workflow Design

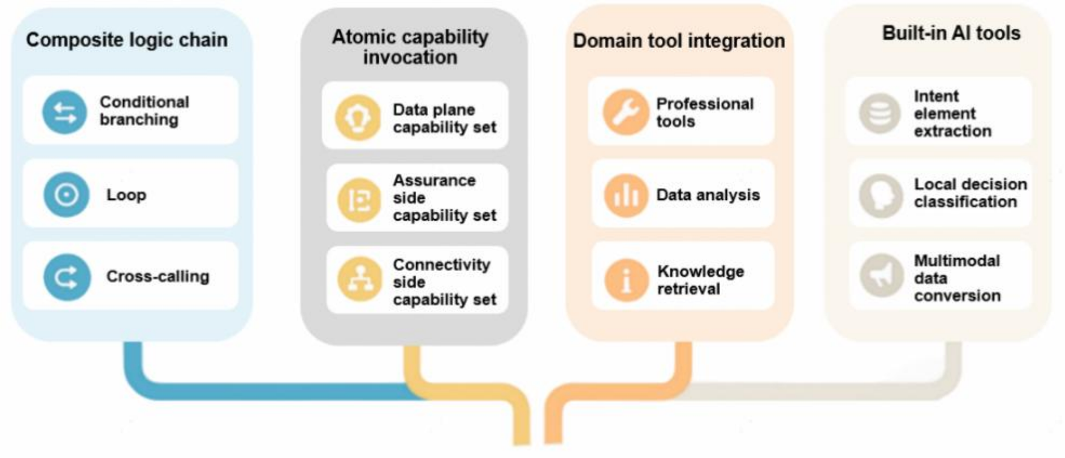

Composite logic chain
Conditional branching
Loop
Cross-calling
Atomic capability invocation
Data plane capability set
Assurance side capability set
Connectivity side capability set
Domain tool integration
Professional tools
Data analysis
Knowledge retrieval
Built-in AI tools
Intent element extraction
Local decision classification
Multimodal data conversion

Figure 2 Workflow capability system

The workflow engine establishes a comprehensive capability framework leveraging a template-driven architecture.

Core Feature	Technical Implementation	Service Benefits
Flexibility	Turing-complete capability library and modular architecture	Enables flexible orchestration of diverse automated network tasks
Efficiency	Template-based design for out-of-the-box deployment of common tasks	Significantly enhances execution efficiency by eliminating redundant orchestration
Stability	Predefined processes providing deterministic execution paths	Superior to real-time generation, ensuring robust operational reliability

Table 1 Three core features of the workflow system

The workflow module capability system covers four categories: composite logic chains, atomic capability invocation, domain tool integration, and built-in AI tools. Together, these form a comprehensive capability spectrum spanning from basic operations to complex scenarios.

During the workflow orchestration and policy generation phase, the system employs a dual-engine mechanism—leveraging both historical experience and LLM mining. This approach balances the carrier-grade stability required for networks with the advanced reasoning capabilities of agents.

- **Historical Experience & Expert Rule Library:** The workflow engine inherits mature assurance policies and expert rule libraries from live networks (e.g., standard QoS parameter sets for high-load scenarios). This ensures deterministic and secure basic service assurance while mitigating policy cold-start risks.

- **LLM Mining & Policy Evolution:** Leveraging the reasoning and generalization capabilities of LLMs, the system expands the solution space beyond historical policies. Particularly in complex edge scenarios (e.g., high interference or weak coverage), the LLM explores optimal parameter combinations—such as the dynamic trade-off between upward frequency-selective scheduling and closed-loop power control—that traditional rules often fail to address. Verified policies are then solidified into the agent's memory, enabling the autonomous evolution of the policy library.

2. Intelligent Translation Model

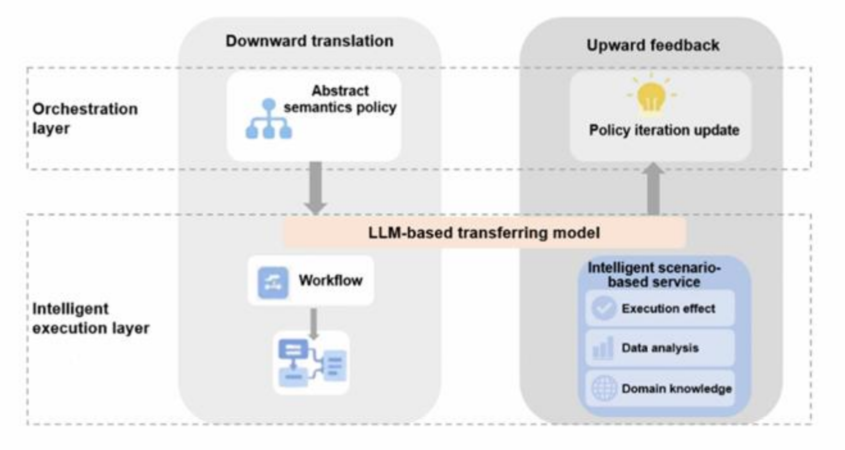


Figure 3 Upward and downward translation

The intelligent translation model is the core component of the execution agent, establishing a bidirectional semantic mapping capability powered by LLMs.

- **Downward Translation (Policy Implementation):** This process decomposes abstract policy instructions from the central agent into structured workflows executable by the execution layer. It encompasses complete operational logic, including parameter extraction, conditional branching, and exception handling.

- **Upward Feedback (Semantic Extraction):** This process summarizes execution-layer outcomes and data, extracting and converting them into high-level semantic insights comprehensible to the upper layers. This feedback loop supports the continuous optimization and dynamic adjustment of policies.

3. Workflow Triggering Mechanism

The workflow system supports multiple triggering modes to accommodate diverse application scenarios.

External Invocation Paths

- **Direct API Invocation:** Service requests directly trigger workflows through standardized APIs.
- **Agent Coordination:** Third-party agent platforms invoke workflows or NF atomic capabilities through NetMCP.

Internal Forwarding Paths

- **Event Policy-driven:** Automated triggering based on real-time network telemetry (e.g., alarms and threshold crossings) or predefined policies (e.g., scheduled tasks and conditional logic).
- **Intelligent Proactive Triggering:** The central agent proactively initiates workflow execution based on integrated perception, monitoring, analysis, and reasoning.

4. Cross-Domain Coordination Orchestration

Network operation tasks are inherently cross-domain, requiring global coordination for E2E assurance. The workflow system addresses these challenges through a "Centralized Orchestration+Distributed Execution" model.

- **Central Agent Orchestration:** The central agent aggregates global information, formulates high-level policies, and performs task decomposition to assign objectives to specific domain agents.
- **Domain Agent Execution & Translation:** Upon receiving tasks, domain agents perform downward translation to convert abstract instructions into executable, structured workflows.
- **Cross-Domain Capability Invocation:** Agents can invoke workflows from other domains as capability nodes, enabling seamless cross-domain coordination through inter-agent calling.
- **Message Bus & Assurance Mechanism:** A high-speed cross-domain message bus facilitates efficient communication and coordination. The central agent monitors task progress in real time, dynamically schedules computing resources, and provides self-healing capabilities to manage disconnections, ensuring the reliable delivery of long-running tasks.
- **Optimized Agent Management:** Based on scenario-specific characteristics, the central agent intelligently consolidates highly correlated agents. This reduces scheduling complexity and minimizes context management overhead for the central agent.

The workflow orchestration and execution system resolves the critical technical challenges of the network running agent—from intent understanding to action execution. Through core mechanisms such as intelligent translation, template-driven orchestration, and cross-domain coordination, complex network tasks are automatically orchestrated and reliably executed, providing a robust technical foundation for autonomous network operations.

3.2.3 Agent Coordination Mechanism

The agent collaboration mechanism is a critical component of the intelligent network running agent technology framework, governing interactions and synergy across different architectural levels. As network intelligence evolves, the rapid growth in agent volume and diversity has made efficient, reliable coordination a decisive factor for system stability. By establishing standardized coordination protocols and interaction modes, this mechanism enables comprehensive capabilities ranging from external agent access to internal multi-agent synergy.

The core challenge of agent coordination lies in resolving the interoperability of heterogeneous agents, which typically possess distinct functional features, interaction protocols, and execution modes. The key to designing this mechanism is achieving effective orchestration while maintaining agent autonomy. Accordingly, the collaboration framework employs a hierarchical architecture: the A2A protocol facilitates peer-to-peer coordination, while NetMCP enables deep integration between agents and network atomic capabilities.

