Draft new Recommendation ITU-T Y.3193 (ex Y.IMT2020-DIC)

Distributed intelligence collaboration in IMT-2020 and beyond

Summary

This Recommendation specifies distributed intelligence collaboration in IMT-2020 and beyond, in terms of general considerations, requirements, functional framework and procedure.

Distributed intelligence collaboration in IMT-2020 and beyond is expected to be of great value for network operators, service providers and end users, providing enhanced support of artificial intelligence (AI) technologies and increased resource utilization.

Keywords

AI agent; distributed intelligence collaboration; IMT-2020 and beyond; machine learning

Table of Contents

Page

1. Scope 3

2. References 3

3. Definitions 4
   3.1. Terms defined elsewhere 4
   3.2. Terms defined in this Recommendation 4

4. Abbreviations and acronyms 4

5. Conventions 5

6. General considerations on distributed intelligence collaboration in IMT-2020 and beyond 5
   6.1. Overview of distributed intelligence collaboration 5
   6.2. Enabling technologies for distributed intelligence collaboration in IMT-2020 and beyond 7

7. Requirements of distributed intelligence collaboration in IMT-2020 and beyond 8

8. Functional framework of distributed intelligence collaboration in IMT-2020 and beyond 9
   8.1. The role of AI agents in the functional framework 9
   8.2. Functional framework of distributed intelligence collaboration 10
   8.3. Reference points 11

9. Procedure of distributed intelligence collaboration in IMT-2020 and beyond 11

10. Security considerations 13

Appendix I 13

Use cases of distributed intelligence collaboration in IMT-2020 and beyond 13

Bibliography 15

Draft new Recommendation ITU-T Y.3193 (ex Y.IMT2020-DIC)

Distributed intelligence collaboration in IMT-2020 and beyond

1. Scope

This Recommendation specifies distributed intelligence collaboration in IMT-2020 and beyond.

This Recommendation addresses the following aspects of distributed intelligence collaboration in IMT-2020 and beyond:

- general considerations;
- requirements;
- functional framework;
- procedure.

Appendix I provides use cases of distributed intelligence collaboration in IMT-2020 and beyond.

2. References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published.

The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T Y.3100] Recommendation ITU-T Y.3100 (2017), *Terms and definitions for IMT-2020 network.*

[ITU-T Y.3101] Recommendation ITU-T Y.3101 (2018), *Requirements of the IMT-2020 network.*

[ITU-T Y.3102] Recommendation ITU-T Y.3102 (2018), *Framework of the IMT-2020 network.*

[ITU-T Y.3104] Recommendation ITU-T Y.3104 (2018), *Architecture of the IMT-2020 network.*

[ITU-T Y.3172] Recommendation ITU-T Y.3172 (2019), *Architectural framework for machine learning in future networks including IMT-2020.*

[ITU-T Y.3174] Recommendation ITU-T Y.3174 (2020), *Framework for data handling to enable machine learning in future networks including IMT-2020*

[ITU-T Y.3176] Recommendation ITU-T Y.3176 (2020), *Machine learning marketplace integration in future networks including IMT-2020*

[ITU-T Y.3179] Recommendation ITU-T Y.3179 (2021), *Architectural framework for machine learning model serving in future networks including IMT 2020.*

3. Definitions

3.1. Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 Artificial intelligence (AI) [b-ETSI GR ENI 004]: Computerized system that uses cognition to understand information and solve problems.

NOTE 1 – [b-ISO/IEC 2382-28] defines AI as "an interdisciplinary field, usually regarded as a branch of computer science, dealing with models and systems for the performance of functions generally associated with human intelligence, such as reasoning and learning".

NOTE 2 – In computer science AI research is defined as the study of "intelligent agents": any device that perceives its environment and takes actions to achieve its goals.

NOTE 3 – This includes pattern recognition, the application of machine learning and related techniques.

NOTE 4 – Artificial-intelligence is the whole idea and concept of machines being able to carry out tasks in a way that mimics human intelligence and would be considered "smart".

3.1.2 AI agent [b-ISO/IEC 22989]: Automated entity that senses and responds to its environment and takes actions to achieve its goals.

3.1.3 control plane [b-ITU-T Y.2011]: The set of functions that controls the operation of entities in the stratum or layer under consideration, plus the functions required to support this control.

3.1.4 data plane [b-ITU-T Y.2011]: The set of functions used to transfer data in the stratum or layer under consideration.

3.1.5 IMT-2020 [ITU-T Y.3100]: Systems, system components, and related technologies that provide far more enhanced capabilities than those described in [b-ITU-R M.1645].

NOTE – [b-ITU-R M.1645] defines the framework and overall objectives of the future development of IMT-2000 and systems beyond IMT-2000 for the radio access network.

3.1.6 machine learning (ML) [ITU-T Y.3172]: Processes that enable computational systems to understand data and gain knowledge from it without necessarily being explicitly programmed.

3.1.7 machine learning overlay [ITU-T Y.3172]: A loosely coupled deployment model of machine learning functionalities whose integration and management with network functions are standardized.

NOTE – A machine learning overlay aims to minimise interdependencies between machine learning functionalities and network functions using standard interfaces, allowing for parallel evolution of functionalities of the two.

