Demonstration of Centralized Multi-Agentic AI for Telecom Network Operation

Pol González and Luis Velasco*

Optical Communications Group (GCO), Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
**e-mail: luis.velasco@upc.edu*

ABSTRACT

The evolution towards autonomous networking requires intent-based management. The application of agentic Artificial Intelligence (AI) to network control and management facilitates such evolution as it enables to respond to user intents by breaking down complex tasks and taking independent actions using tools. In this demonstration, we showcase a centralized multi-agentic AI control plane for autonomous telecom network operation. The proposed architecture combines a Multi-Agent System (MAS) and an on-premises Large Language Model (LLM). Network tools and traditional control are integrated to ensure determinism and explainability to operational decision making. A workflow for packet connectivity provisioning is used to illustrate how natural-language operator intents are translated into network configuration actions.

1 OVERVIEW

Multi-Agent Systems (MAS) have demonstrated to play a key role the distributed control of telecom network operation, especially when making operational decisions near-real time is required (see, e.g., our previous work in [1], [2]). However, the use of Artificial Intelligence (AI) in the centralized control plane has been somehow confined to auxiliary systems, like Network Digital Twins [3], as network operation remains dominated by traditional software systems, like Software-Defined Networking (SDN) controllers.

However, the recent advances in Large Language Models (LLM) are enabling the emergence of autonomous AI agents capable of planning, reasoning and interacting with complex environments, which pays the way to its application for Telecom Network Operation. Nonetheless, several issues remain unsolved that prevent their fully deployment for operating commercial telecom networks, including: *i*) using *frontier* LLMs running outside operator's premises entails privacy, security and economic concerns; *ii*) operational decision making must be deterministic and explainable, which is difficult to achieve when such decision-making is based on LLM inference; and *iii*) the SDN layer is a key component for the operation of currently deployed networks.

The demonstration exhibits a centralized control plane architecture that include an AI MAS, where agents have specific roles and coordinate among them using the open Agent2Agent (A2A) protocol [4]. AI agents make use of an on-premises LLM, which provides the required planning and reasoning capabilities, while avoiding security issues. Network operation tools and data sources are integrated using the open Model Context Protocol (MCP) [5] so they can participate in operational workflows, which brings the needed determinism to network operation. The SDN-based control plane is integrated by extending their north-bound interface to MCP, which keeps the interfaces with the network infrastructure unchanged. Such integration of Agentic AI and SDN is follows the guidelines being defined by the Telecom Infra Project [6]. A workflow for connectivity set up at the packet network is used to illustrate the operation capabilities of the proposed multi-agentic AI-based control plane. The live demonstration aims to showcase the practical benefits of Agentic AI to achieve Autonomous Network Level 4 (ANL4) [7] and give guidelines on how Agentic AI and LLMs can integrate with existing network tools and SDN control.

2 INNOVATION AND RELEVANCE

The demonstration is aimed at network operators, system vendors, and researchers focused on the application of Agentic AI to the to extend the autonomous control of telecom infrastructures. The demo is particularly relevant today, as AI evolution to Agentic AI enables to accomplish complex, multi-step goals with limited supervision. AI agents can work independently to achieve objectives by breaking down high-level requests into a sequence of smaller tasks. In addition, AI agents can use external tools to ensure determinism and to take action in the network infrastructure. LLMs empower AI agents to reason and choose the best course of action, facilitating continuous learning. All these are characteristics of ANL4.

979-8-3195-4420-9/26/$31.00

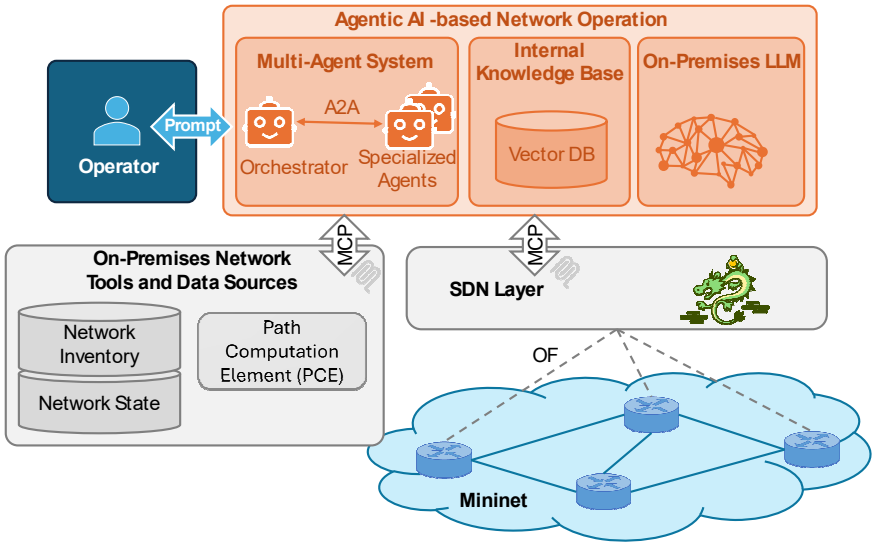

Figure 1 Proposed Agentic AI-Based Network Control Plane Architecture

In this demonstration, we aim to showcase the capabilities of Agentic AI for the control of telecom networks. The setup will include an agentic AI-based network operation layer capable to process human operator prompts. This layer, will include an AI MAS and an on-premises LLM, all running on an AI desk computer. Integration with external tools, like path computation, as well as with a SDN controller will be demonstrated through a workflow for connectivity provisioning. The live demonstration has been designed to resonate with the audience, especially those interested in autonomous network operation, Agentic AI and the integration of LLMs to the control of the telecom network.