External Agent Access and Coordination

External agent access highlights the open ecosystem architecture of the intelligent network running agent, primarily through two core collaboration modes. One is user-oriented agent proxy interaction. The central agent within the network does not interact with end users directly. Instead, it engages through user-side agent proxies such as Lingxi, Doubao, or DeepSeek. This allows users to express service requirements through their preferred interfaces. These proxies then collaborate with the network's central agent through the A2A protocol to facilitate the intelligent acquisition of network services.

The other is embodied agent and intelligent device integration. This mode focuses on the access and coordination of embodied agents and autonomous devices, including UAVs, intelligent vehicles, and industrial robotics. These agents act as both consumers of network services and providers of network capabilities. By establishing collaboration via the standardized A2A protocol, they enable dynamic resource application, intelligent scheduling, and the collaborative optimization of network resources.

This hierarchical agent collaboration model ensures a seamless user experience while fostering the efficient openness of network capabilities. It effectively extends the service boundaries and application scenarios for network running agents.

Internal Agent Coordination Mechanism

Internal agent coordination governs the coordination and synergy between the central agent and the domain execution agents. This relationship is defined by a rigorous hierarchical architecture. As the core controller, the central agent is responsible for overall planning and orchestration, including task decomposition, resource scheduling, progress synchronization, and result aggregation. Within their respective domains, execution agents are tasked with instruction reception, downward translation, task execution, status feedback, and exception handling.

Coordination Level	Central Agent Responsibility	Execution Agent Responsibility
Task Management	Decompose complex tasks into domain-specific subtasks.	Instruction understanding and domain-specific workflow orchestration.
Resource Coordination	Unified scheduling and orchestration of capabilities across domains.	Execute intra-domain operations and perform cross-domain capability invocations.
Status Control	Global progress monitoring and holistic exception management.	Real-time status feedback and granular exception reporting.

Table 2 Responsibilities of the central agent and execution agent

This coordination mechanism ensures the efficient execution and reliable coordination of cross-domain network operation tasks through standardized message buses and coordination protocols. In complex E2E service scenarios, the central agent maintains global situational awareness, orchestrating the collective efforts of domain-specific execution agents. This creates a seamless, closed-loop system that transforms user intents into precise, verified network operations.

Network Service Capability Enhancement of RAN4Agent

In coordination with embodied agents (such as industrial robots and intelligent connected vehicles), the network agent provides general-purpose computing services:

- **Proactive Perception of Service Characteristics:** The RAN agent proactively perceives and predicts the service traffic characteristics (such as the token generation interval of AI inference and the size of burst data flows) of the peer AI agent through coordination protocols, and dynamically adjusts the transmission scheduling rhythm (DRX/scheduling priority) of the air interface based on the sensing results, achieving a perfect match between network resources and service rhythm.
- **Real-Time Offloading of Terminal-Network Computing Power:** A coordination process for computing task offloading is established. When the computing power of a terminal is limited, the RAN agent can schedule edge computing resources, take

over real-time inference tasks of AI models, and ensure microsecond-level latency of computing result backhaul, implementing integrated coordination services of "connection+computing."

4 NetMCP Design

4.1 From MCP to NetMCP

4.1.1 MCP Background

As an AI-native protocol, MCP standardizes the interaction between AI and external systems, delivering significant values by simplifying tool integration, enhancing AI flexibility, and fostering a robust AI ecosystem.

Prior to the introduction of MCP, agent tool invocation relied primarily on the function calling technology. Under this paradigm, the LLM invokes external tools through vendor-defined interfaces. These interfaces are incompatible, complicating cross-platform development and resulting in M x N complexity of tool invocation. Furthermore, the function calling technology lacks native identity authentication and permission control, undermining the security of tool invocation. To solve these challenges, MCP defines a unified model context structure and a standardized tool invocation interface, reducing the complexity of tool invocation from M x N to M + N. In addition, MCP enables robust mechanisms such as identity authentication and token authorization at the transport layer, significantly hardening the security of tool invocation.

MCP utilizes the host-client-server architecture, encompassing basic protocols, server/client features, and practical tools. The basic protocols define the JSON-RPC message types, transmission mechanism, connection lifecycle management that MCP must support, and optional authorization frameworks for MCP implementation. The server/client features specify the recommended capability interfaces; for the server, these features include tools, resources, and prompts, while for the client, they include the roots, sampling, and elicitation. Besides, utilities provide other general capabilities for client/server, such as parameter completion, logging, and pagination.

As a critical bridge connecting agents to external environments, MCP has attracted wide attention from enterprises and developers, becoming a cornerstone of a thriving community ecosystem. On one hand, leading AI companies such as OpenAI, Google, and Alibaba have successively integrated compatibility for MCP into their platforms. On the other hand, the global developer community has actively contributed various MCP tools based on open-source platforms such as Hugging Face and GitHub, covering scenarios such as intelligent office, software development, and IoT applications. Furthermore, the emergence of commercial hosting and distribution platforms has catalyzed a new "Tools-as-a-Service" (TaaS) business model.

4.1.2 NetMCP Design Principles

MCP is primarily designed for interactions between agents and external environments in general scenarios. However, it is not natively optimized for the running mechanisms of mobile networks, particularly regarding their distributed architecture, high reliability, and high trustworthy requirements. Targeting to adapt the network-native distributed features and runnnig mechanisms , NetMCP reconstructs MCP for telecommunications and builds a bridge connecting the intelligent ecosystem with communications networks. Specifically, the core design principles of NetMCP focus on high compatibility, high controllability, high availability, and high real-time performance.

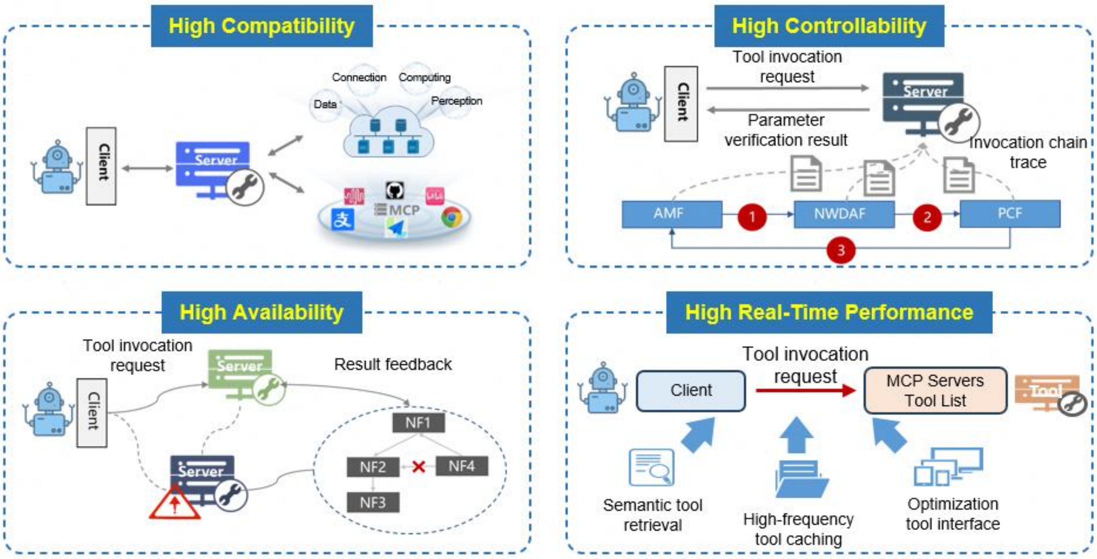


Figure 4 NetMCP protocol design principles

High Compatibility: NetMCP's compatibility is manifested in three dimensions. First, NetMCP needs to be compatible with existing communication network protocols and operational mechanisms to ensure a seamless and cost-effective evolution of the network architecture. On a protocol level, NetMCP entails compatibility with Service-Based Interfaces (SBIs); regarding mechanisms, NetMCP supports service registration/discovery and access authentication. Second, NetMCP needs to align with the third-party MCP ecosystem to foster mutual growth and prosperity. This includes supporting the registration and invocation of MCP tools, as well as ensuring seamless interconnection with agents within the MCP ecosystem. Third, to simplify the interactions between agents and networks, NetMCP emphasizes consistency between traditional API services and emerging agent services through aggregation and orchestration capabilities. By coordinating with the network agent, NetMCP orchestrates and aggregates atomic network capabilities, encapsulating them into one-stop service capabilities for open access. If traditional APIs are required, NetMCP facilitates the mapping between network APIs and tools, significantly reducing the complexity of service invocation by applications.