3.1.8 network function [b-ITU-T Y.3100]: In the context of IMT-2020, a processing function in a network.

3.2. Terms defined in this Recommendation

None.

4. Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

A2A Agent-to-Agent

AF Application Function

AI Artificial Intelligence

AN Access Network

API Application Programming Interface

CN Core Network

LLM Large Language Model

MAS Multi-agent system

MCP Model Context Protocol

ML Machine Learning

NAS Non-Access Stratum

NF Network Function

NMS Network Management System

QoS Quality of Service

UE User Equipment

5. Conventions

In this Recommendation:

The keywords "is required to" indicate a requirement which must be strictly followed and from which no deviation is permitted, if conformance to this Recommendation is to be claimed.

The keywords "is recommended" indicate a requirement which is recommended but which is not absolutely required. Thus, this requirement need not be present to claim conformance.

The keywords "can optionally" indicate an optional requirement which is permissible, without implying any sense of being recommended. This term is not intended to imply that the vendor's implementation must provide the option, and the feature can be optionally enabled by the network operator/service provider. Rather, it means the vendor may optionally provide the feature and still claim conformance with this Recommendation.

6. General considerations on distributed intelligence collaboration in IMT-2020 and beyond

6.1. Overview of distributed intelligence collaboration

The distributed intelligent collaboration in IMT-2020 and beyond aims to achieve two objectives as follows.

Objective 1: the distributed intelligent collaboration aims to enhance the network intelligence capabilities, including enhancing the network's autonomous management capabilities, adaptive capabilities, and overall operational performance, ultimately realizing artificial intelligence (AI)-empowered networking. Specifically, this involves multiple network functions (NFs) jointly undertaking AI tasks (AI-enabled tasks, named for simplicity "AI tasks"), utilizing multi-task collaboration mechanisms to handle complex network intents [b-3GPP TS 28.312]. The AI agent technology enables the implementation of these network intelligence capabilities and perform the corresponding AI tasks.

NOTE 1 – An intent expresses the metrics that need to be achieved (the "what"), not the specific methods for how to achieve them [b-3GPP TS 28.312]. Complex network intents extend this concept to describe high-level objectives comprised of multiple, interdependent, and often conflicting goals. These intents are typically declared by a user or application to define a desired outcome, without specifying the implementation. For example, a business-level intent could be "ensure seamless quality of experience for premium video streaming subscribers during peak evening hours". And a supporting operational-level intent could be "dynamically reallocate

bandwidth and optimize routing paths within the access network to maintain the contracted service level for the streaming application without affecting other network users".

NOTE 2 – An example of joint undertaking of AI tasks by multiple network functions is as follows: to achieve the goal of optimizing the user service experience in a certain area, with the help of AI agents, the network first conducts a comprehensive analysis to clarify the task decomposition steps required to achieve this goal, including service identification, service experience evaluation, service degradation problem diagnosis, and service optimization decision-making. Subsequently, the network distributes these tasks to the most appropriate network functions located across the network. Enabled by the collaboration among AI agents, automated analysis, diagnosis, and optimization of service experience can be achieved.

Objective 2: the distributed intelligent collaboration aims to empower the network with distributed AI-based applications. The distributed intelligent collaboration can seamlessly orchestrate AI tasks between terminal/third-party agents and network-located agents, ensuring low-latency and high-Quality of Service (QoS) guarantees that significantly enhance the application experience.

NOTE 3 – The collaboration methods for AI tasks include AI joint inference based on AI model splitting [b-Patel] or speculative decoding [b-Chen], AI joint training based on federated learning [b-ITU-T Y.3186], and iterative optimization of multi-agent reinforcement learning [b-Yang2].

Based on the two objectives described above, Figure 6-1 provides an overview of distributed intelligence collaboration in IMT-2020 and beyond, performed via the collaboration of AI tasks distributed across the IMT-2020 network and the user equipment (UE).