High Controllability: NetMCP must ensure that the tool invocation parameters are both trustworthy and controllable. To maintain high network stability, NetMCP enables parameter verification, ensuring that the tool-invocation-related requests remain secure and controllable. In addition, the tool execution process must be observable and traceable. Accordingly, NetMCP must be able to trace and monitor the NF invocation chain during tool execution, delivering fine-grained results. This enhances the overall maintainability and optimizability of the system.

High Availability: NetMCP supports distributed load balancing to mitigate the processing strain of high-concurrency requests, eliminate single points of failure, and enable seamless horizontal scaling. This includes balancing loads between agents and servers, as well as between servers and their respective tools. Furthermore, NetMCP incorporates distributed disaster recovery across multiple scenarios to prevent system deterioration triggered by tool execution errors or device faults—such as faults within agents, servers, networks, or links.

High Efficiency: NetMCP optimizes the connectivity and invocation between agents and tools through both transport protocols and model contexts. On one hand, NetMCP supports QUIC at the transport layer to minimize handshake round-trip times (RTT) and enable optimal congestion control. On the other hand, NetMCP uses mechanisms such as semantic tool retrieval and high-frequency tool caching to improve discovery accuracy and minimize the context length that needs to be processed by the LLM. In addition, NetMCP optimizes tool interfaces to reduce the volume of parameters requiring inference and analysis by the LLM.

4.2 Core Architecture and Interaction Model

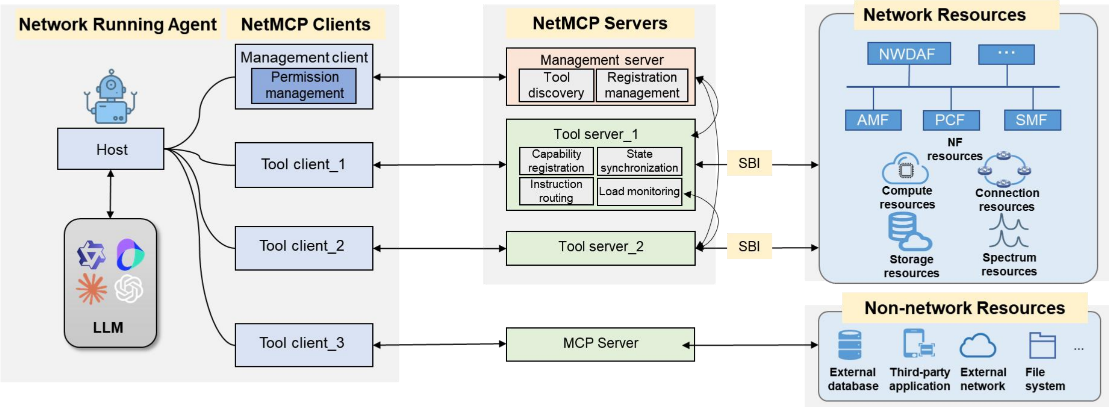


Figure 5 NetMCP architecture and interaction model

To maintain full compatibility with the MCP ecosystem, NetMCP adopts a client-server architecture similar to that of MCP. Within its protocol stack, NetMCP supports multiple transport layer options, including HTTP/S, Stdio, and QUIC. The data exchange format is standardized on JSON, ensuring smooth interoperability and meeting the fundamental compatibility requirements of the MCP ecosystem.

NetMCP consists of two types of servers: the tool servers and management servers.

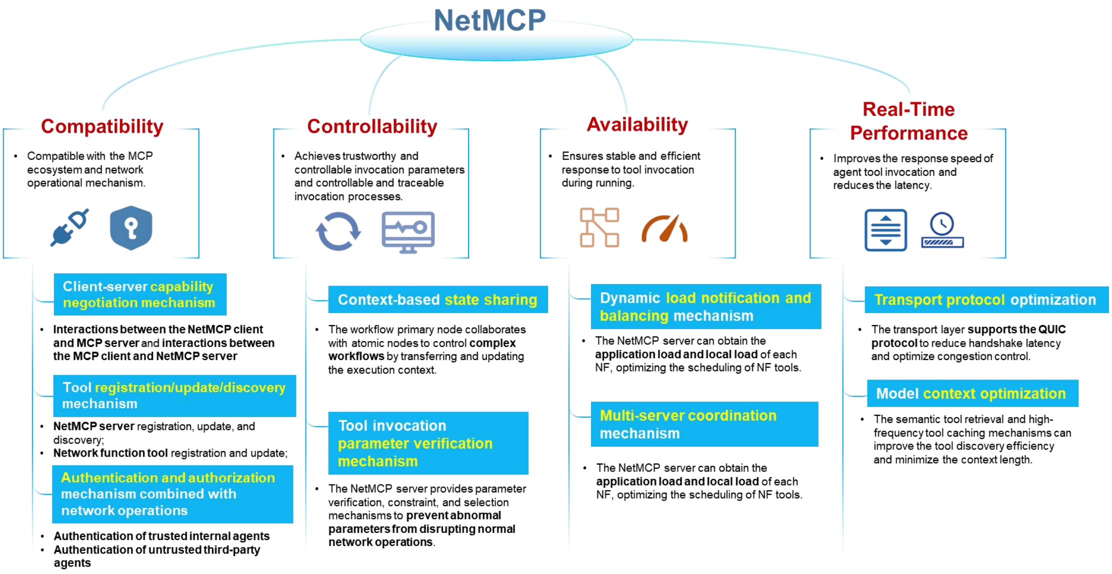
Tool Servers register, discover, and invoke NF tools and interact with NFs through SBIs. To ensure effective load balancing and process control within distributed environments, the tool servers support mechanisms such as instruction routing and state synchronization among each other. In terms of compatibility, it incorporates the basic capabilities of the MCP server—including tools, resources, and prompts—while maintaining full compatibility with the MCP client to facilitate the accessibility of network capabilities to third parties.

Management Servers register, discover, and authenticate the tool servers to dynamically manage network capabilities. As specialized communication nodes introduced by NetMCP, the manager servers can directly interact with the tool servers and provide services for network running agents.

In addition, NetMCP enables efficient management of heterogeneous resources.

Heterogeneous Resources are categorized as any operable objects on the server, encompassing both network and non-network assets. Network resources mainly refer to NFs, computing power, storage, connections, and spectrum, while non-network resources consist of third-party databases, external networks, and external file systems or applications.

4.3 Key Mechanisms



- Compatible with the MCP ecosystem and network operational mechanism.

- Achieves trustworthy and controllable invocation parameters and controllable and traceable invocation processes.

- Ensures stable and efficient response to tool invocation during running.

- Improves the response speed of agent tool invocation and reduces the latency.

Client-server capability negotiation mechanism

- **Interactions between the NetMCP client and MCP server** and **interactions between the MCP client and NetMCP server**

Tool registration/update/discovery mechanism

- **NetMCP server** registration, update, and discovery;
- **Network function tool** registration and update;

Authentication and authorization mechanism combined with network operations

- **Authentication of trusted internal agents**
- **Authentication of untrusted third-party agents**

Context-based state sharing

- The workflow primary node collaborates with atomic nodes to control **complex workflows** by transferring and updating the execution context.

Tool invocation parameter verification mechanism

- The NetMCP server provides parameter verification, constraint, and selection mechanisms to **prevent abnormal parameters from disrupting normal network operations**.

Dynamic load notification and balancing mechanism

- The NetMCP server can obtain the **application load and local load** of each NF, optimizing the scheduling of NF tools.

Multi-server coordination mechanism

- The NetMCP server can obtain the **application load and local load** of each NF, optimizing the scheduling of NF tools.

Transport protocol optimization

- The transport layer **supports the QUIC protocol** to reduce handshake latency and optimize congestion control.

Model context optimization

- The semantic tool retrieval and high-frequency tool caching mechanisms can improve the tool discovery efficiency and minimize the context length.