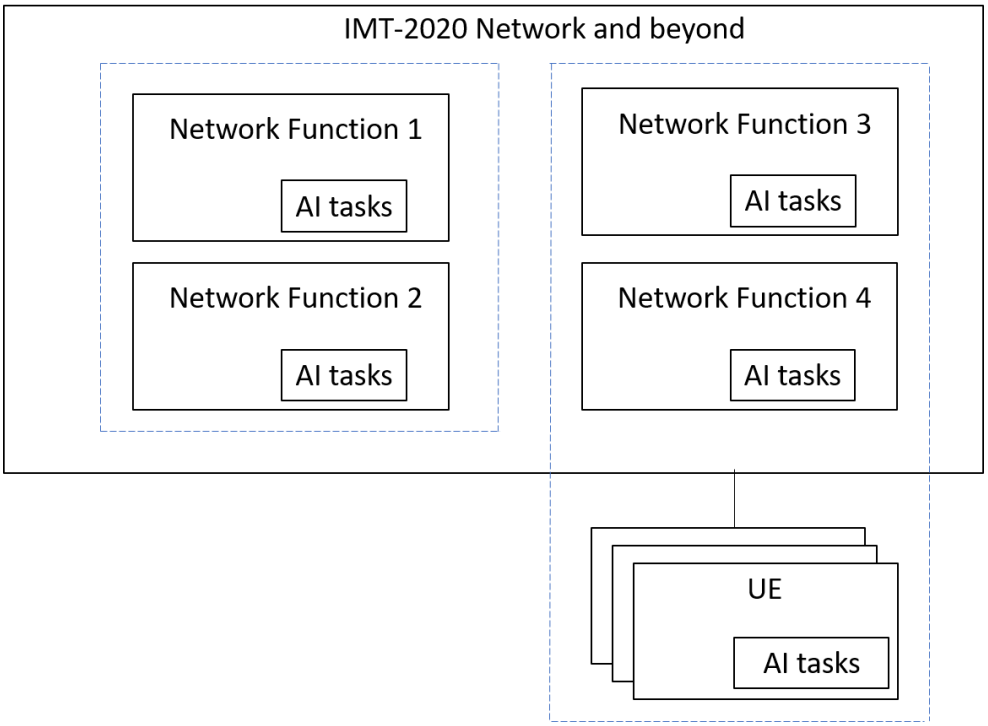

Figure 6-1 Overview of distributed intelligence collaboration in IMT-2020 and beyond

In the context of IMT-2020 networks, a network function is a processing function inside the network [ITU-T Y.3100]. AI technologies are integrated into network functions to execute specific tasks, this being represented by network functions with built-in AI tasks in Figure 6-1. The same applies to UEs with built-in AI tasks.

NOTE 4 - The two dashed boxes in Figure 6-1 represent two examples of distributed intelligent collaboration: the dashed box encompassing Network Function 1 and Network Function 2 indicates that the two network functions collaborate in a distributed manner to execute AI tasks, representing AI-empowered network scenarios, such as the AI-based optimization of user service experience; the

dashed box encompassing UE, Network Function 3 and Network Function 4 also indicates that UE and the two network functions collaborate in a distributed manner to execute AI tasks, but representing AI-empowered network-UE scenarios, such as the AI joint inference between the network and the UE.

6.2. Enabling technologies for distributed intelligence collaboration in IMT-2020 and beyond

6.2.1 AI agent as the core cognitive engine for AI collaboration

The rapid advancement of Large Language Models (LLMs) has catalyzed the development and application of AI agents as an emerging technology. An AI agent is an automated entity that senses and responds to its environment and takes actions to achieve its goals [b-ISO/IEC 22989]. AI agents will serve as the fundamental building blocks of an intelligent network.

While powerful, single AI agent's capabilities are inherently limited when dealing with complex, dynamic environments. To tackle more sophisticated tasks, the interest is now shifting from single AI agent systems to multi-AI agent systems capable of collaborative perception, learning, reasoning, and action.

AI agents can assume different roles and perform different AI tasks, thereby enabling intelligent collaboration. The key aspect lies in the enablement of a distributed intelligent collaboration, where AI agents are deeply integrated with and empowered by the network.

The AI agent technology enables the use of AI agents as the core cognitive engine for AI collaboration.

AI agents use a large model as their core cognitive engine to power three essential modules: Planning (for goal decomposition), Tools (for external interaction), and Memory (for context retention). This enables the AI agents to reason, learn from interactions, and execute complex tasks in dynamic environments, forming a robust and adaptive system for generalized problem-solving [b-Julia].

NOTE 1 - The cognitive capabilities of an AI agent may be implemented using models of varying scales and types. While large models provide the foundation for robust generalization, reasoning, and planning capabilities [b-Xi], compressed models (obtained via techniques like pruning and distillation [b-Liu]) or specialized small models offer viable alternatives to achieve high efficiency, low latency, or domain-specific optimization.

AI agents enable the translation of high-level business or operational intents into network strategies, referring actionable steps for task execution. Subsequently, they can perform task decomposition and orchestration across network domains, and can finally generate dynamic, context-aware policies for execution. These capabilities are fundamental for achieving autonomous, intent-driven networking, ensuring that collective network behaviours are aligned with overarching objectives.

NOTE 2 – As an example, upon receiving the complex AI task requirement "Live Broadcast QoS Guarantee for Mega Sporting Events", an AI agent can translate ambiguous business-level requirements into precise network intents by integrating telecom concepts (such as QoS, Application Programming Interface (APIs), network topology) with enterprise context, and dynamically orchestrate and generate task execution strategies to enforce specific QoS policies for live broadcast traffic.

6.2.2 AI tasks distribution for advanced scheduling and control

Advanced scheduling and control can be enabled by the distribution of AI tasks across different user and network devices, thereby achieving optimization in terms of resource utilization, energy efficiency, and service quality. Moreover, the distributed deployment of AI tasks enables proximity to data sources or data consumers during execution, thereby optimizing task performance. The AI

tasks distribution methods include online real-time processing methods and offline pre-training processing methods.

NOTE 1 – Online real-time processing methods involve low-complexity optimization-based approaches to provide real-time computational processing, real-time information exchange, and real-time scheduling deployment, thus achieving real-time scheduling and control.

NOTE 2 – Offline pre-training processing methods employ AI-based approaches to conduct preliminary analysis or training on vast amounts of tasks and network data, thereby forming intelligent inference tools. Through historical data analysis, these tools provide task scheduling strategy, thus having high complexity yet precise control. These approaches can be implemented based on deep learning or reinforcement learning [b-ITU-T Y.3181].

6.2.3 Efficient collaboration execution mechanisms for multiple AI tasks

These efficient collaboration execution mechanisms are categorized based on the primary objective of the AI tasks: distributed AI training focuses on facilitating model parameter convergence across data and devices; distributed AI inference aims at accelerating the process of inference execution; and distributed AI collaboration orchestrates multiple AI agents to jointly accomplish complex tasks through well-defined interaction protocols.

NOTE 1 – For AI training tasks, advanced techniques for distributed AI training include parameter servers, allreduce communication, decentralized aggregation [b-Langer], and federated learning [b-ITU-T Y.3186]. These mechanisms optimize the training speed, enabling swift exchange of model parameters and gradients.

For AI inference tasks, support for AI models - especially large AI models - is provided, facilitated by distributed deployment mechanisms. Due to the limitations of the computing power and memory capacity of a single device, large AI models may be segmented into sub-models and allocated to multiple user and network devices for parallel inference. Advanced techniques for distributed AI inference include speculative execution through collaboration between large and small models [b-Chen], and split inference of AI models [b-Patel].

For AI collaborative tasks, the execution is achieved through multi-agent systems (MAS) where heterogeneous agents perform distinct sub-tasks in parallel. In such multi-agent collaborative scenarios, AI agents assume differentiated roles, including orchestrators for control and coordination, analysts for specialized task analysis, and executors for specialized task execution. Their collaboration relies on two key technologies: inter-agent communication (e.g., Agent-to-Agent [b-Dibia]) for information exchange; and dynamic resource coordination for real-time sharing of computing, network and storage resources [b-Dibia].

7. Requirements of distributed intelligence collaboration in IMT-2020 and beyond

- It is required for distributed intelligence collaboration to support the analysis and understanding of complex network intents, such as high-level business or operational objectives.
- It is required for distributed intelligence collaboration to support the translation of complex network intents into network strategies, referring to actionable steps for task execution.
- It is required for distributed intelligence collaboration to support the dynamic generation of AI tasks according to the network strategies.
- It is required for distributed intelligence collaboration to support the distributed deployment of AI tasks.
- It is required for distributed intelligence collaboration to support advanced scheduling and control of AI tasks.

- It is required for distributed intelligence collaboration to support coordinated control of multiple AI tasks, including task execution node selection, task deployment to nodes, inter-task collaboration (e.g., managing execution order and dependencies), task monitoring, and policy control.
- It is required for distributed intelligence collaboration to support distributed collaborative execution of multiple AI tasks, including distributed AI inference, distributed AI training and distributed AI collaboration.
- It is required for distributed intelligence collaboration to support information exchange between AI tasks, such as parameter passing between AI tasks, result notification, task triggering.
- It is required for distributed intelligence collaboration to mark AI decisions and AI-generated data at the APIs and reference points.

8. Functional framework of distributed intelligence collaboration in IMT-2020 and beyond

This clause provides a description of the functional framework of distributed intelligence collaboration and its functional components, and provides guidelines for the realization of such framework.

The functional framework of distributed intelligence collaboration described in this clause supports the requirements defined in clause 7.

8.1. The role of AI agents in the functional framework

In order to support intelligent analysis, orchestration, scheduling, and efficient execution of AI tasks for diverse and complex business demands, this framework introduces multiple AI agents into the network. The core concept of the distributed intelligence collaboration framework is to deploy multiple AI agents to serve the different tasks. Depending on their specific responsibilities, these AI agents are of different types, assume different roles, are deployed across different locations within the network, and collaborate to achieve their objectives.

These AI agents can be categorized into two types: Orchestration agents and Execution agents. Orchestration agents are responsible for analyzing and decomposing the overarching goal into specific tasks, and for orchestrating and scheduling their execution. Leveraging large AI models as their foundation, they are categorized as “strong agents” with high-level cognitive capabilities. Execution agents, on the other hand, handle relatively simpler and/or specific AI tasks, typically utilizing small or compressed AI models as their foundation, and are categorized as “weak agents” specialized in context-specific implementations. Through the distributed deployment and collaboration of AI agents with different roles, complex network tasks can be accomplished collectively.
NOTE - Since AI large models require more resources and longer execution time compared to small or compressed AI models, deploying AI agents exclusively based on large models throughout the network may be not cost-effective or efficient. Consequently, AI agents leveraging large models may be employed to analyze and decompose complex problems, whereas AI agents utilizing small or compressed AI models may handle relatively simpler and/or specific AI tasks. This strategy optimizes both cost and execution efficiency while satisfying the requirements for complex tasks.

Depending on the specific objectives and tasks, there may exist different kinds of Orchestration agents and Execution agents:

- Orchestration agents may be, for example, agents responsible for end-to-end network service experience assurance, or agents that orchestrate multi-agent collaborative services for smart logistics.
- There are two kinds of Execution agents, one providing service-specific capabilities (e.g., service experience assessment, problem diagnosis, optimization strategy generation) while the other one providing foundational capabilities (e.g., data, computing, and connectivity).