Figure 6 NetMCP technical system

The NetMCP technical system is built upon its four enhanced features. Regarding compatibility, NetMCP introduces an enhanced capability negotiation framework, a dynamic tool management mechanism, and an authentication and authorization mechanism. To address controllability, the system implements a state-sharing mechanism based on the execution context alongside tool invocation parameter verification. In terms of availability, NetMCP proposes a dynamic load notification and balancing mechanism as well as a multi-server coordination framework. Finally, to enhance real-time performance, the system defines specific transport protocol and model context optimization policies.

4.3.1 NetMCP Capability Negotiation

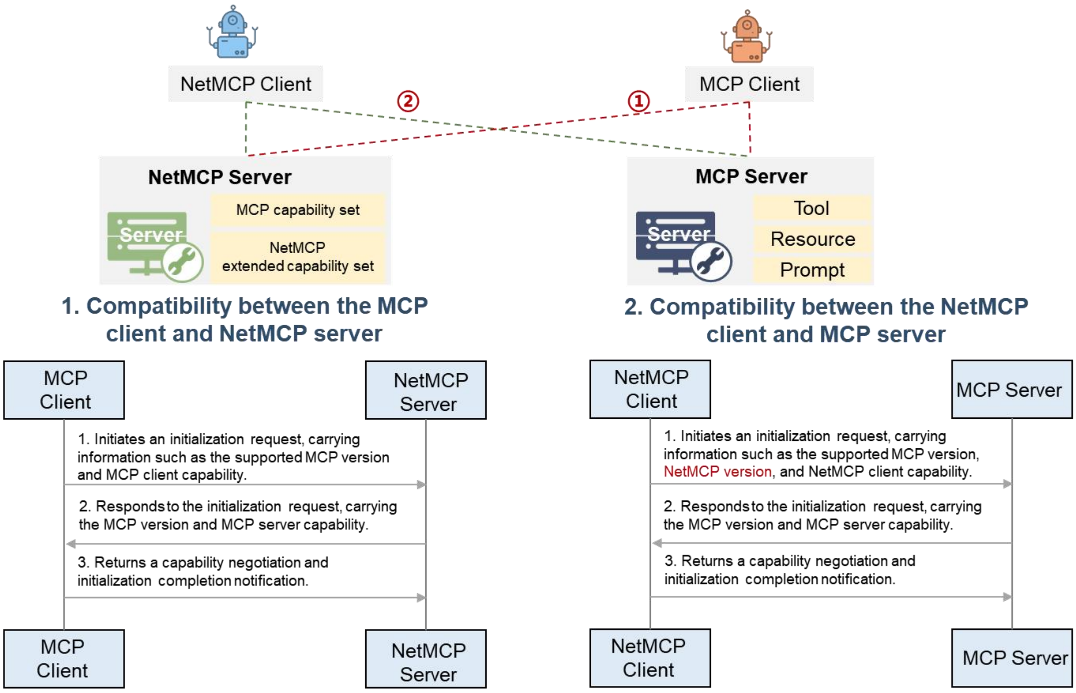
NetMCP adopts a capability negotiation mechanism compatible with MCP. The initialization request can carry both the NetMCP and MCP protocol version numbers. Consequently, the NetMCP server identifies the client type by verifying whether the initialization request carries the supported NetMCP protocol version number. Similarly, the NetMCP client determines the server type based on whether the server's response carries the NetMCP protocol version number.

The capability negotiation workflow between the NetMCP client and the MCP server is as follows:

Step 1: The NetMCP client sends an initialization request to the MCP server, carrying information such as the supported MCP version, NetMCP version, and NetMCP client capability.

Step 2: The MCP server responds with information such as the MCP version and MCP server capability, while ignoring the NetMCP version number carried in the request.

Step 3: Upon receiving the response, the NetMCP client determines the server as an MCP server and confirms that the MCP protocol version is supported; it then returns a capability negotiation and initialization completion notification.



1. Compatibility between the MCP client and NetMCP server

1. Initiates an initialization request, carrying information such as the supported MCP version and MCP client capability.

2. Responds to the initialization request, carrying the MCP version and MCP server capability.

3. Returns a capability negotiation and initialization completion notification.

2. Compatibility between the NetMCP client and MCP server

1. Initiates an initialization request, carrying information such as the supported MCP version, NetMCP version, and NetMCP client capability.

2. Responds to the initialization request, carrying the MCP version and MCP server capability.

3. Returns a capability negotiation and initialization completion notification.

4.3.2 Dynamic Tool Management

Dynamic tool management covers two scenarios. In the first scenario, tool servers can register and update their capabilities with the management server, enabling flexible and dynamic expansion of network capabilities. Meanwhile, agents can discover these capabilities through the management server and dynamically manage their connections to the tool servers.

More specifically, during registration or updates, tool servers can submit their metadata, such as vendor, maximum load, and service area, along with their capability descriptions to the management server based on local configurations. These capability descriptions provide a concise natural-language summary of all functions the tool servers offer, such as perception-related tools or connection management tools. During capability discovery, agents can obtain the capability descriptions of all registered tool servers within their service areas from the management server. They then select the tool servers relevant to the task and establish connections with the tool servers based on task-intent understanding and reasoning. Additionally, during capability registration and discovery, the management server and tool servers can provide authentication mechanisms to ensure that only authorized tool servers can register with the management server, and only authorized agents can invoke the tool servers.

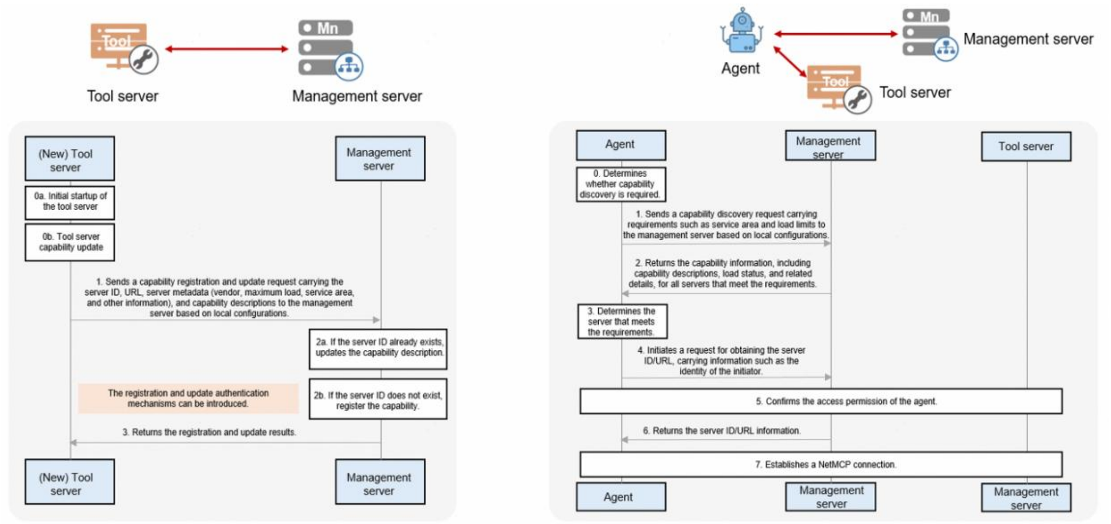

Tool
Mn
Tool server
Management server
(New) Tool server
Management server
0a. Initial startup of the tool server
0b. Tool server capability update
1. Sends a capability registration and update request carrying the server ID, URL, server metadata (vendor, maximum load, service area, and other information), and capability descriptions to the management server based on local configurations.
2a. If the server ID already exists, updates the capability description.
The registration and update authentication mechanisms can be introduced.
2b. If the server ID does not exist, register the capability.
3. Returns the registration and update results.
(New) Tool server
Management server
Agent
Mn
Management server
Tool
Tool server
Agent
Management server
Tool server
0. Determines whether capability discovery is required.
1. Sends a capability discovery request carrying requirements such as service area and load limits to the management server based on local configurations.
2. Returns the capability information, including capability descriptions, load status, and related details, for all servers that meet the requirements.
3. Determines the server that meets the requirements.
4. Initiates a request for obtaining the server ID/URL, carrying information such as the identity of the initiator.
5. Confirms the access permission of the agent.
6. Returns the server ID/URL information.
7. Establishes a NetMCP connection.
Agent
Management server
Management server

Figure 8 NetMCP-based dynamic tool management

In the second scenario, NFs are abstracted as tool capabilities for dynamic management. An NF can publish its capabilities and perform tool registration, updates, or deletion either through the NRF or by interacting directly with tool servers, enabling finer-grained expansion of network capabilities. When initiating tool registration with a tool server, the NF can include tool labels and a summary of its own capabilities to help the tool server determine, through mechanisms such as semantic matching, whether the new tool needs to be registered locally.