8.2. Functional framework of distributed intelligence collaboration

Figure 8-1 depicts the functional framework of distributed intelligence collaboration in IMT-2020 and beyond, with [ITU-T Y.3102] being the reference functional framework of the IMT-2020 network over which this framework is built.

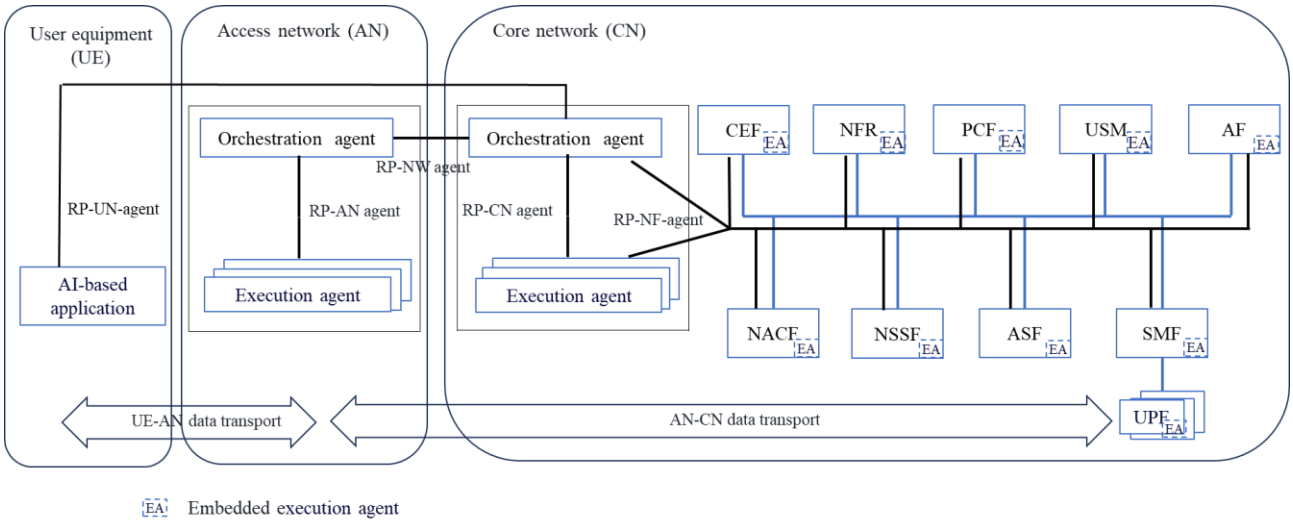

Figure 8-1: Functional framework of distributed intelligence collaboration

This framework introduces Orchestration agents and Execution agents in the IMT-2020 network.

An Orchestration agent is an independent network function. Typically, a single Orchestration agent is deployed per network domain (e.g., core network (CN) or access network (AN)) for centralized intent parsing and orchestration, though multiple Orchestration agent deployments are possible for performance scaling. The Orchestration agent in the CN is responsible for processing both user-originated and network-originated intents. The Orchestration agent in the AN, however, is solely responsible for processing network-originated intent requests from the AN.

An Orchestration agent autonomously analyzes and makes decision for complex AI tasks, whether triggered by pre-configured network requirements or dynamic user requests. It generates task execution strategies and sends task instructions to multiple Execution agents, while specifying the upstream and downstream relationships among these Execution agents to orchestrate the task workflow.

An Orchestration agent monitors and evaluates the effectiveness of task execution (e.g., completion rate, resource utilization efficiency, and achievement of QoS targets) by the associated Execution agents. When the effectiveness of a task execution does not meet the predefined objectives, the Orchestration agent conducts further analysis to determine whether adjustments to the task execution strategy are needed.

An Execution agent may be an independent network function or may be embedded as a component within an existing Network Function (NF), enhancing the NF's intelligent capabilities. The Execution agents are typically deployed in a distributed manner, which facilitates the deployment of AI tasks at the most suitable locations, such as those closest to users or data sources, thereby enhancing the execution efficiency of AI tasks.

The Execution agents execute AI tasks according to the task instructions from the Orchestration agent, and report the task execution results or interact with other Execution agents as required by the task instructions.

The Orchestration agent and Execution agents work in tandem, with a clear division of labor, leveraging intelligent technologies for decomposing complex AI tasks and coordinating their execution with the support of AI/ML technologies.

The UE hosts AI-based applications, which are supported by distributed AI tasks within the network architecture.