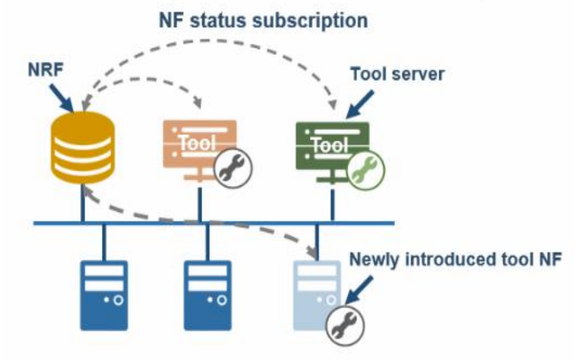


Figure 9 NRF-based dynamic tool management

4.3.3 Tool Authentication Mechanism

Authentication is required when an agent invokes a network capability through NetMCP. During authentication, the management server serves as a centralized authorization server.

Specifically, two situations are involved. For a trusted agent that already holds client credentials, it can directly request a token from the management server by providing its client credentials, the resource identifier, and other required information. The management server verifies the agent's access rights to the requested resource based on these credentials. If the verification succeeds, it returns the access token of the tool server to complete authorization.

For an untrusted third-party agent that needs to access user-related data, it can first obtain client credentials by registering with the management server. The user is then guided through a portal to perform authorization for this agent on the tool management server and obtain an authorization code. Using this authorization code together with the resource name and other required information, the agent can request an access token from the management server to complete authorization.

4.3.4 State Sharing Mechanism

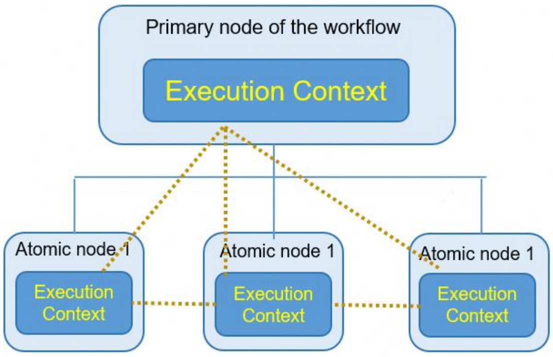


Figure 10 State sharing mechanism

Execution context is a core object that persists throughout the lifecycle of a workflow instance. It can be passed to every atomic capability in the workflow, providing an isolated, standardized, and feature-rich runtime environment that fully decouples atomic capabilities from one another.

High decoupling: Each atomic capability focuses solely on its own logic without managing how other capabilities are invoked.

State keeping: supports complex, long-running workflows that require information to be passed across steps.

Flexible execution: natively supports execution modes essential for task scenarios such as asynchronous operations, latency, and scheduling.

The core components of the execution context include the state manager, message queue, and scheduler. The state manager provides K-V storage to support data sharing and transfer across steps. The message queue enables task decoupling and asynchronous processing, including fan-out patterns. The scheduler handles timing-related logic such as scheduling, latency, and retries.

4.3.5 Parameter Verification Mechanism

NetMCP servers can provide parameter verification, constraint, and selection mechanisms to prevent abnormal parameters from disrupting normal network operations. Verification may rely on predefined rules such as value ranges or inter-parameter constraints, or on model-based reasoning, for example by invoking expert models dedicated to parameter verification. For cases where a clear determination cannot be made, the system may request human intervention or treat the parameters as invalid.

4.3.6 Load Optimization

- Multi-server Collaboration

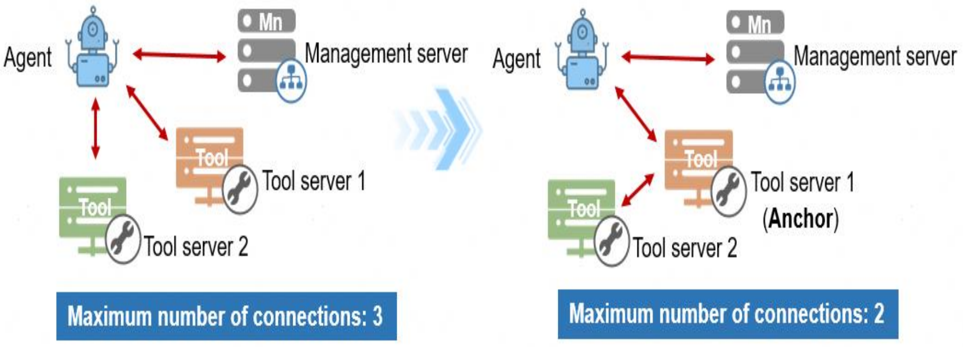


Figure 11 Collaboration among multiple NetMCP servers

Based on the server-to-server interaction mechanism provided by NetMCP, instruction routing can be established between agents and multiple servers, reducing connection pressure on the agent side and enabling system-wide load balancing and efficiency optimization. At the same time, designating an anchor server as the central point for information aggregation enables coordinated capability execution across multiple servers, reducing the agents' instruction exchanges and polling frequency.

- Dynamic Load Notification and Balancing Mechanism

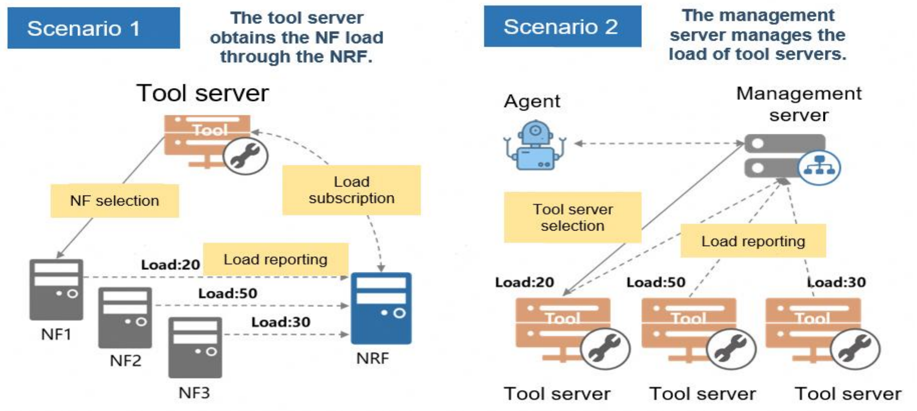


Figure 12 NetMCP dynamic load notification and balancing mechanism

NetMCP tool servers can leverage existing network operation mechanisms—such as the NRF—to obtain NF load information. They can also report their own load to the management server. Based on dynamic load conditions and local policies, the management server can then optimize tool server access, preventing overload and achieving system-wide load balancing.

4.3.7 Transmission Protocol Optimization

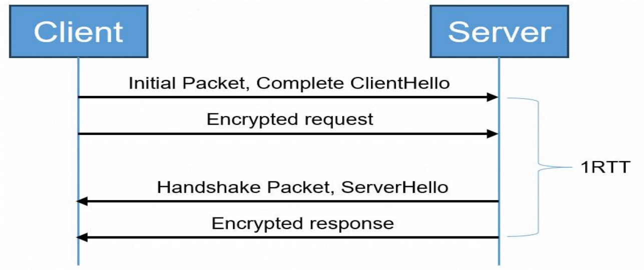

Client
Server
Initial Packet, Complete ClientHello
Encrypted request
1RTT
Handshake Packet, ServerHello
Encrypted response

Figure 13 QUIC transmission protocol

The latest MCP protocol is implemented on top of Streamable HTTP, using TCP at the transport layer. To reduce latency, NetMCP can support HTTP/3, which uses QUIC at the transport layer, lowering handshake latency and improving congestion control to deliver more stable and efficient communication.

4.3.8 Context Optimization

MCP provides only the capability of obtaining the tool list. When the number of tools on the MCP server increases and the interface definition becomes complex, model contexts bloat, increasing hallucination risks (for example, a tool that does not exist is called) and reducing inference efficiency. To address this issue, NetMCP samples multiple model context optimization strategies.

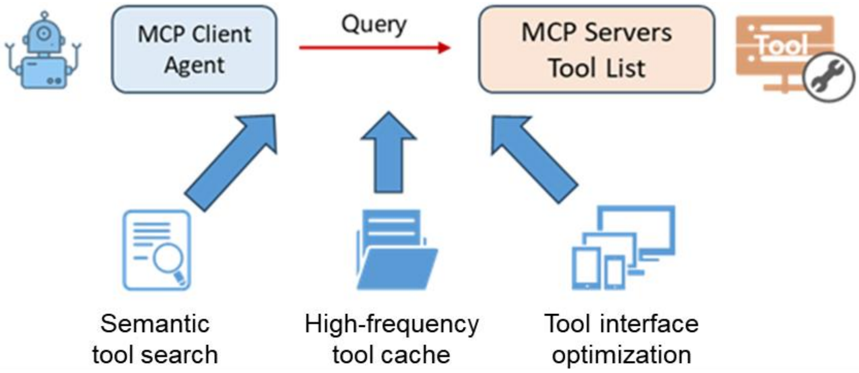


Figure 14 Model context optimization mechanism of NetMCP

Semantic tool search: The NetMCP client can send a tool discovery request containing the tool requirement description to the NetMCP server. The NetMCP server uses the semantic search module to filter a list of local tools that best meet the tool requirement description and sends the list to the client. This mechanism mitigates context bloat caused by task-independent tools.

Cache mechanism for high-frequency tools: The NetMCP server can locally maintain the frequency at which the client accesses each tool. When the client initiates the latest tool list obtaining or tool discovery request, the server sorts tools based on their access frequency and returns high-frequency tools first, improving the hit rate of a single tool discovery.

Tool interface optimization: Tool interfaces are simplified to avoid complex interface parameters. Parameters with fixed values or parameters that can be inferred based on rules can be automatically supplemented. This mechanism can reduce agents' calculation burden of determining a tool calling parameter and improve tool calling efficiency.

Tool self-optimization mechanism: The NetMCP server can learn the habits of users (agents) in calling tools, aggregate and generate new tools, and register and open the tools to simplify tool calling by agents.

5 Innovative Application Scenarios and Solutions

5.1 Personalized Intelligent Network Services

Development Vision

Conventional network services are mainly based on standardized package plans, which cannot meet users' increasing personalized requirements. By deeply understanding user intents, network running agents implement on-demand customization and dynamic allocation of network capabilities, and transform network services from the unified mode to the personalized mode. This is not only an upgrade of the service mode, but also a reshaping of the network value chain. It guarantees dedicated network experience for each user and makes the network a truly intelligent partner that understands and serves users.

Key Capabilities

- **Intent perception and understanding:** Users' explicit requirements and implicit expectations can be accurately translated based on natural language processing technologies.
- **Scenario-based service orchestration:** The optimal service solution can be intelligently orchestrated based on multi-dimensional factors, such as time, location, and service characteristics.
- **Dynamic resource scheduling:** Network status can be perceived in real time and resource configuration can be dynamically adjusted to ensure consistent service quality.
- **Closed-loop optimization mechanism:** User preferences can be learned and service policies can be optimized continuously to improve user experience.

Typical Practices

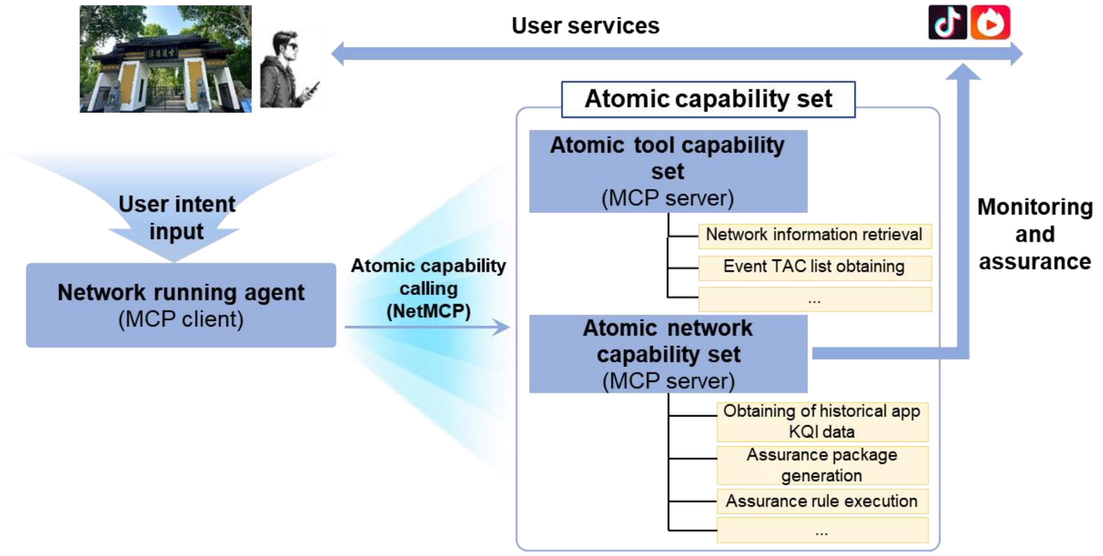
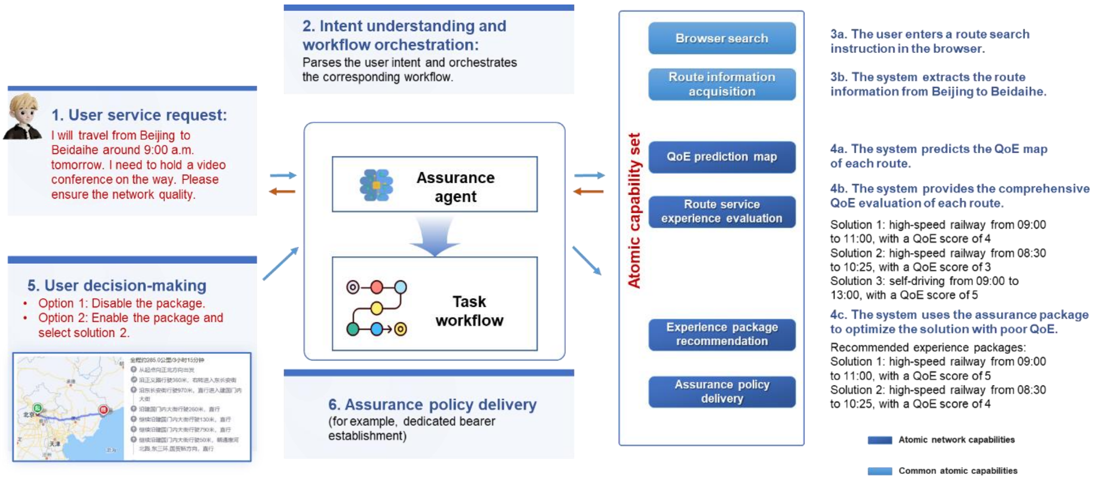


Figure 15 Major event assurance

Scenario 1: Intelligent Network Assurance for Major Events

When a user plans to perform HD live streaming at a large-scale activity such as a concert or a sports event, the user only needs to express a requirement in a natural language as follows: "I want to perform TikTok live streaming at a concert." The network running agent can automatically complete complex processes such as network quality evaluation, assurance solution generation, and resource reservation configuration, and dynamically adjust network parameters during the activity to ensure the ultimate experience throughout the live streaming.


Assurance agent
Task workflow
Atomic capability set
Browser search
Route information acquisition
QoE prediction map
Route service experience evaluation
Experience package recommendation
Assurance policy delivery
Atomic network capabilities
Common atomic capabilities

1. User service request:
I will travel from Beijing to Beidaihe around 9:00 a.m. tomorrow. I need to hold a video conference on the way. Please ensure the network quality.

2. Intent understanding and workflow orchestration:
Parses the user intent and orchestrates the corresponding workflow.

3a. The user enters a route search instruction in the browser.

3b. The system extracts the route information from Beijing to Beidaihe.

4a. The system predicts the QoE map of each route.

4b. The system provides the comprehensive QoE evaluation of each route.

Solution 1: high-speed railway from 09:00 to 11:00, with a QoE score of 4
Solution 2: high-speed railway from 08:30 to 10:25, with a QoE score of 3
Solution 3: self-driving from 09:00 to 13:00, with a QoE score of 5

4c. The system uses the assurance package to optimize the solution with poor QoE.