8.3. Reference points

As illustrated in figure 8-1, the reference points of distributed intelligence collaboration in IMT-2020 and beyond are as follows:

- RP-UN-agent: The reference point between the UE and the Orchestration agent in the CN. The RP-UN-agent enables the collaboration between the UE and the Orchestration agent in the CN. The RP-UN-agent is expected to adopt a Non-Access Stratum (NAS)-based protocol [ITU-T Y.3102].
- RP-NW-agent: The reference point between the Orchestration agent in the CN and the Orchestration agent in the AN. The RP-NW-agent enables the collaboration between the Orchestration agents in the CN and the AN. The RP-NW-agent is expected to adopt a protocol designed for inter-agent communication, such as the Agent-to-Agent (A2A) protocol [b-Dibia] .
- RP-CN-agent: The reference point among AI agents in the CN. The RP- CN-agent enables the collaboration among AI agents within the CN, both between Orchestration agent and Execution agents, and among different Execution agents. The RP-CN-agent is expected to adopt a protocol designed for inter-agent communication, such as the A2A protocol.
- RP-AN-agent: The reference point between AI agents in the AN. The RP- AN-agent enables the collaboration among AI agents in the AN, both between Orchestration agent and Execution agents, and among different Execution agents. The RP-AN-agent is expected to adopt a protocol designed for inter-agent communication, such as the A2A protocol.
- RP-NF-agent: The reference point between AI agent(s) in the CN (Orchestration agent or Execution agents) and NFs in the CN. The RP-NF-agent enables the interaction between AI agent(s) and NFs within the CN. The RP-NF-agent is expected to adopt a protocol designed for communication among AI agents and non-AI agent entities (such as functions and resources), e.g., the Model Context Protocol (MCP) [b-Dibia].

9. Procedure of distributed intelligence collaboration in IMT-2020 and beyond

Figure 9-1 depicts the procedure of distributed intelligence collaboration.

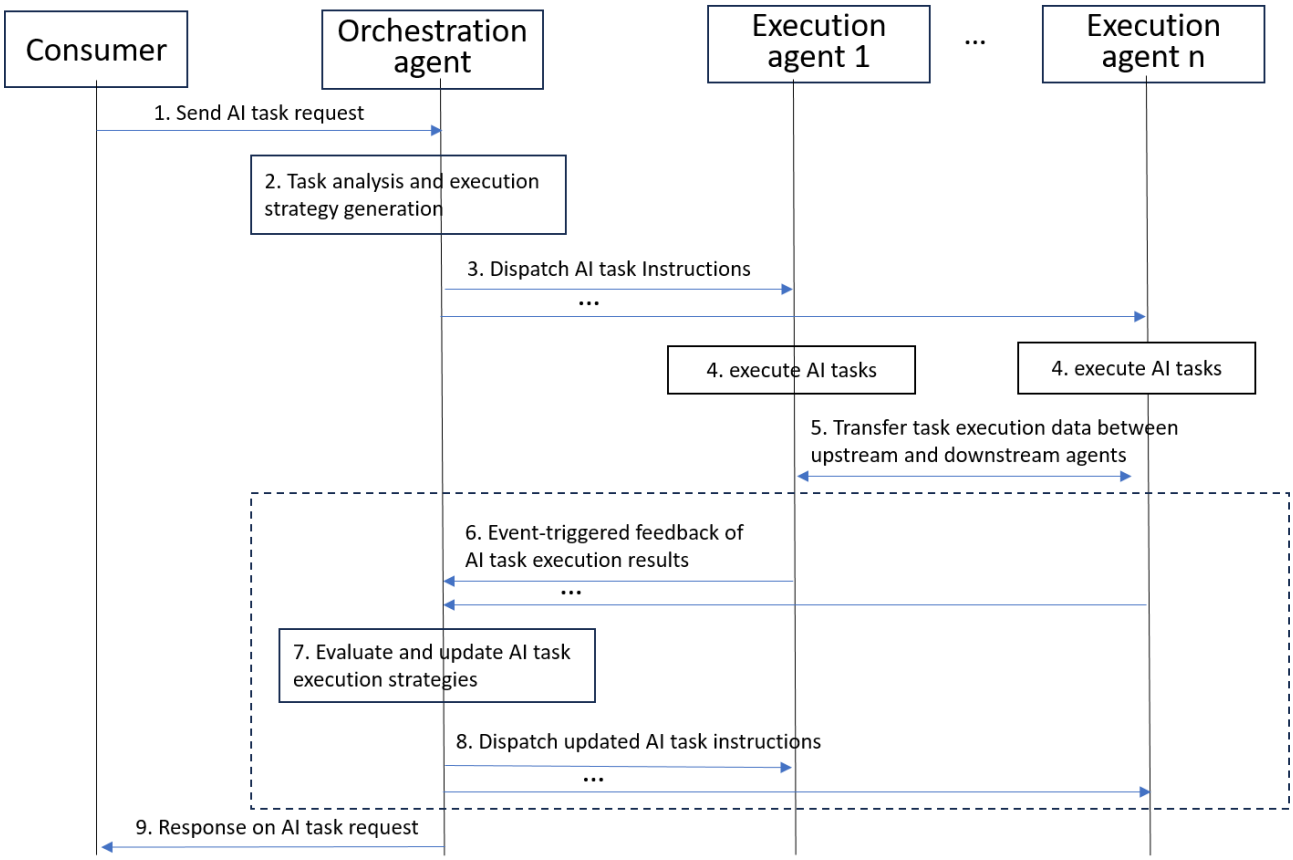

Figure 9-1 – Procedure of distributed intelligence collaboration

The steps shown in Figure 9-1 are as follows:

1) The consumer sends an AI task request to the Orchestration agent.

NOTE 1 – The consumer may be a NF or UE. In addition to dynamic requests from consumers, the Orchestration agent can also receive AI task requests configured and delivered by the Network Management System (NMS).

2) The Orchestration agent responsively analyzes the AI task request, generating task execution strategies (composed by task instructions and deployment schemas)

NOTE 2 – As a strong agent, the Orchestration agent dynamically orchestrates task execution strategies by deriving task requirements from the received requests, invoking external capabilities through the tools module, retrieving historical expertise and competencies from the memory module, querying network runtime information and decomposing high-level goals into actionable steps via the planning module.

3) The Orchestration agent dispatches the task instructions to the Execution agents, including task identifiers, model identifiers, execution parameters, task requirements, upstream/downstream Execution agent dependencies, trigger conditions.

4) The Execution agents execute the AI tasks.

NOTE 3 – Different Execution agents may leverage distinct AI models as described in Clause 8.1.

5) During the AI tasks execution, the Execution agents transfer task execution data between upstream and downstream peers according to task instructions.

6) Upon meeting trigger conditions stipulated in the task instructions, the Execution agents asynchronously feedback the AI tasks execution results to the Orchestration agent.

7) Leveraging the AI tasks execution results fed back by the Execution agents, the Orchestration agent evaluates the effectiveness of the deployed task execution strategies and dynamically updates the AI task execution strategies if necessary.

8) The Orchestration agent dispatches the updated AI tasks instructions to the Execution agents.

9) The Orchestration agent responds on the AI task request to the consumer.

10. Security considerations

The security considerations on distributed intelligence collaboration include the following aspects:

- The security aspects related to the AI task request process;
- The security aspects related to the AI task distribution process and to the AI task execution results among the Orchestration agent and the Execution agents, as well as to the data transfer across Execution agents.

In addition, the security and personal data protection considerations on distributed intelligence collaboration should be aligned with the requirements specified in [ITU-T Y.3101].

Appendix I

Use cases of distributed intelligence collaboration in IMT-2020 and beyond

(This appendix does not form an integral part of this Recommendation.)

The following describes use cases of distributed intelligence collaboration in IMT-2020 and beyond.

Use case 1: AI agent-enabled service customization in the XaaS network

This use case leverages AI agents to introduce dynamic intelligence and service customization into the XaaS (Everything as a Service) network. While the modern XaaS framework already expands network capabilities beyond mere connectivity to include sensing, data, computing, and AI, service fulfilment remains largely constrained by pre-defined rules and templates. The introduction of AI agents overcomes this limitation. The AI agents interpret diverse and complex user requests, intelligently orchestrating the underlying capabilities of the XaaS fabric. This enables truly personalized, on-demand services, moving beyond static procedures to autonomously deliver optimized outcomes tailored to specific consumer needs.

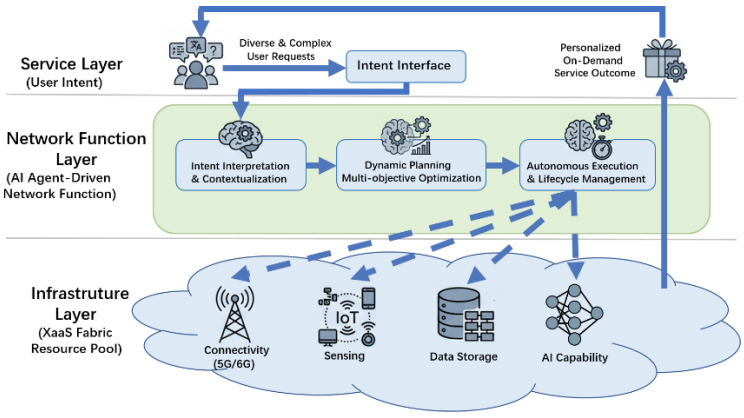

Figure I.1 – AI agent-enabled service customization in the XaaS network

Use case 2: Intent-driven service realization via UE-CN AI agent collaboration

This use case enables a network operator to proactively understand and fulfil complex user intents through direct, intelligent communication between AI agents residing in the User Equipment (UE AI agent in figure I.2) and the core network (CN AI Agent in figure I.2). Instead of relying on manual configurations or rigid service policies, the UE AI agent interprets the user's high-level goals (e.g., "optimize my video conference quality during my commute") and communicates this intent to its CN counterpart. The CN AI agent then collaborates with network functions to dynamically orchestrate the necessary resources - such as bandwidth, computing, or security policies - to realize the requested service. This closed-loop interaction enables the network operator to offer highly personalized, context-aware services, transforming the network from a static pipe into an intelligent, responsive partner that anticipates and fulfils user needs.