Recommended experience packages:
Solution 1: high-speed railway from 09:00 to 11:00, with a QoE score of 5
Solution 2: high-speed railway from 08:30 to 10:25, with a QoE score of 4

5. User decision-making
- Option 1: Disable the package.
- Option 2: Enable the package and select solution 2.

6. Assurance policy delivery
(for example, dedicated bearer establishment)

Figure 16 Network experience assurance for travel routes

Scenario 2: Network Experience Assurance for Travel Routes

Scenario Description

Business travelers require network continuity during an inter-city trip (e.g., from the airport to the destination) to complete key services such as video conferences and file transfers. Conventional network services are unable to cope with network quality fluctuations and route switching issues in mobile scenarios.

Step 1: Intelligent intent identification

A user enters the information "I will fly to Shanghai tomorrow afternoon and have a video conference in the evening. I need network assurance throughout the journey." on the terminal. The terminal agent then uses the A2A protocol to send the requirement to the network's central agent. The central agent parses the travel time, route nodes, service types, and service priorities, and understands that the user needs to obtain video conference–level network quality assurance in the mobile scenario.

Step 2: Full-route network pre-evaluation

The central agent schedules the execution agent in each domain for coordinated analysis. The access network agent calls network capabilities through the NetMCP protocol to obtain real-time coverage data of the departure place, destination, and intermediate sections, and identifies weak coverage areas. The core network agent predicts indicators such as the bandwidth and latency required for video conferences based on historical data. Then, the system generates a full-route network quality map, with high-risk points marked (such as in-flight Wi-Fi fluctuations and weak coverage areas on the ground).

Step 3: Generation and execution of an assurance solution

The central agent generates a personalized assurance solution. The execution agent in each domain call NF capabilities through the NetMCP protocol to complete the configuration. This includes reserving resources for video conferences and configuring QoS policies, presetting a smooth 5G-to-4G handover mechanism in weak coverage areas, and configuring priority assurance for key services. Related NF parameter optimization and policies are delivered before the user travels.

Step 4: Real-time monitoring and dynamic optimization

During the travel, the network agent continuously collects NF-reported real-time data such as signal strength, rate, and delay through the NetMCP, and monitors the service status. When an exception is detected, the following emergency measures are automatically triggered to ensure service continuity: switching the service to the standby network in weak coverage areas, dynamically adjusting the QoS priority in case of congestion, and proactively sending network status notifications.

Step 5: Experience feedback and optimization

After the trip ends, the central agent collects the experience data of the entire trip to analyze the assurance effect. In addition, the agent accumulates successful experience and improvement points in the knowledge base to optimize the assurance policies for similar scenarios in the future. This way, a closed-loop of continuous improvement is formed.

Scenario 3: Proactive rights assurance based on connection status analysis

For high-value users or specific service scenarios (such as HD live streaming and instant video conference), the access network agent leverages its real-time perception capability to upgrade the service mode from passive response to proactive assurance. The agent monitors the wireless connection statuses (such as the CQI and throughput margin) of users in real time and performs intelligent analysis based on user profile tags (such as business traveler and gamer). When detecting that a user is in an area with good coverage but faces potential resource competition risks, the agent proactively recommends the user to subscribe to the deterministic assurance package or acceleration service. After the user confirms the recommendation, the agent calls atomic capabilities through the NetMCP to immediately lock air-interface RB resources and scheduling priority, providing deterministic bandwidth and latency assurance and delivering a network service experience that is exactly as expected.

Scenario 4: Proactive assurance during e-commerce promotions

To cope with the instantaneous surge in concurrent transactions and high service continuity requirements, the network running agent can automatically enable the intelligent data collection enhancement mechanism. In this case, the agent can dynamically increase the collection frequency and granularity of core service interfaces such as the payment gateway and order center, and synchronize multi-dimensional service flow and network quality data in real time. Based on high-quality real-time data input, the agent dynamically constructs service-network experience profiles, accurately maps the relationship between service phases and network performance, and visualizes and locates experience bottlenecks. Once the payment latency increases or the quality of key links fluctuates, the agent immediately triggers end-to-end accurate demarcation and root cause analysis, and coordinates with the resource scheduling module to implement dynamic optimization. In this case, user experience is restored without affecting services, and an integrated closed-loop assurance mechanism is formed, covering data collection, analysis, diagnosis, and optimization, so as to ensure smooth and stable transactions during promotions.

5.2 Socialized Intelligent Care Services

Development Vision

The network is not only a channel for information transmission, but also an intelligent platform that safeguards social well-being. The network running agent extends network capabilities to social livelihood fields such as education, health, and security through in-depth coordination with terminals and applications, building an intelligent protection system that covers the entire life-cycle. This transformation will drive communication operators to upgrade from infrastructure providers to social value creators, making technological innovation truly serve the holistic development of human society.

Key Capabilities

- Multi-dimensional perception and convergence: Data from multiple sources, such as network, terminals, and applications, are integrated to build a multi-dimensional perception system.
- Intelligent decision-making engine: The inference capabilities of LLMs are used to implement intelligent judgment and precise intervention in complex scenarios.
- Privacy protection mechanism: Necessary monitoring and protection functions are implemented while ensuring user privacy.
- Human-machine coordinated interaction: Natural and friendly interaction modes are provided to make technical services more considerate.

Typical Practices

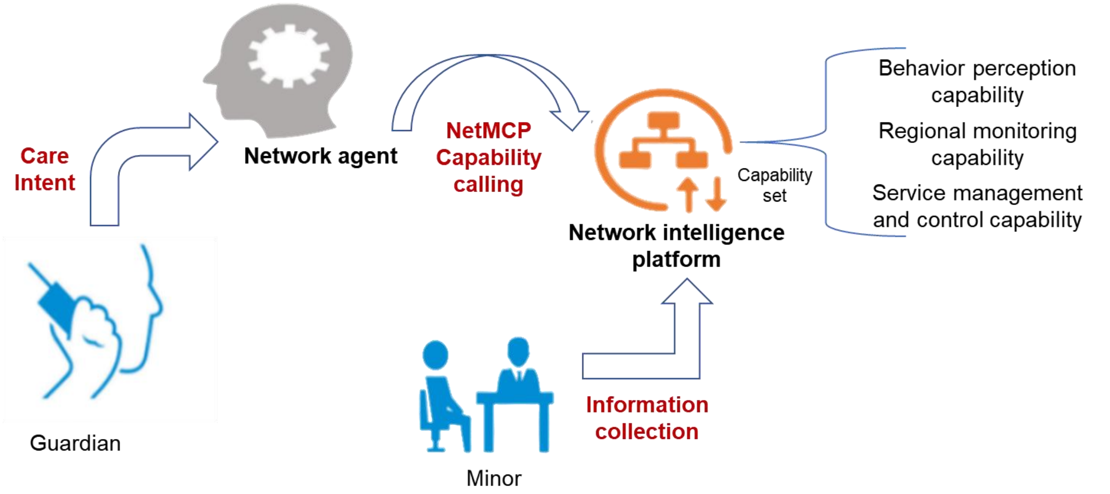

Care Intent
Network agent
NetMCP Capability calling
Capability set
Network intelligence platform
Behavior perception capability
Regional monitoring capability
Service management and control capability
Guardian
Minor
Information collection

Figure 17 Care for minors

Scenario 1: Growth protection for minors

The network running agent can intelligently generate personalized protection solutions based on the education concepts set by parents. The system not only implements basic time control and content filtering, but also identifies potential risks through intelligent analysis, so as to guide healthy network usage habits in a timely manner. For example, during holidays, the agent can help formulate a "learning+entertainment" balance plan to ensure the learning effect of a minor without affecting the necessary relaxation.

Scenario 2: Intelligent accompanying for the elderly

To address the difficulties for the elderly to use digital devices, the network running agent provides simplified and user-friendly service interfaces to help them conveniently use network services through voice interactions. In addition, the agent provides considerate functions such as health monitoring and emergency help, enabling technologies to truly benefit every group.

5.3 Industrial Collaboration Intelligent Services

Development Vision

In the digital economy era, a more intelligent and efficient collaboration mechanism is required for industry development. Network running agents facilitate the establishment an open capability platform and a standard collaboration framework, which in turn promotes the in-depth integration of the network and agents in various industries. This integration will create synergy beyond the sum of the individual agents. This will give rise to a brand-new industry ecosystem and business model, making the network the intelligent foundation that connects everything and empowers all industries, thus accelerating the digital transformation and intelligent upgrade of the entire society.

Key Capabilities

- Open capability platform: Standardized network atomic capabilities are packaged as services, enabling easy invocation and integration by third-party agents.