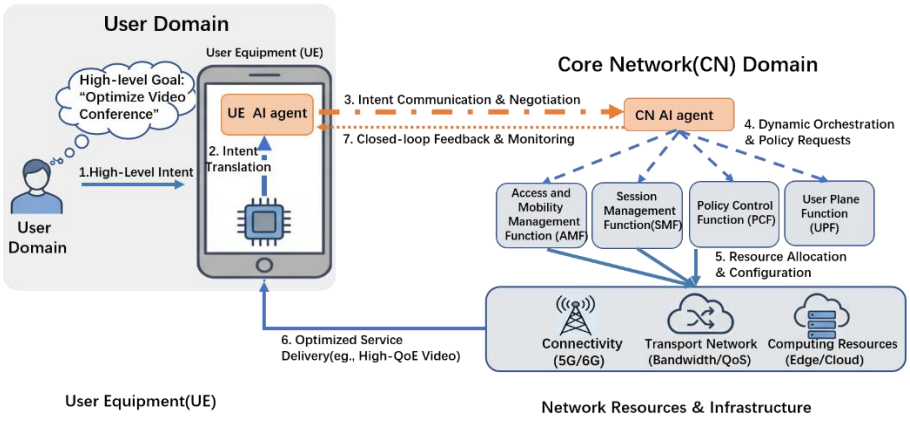

Figure I.2 –Intent-driven service realization via UE-CN AI agent collaboration

Use case 3: AI agent-driven autonomous optimization for mobile edge services

This use case demonstrates how AI agents enable fully autonomous optimization of edge services in dynamic mobile environments. A distributed network of AI agents - residing at the UE, edge site, and application level - continuously collaborates to maintain optimal service quality. The UE AI agent monitors real-time user experience and device context, while Edge AI agents analyze application performance and network conditions across different applications. Through autonomous awareness and self-reflection capabilities, these agents proactively coordinate resource allocation and configuration adjustments in real-time. This closed-loop intelligence allows the system to adapt seamlessly to mobility patterns and network fluctuations, delivering consistent high-quality experience without manual intervention.

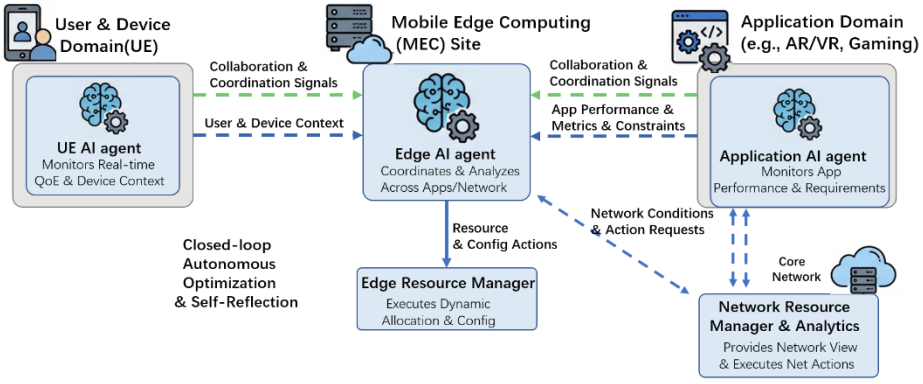

Figure I.3 –AI agent-driven autonomous optimization for mobile edge services

Bibliography

[b-3GPP TS 28.312] 3GPP TS 28.312 V18.0.0 (2023), Intent driven management services for mobile networks (Release 18).

[b-ISO/IEC 22989] ISO/IEC 22989:2022, Information technology — Artificial intelligence — Artificial intelligence concepts and terminology.

[b-ITU-T Y.3181] Recommendation ITU-T Y.3181 (2022), *Architectural framework for machine learning sandbox in future networks including IMT-2020*

[b-ITU-T Y.3186] Recommendation ITU-T Y.3186 (2024), *Requirements and framework for distributed joint learning to enable machine learning in future networks including IMT-2020*

[b-ETSI GR ENI 004] ETSI GR ENI 004 V3.1.1 (2023), Experiential Networked Intelligence (ENI); Terminology.

[b-Chen] Chen C, Borgeaud S, Irving G, et al. Accelerating large language model decoding with speculative sampling[J]. arXiv preprint arXiv:2302.01318, 2023.

[b-Dibia] Dibia V. Designing Multi-Agent Systems: Principles, Patterns and Implementation for AI Agents[M]. Victor Dibia, 2025.

[b-Julia] Julia Wiesinger, Patrick Marlow and Vladimir Vuskovic. Agents [white paper], Google, 2024.

[b-Langer] Langer M, He Z, Rahayu W, et al. Distributed Training of Deep Learning Models: A Taxonomic Perspective[J]. IEEE Transactions on Parallel & Distributed Systems, 2020, 31(12): 2802-2818.

[b-Liu] Liu D, Zhu Y, Liu Z, et al. A survey of model compression techniques: Past, present, and future[J]. Frontiers in Robotics and AI, 2025, 12: 1518965.

[b-Patel] Patel P, Choukse E, Zhang C, et al. Splitwise: Efficient generative llm inference using phase splitting[C]//2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA). IEEE. 2024

[b-Xi] Xi Z, Chen W, Guo X, et al. The rise and potential of large language model based agents: A survey[J]. Science China Information Sciences, 2025, 68(2): 121101.

[b-Yang2] Yang Y , Wang J .An Overview of Multi-Agent Reinforcement Learning from Game Theoretical Perspective[J]. 2020.

_____