- Cross-domain collaboration mechanism: Efficient collaboration and information sharing are supported among agents in different domains and at different levels.

- Ecosystem governance system: Robust authentication, authorization, and charging mechanisms are established to ensure the healthy development of the ecosystem.

- Value co-creation model: Flexible business models facilitate mutually beneficial development for all stakeholders in the industry chain.

Typical Practices

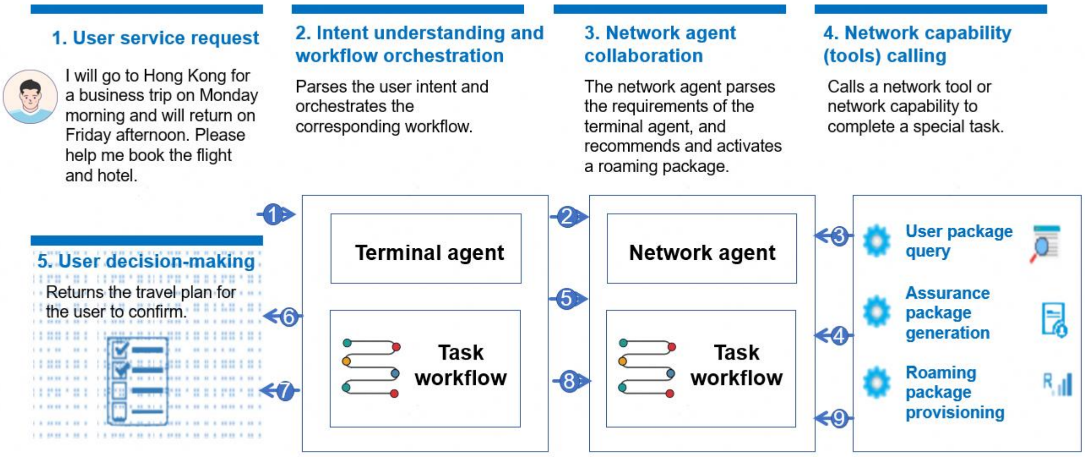

1. User service request
I will go to Hong Kong for a business trip on Monday morning and will return on Friday afternoon. Please help me book the flight and hotel.
2. Intent understanding and workflow orchestration
Parses the user intent and orchestrates the corresponding workflow.
3. Network agent collaboration
The network agent parses the requirements of the terminal agent, and recommends and activates a roaming package.
4. Network capability (tools) calling
Calls a network tool or network capability to complete a special task.
5. User decision-making
Returns the travel plan for the user to confirm.
Terminal agent
Task workflow
Network agent
Task workflow
User package query
Assurance package generation
Roaming package provisioning
1
2
3
4
5
6
7
8
9

Figure 18 Travel service based on multi-agent collaboration

Scenario 1: One-stop smart travel service

An OTT agent or terminal agent requests the network to start evaluating and analyzing key services and to optimize them. For example, the user only needs to express a simple intent of going to Hong Kong next week, and the terminal agent can collaborate with agents in multiple domains such as flight, hotel, and network to automatically complete full-process services such as air ticket booking, hotel arrangement, and international roaming provisioning. The network running agent not only ensures communication services, but also functions as a collaboration hub to ensure the seamless transition of various services.

Scenario 2: Collaborative upgrade of industrial digitalization

In vertical industries such as smart manufacturing and smart logistics, the network running agent can be deeply integrated with enterprise production systems and supply chain platforms. Through real-time data analysis and intelligent decision-making, production scheduling is optimized and operating costs are reduced, facilitating the digital transformation of traditional industries.

6 Industry Promotion Plan

GTI will collaborate with industry partners to promote the industrialization and ecosystem prosperity of network running agents. Our efforts will focus on four aspects: standardization, ecosystem construction, scenario exploration, and industry practices.

In terms of standardization, the CCSA and IMT-2030 Promotion Group promote the research on scenarios, architectures, protocols, and key technologies of network running agents. International organizations such as 3GPP and ITU promote the research and discussions on 6G-oriented network agents, including capability definitions, architecture solutions, key processes, and protocol design. At the same time, GTI and industry partners will follow up the research progress of agent-related protocols in IETF.

In terms of ecosystem construction, we have designed and developed the NetMCP prototype and SDK using a simulated live network environment. We will continuously improve the technical specifications and optimize the performance of NetMCP. We will also actively engage with open-source communities to foster a vibrant NetMCP ecosystem and unite the industry behind a common goal. By establishing a collaboration mechanism with partners, operators' self-built capabilities and third-party partners' capabilities are aggregated to build an open and co-constructed agent ecosystem.

In terms of scenario exploration, we will further explore the application potential of network running agents across various vertical industries, building upon existing use cases such as concert livestreaming support, care and protection services, and user travel experience. Our focus is on unlocking the value of multi-agent collaboration in complex service scenarios, driving the evolution of agent capabilities from standalone applications to end-to-end, collaborative scenarios. Aligning with the development trends of emerging technologies such as 5G private network and edge computing, we will proactively explore integration scenarios for network running agents and industrial agents. This broadens the scope of agent technology applications across fields including smart manufacturing, smart transportation, and smart healthcare.

In terms of industry practices, we will accelerate the transformation of the network running agents from technical verification to large-scale application. A new service model, leveraging intent understanding and intelligent decision-making, will be developed to explore differentiated business monetization strategies. Technical tests will be conducted in typical scenarios, and an effect evaluation system will be established to accumulate experience for subsequent large-scale promotion. A complete industry chain from technology R&D, standards formulation, to business implementation will be formed to promote the industrial development of network running agents.

7 Summary and Outlook

With the joint efforts of all parties in the industry, network running agents have made phased achievements in architecture design, technical breakthrough, standard formulation, and application exploration. The overall development has entered a key phase of promotion. At the same time, with the accelerated evolution of future network, the convergence of 6G and network running agents has become a strategic priority of the industry. The combination of the two is injecting core momentum into the transformation of network operation mode and reshaping of network architecture. This not only promotes the collaboration between AI and 6G technologies to achieve closed-loop intelligence at the service level, but also facilitates the upgrade and construction of new infrastructure, thereby spawning new industry, O&M, and service models. The development of network running agents and their application scenarios present an opportunity to promote network intelligence transformation. In the future, network running agents are expected to lay a solid network foundation for digital economy development.

In this context, GTI has compiled the White Paper on Network Agent and NetMCP Technology. This white paper defines the core concepts and technical characteristics of network running agents, proposes a comprehensive technical framework, develops interaction protocols tailored to network running mechanisms, details application practice cases and scenario planning, and elaborates on the evolutionary path through standardization, ecosystem construction, scenario exploration, and industry practices to guide industry development.

We would like to take this opportunity to invite industry partners and colleagues to join us in the research, development, and application of network running agents. By enriching the ecosystem, we can accelerate the in-depth integration and endogenous evolution of the network and AI, and help the network evolve from intelligent to smart, thereby continuously empowering high-quality social development.

Contributors

China Mobile Research Institute
China Academy of Information and Communications Technology
Huawei Technologies Co., Ltd.
ZTE Corporation
vivo Mobile Communication Co., Ltd.
Guangdong OPPO Mobile Telecommunications Corp., Ltd.
CICT Mobile Communication Technology Co., Ltd.
Nokia Shanghai Bell Co., Ltd.
Innomix Group Limited

Acronyms and Abbreviations

Abbreviation	Full Name
3GPP	The 3rd Generation Partnership Project
5G	5th Generation of Cellular Mobile Communications

Abbreviation	Full Name
6G	6th Generation of Cellular Mobile Communications
A2A	Agent2Agent
AI	Artificial Intelligence
CCSA	China Communications Standards Association
ETSI	European Telecommunications Standards Institute
IETF	Internet Engineering Task Force
ITU	International Telecommunication Union
LLM	Large Language Model
MCP	Model Context Protocol
QoE	Quality of Experience
QoS	Quality of Service
QUCI	Quick UDP Internet Connections
TCP	Transmission Control Protocol

References

[1] Anthropic. Model Context Protocol (MCP) Specification[EB/OL]. (2024-11). https://modelcontextprotocol.io/docs/getting-started/intro

[2] Google. Agent 2 Agent (A2A) Protocol Technical Documentation[EB/OL]. (2025-04). https://developers.google.com/

[3] 3GPP. Technical Report 22.870 (6G Requirements Including Agent Definition)[S]. 3GPP TR 22.870, 2025.