3 DEMO CONTENT AND IMPLEMENTATION

Figure 1 presents the proposed agentic AI control plane for telecom network operation. The system includes an AI MAS where a number of specialized AI agents collaborate through the A2A protocol. Agents can be specialized in different network layers, e.g., optical or packet, and/or tasks, e.g., connectivity management. The specialized agents are coordinated by an AI orchestrator that manages work distribution, context sharing, and result aggregation. Specifically, a *handoff* agent orchestration pattern has been selected to enable dynamic delegation of tasks between agents. The AI orchestrator includes a semantic engine that captures the intent of the prompts and maps it to a specific workflow plan by using on-premises LLM reasoning and then, it decides which specialized agent should take over each task. This approach is key to implementing complex agentic AI workflows. The specialized agents use existing tools and data sources, e.g., a network planning tool, the network state, and the network inventory. The use of such tools ensures deterministic, explainable decision making. In addition, network programmability is delegated to the SDN control layer, which ensures seamless integration with the Data Plane (DP).

The implementation has been kept as simple as possible to showcase the main innovations of the demo and only the needed and simpler external tools have been integrated. Specifically, a simple Path Computation Element (PCE) is in charge of computing optimal paths for the current network state. The PCE integrates a MCP server as a standardized way to connect AI agents to external tools. The same approach has been followed with the Ryu SDN controller [8], which uses OpenFlow (OF) to control the packet switches in a virtual network deployed on Mininet. A similar deployment can be configured using up-to-date SDN controllers, like the Teraflow SDN framework [10].

The simplified workflow for packet connectivity set up is shown in Figure 2. The workflow is initiated by the human operator that instructs the AI-based network control plane to set up packet connectivity using text, which include the needed parameters, like the end points, required bandwidth, etc. (step 1 in Figure 2). Because zero-shot prompting is desired, no examples are included in the prompt; instead, the workflow to be followed to implement the task is already defined and stored in the vector database (DB). The orchestrator takes the user prompt, generates the embeddings and finds the workflow in the vector DB (this step is not shown in the workflow in Figure 2 for simplicity). The vector DB is initially populated from a configuration file, in which each workflow is defined by a name, a description, and an ordered sequence of steps. For this connectivity provisioning workflow, this results, e.g., on a two-step plan: first, *"compute the shortest route between source and destination nodes"*, and second, *"provision and bring up a new IP connection on the SDN-controlled network"*.

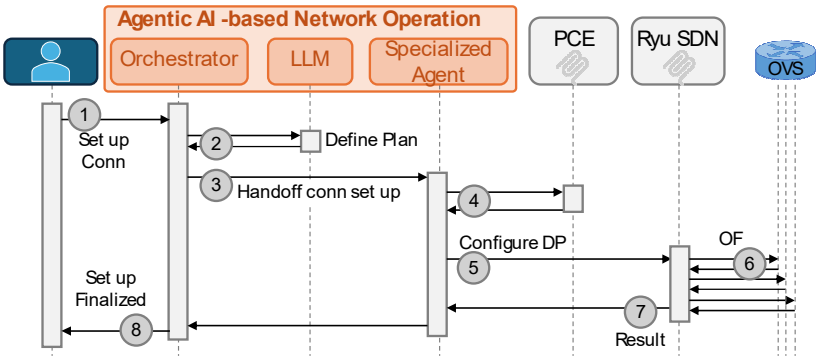

Figure 2 Simplified workflow for packet connectivity set up

Next, the orchestrator AI agent invokes the LLM to extract the arguments from the prompt and to generate a plan based on the defined workflow (step 2). Armed with the specific plan, the orchestrator finds the most suitable specialized agent to implement the plan. For this demo, we have implemented a specialized AI agent for packet network. Then, the orchestrator delegates the execution of the plan to the specialized agent (step 3). The specialized packet agent requests the PCE using MCP commands to compute an optimal path using the specification from the user and taking into account the current network state (step 4). This step is also simplified in Figure 2 as it involves several iterations, including the LLM. Once the optimal path has been computed human-in-the-loop can be implemented; in that case the human operator should explicitly accept the found configuration for the connectivity before the rules are actually implemented in the network. The specialized AI agent uses the defined configuration and requests the SDN layer to implement the needed rules in the packet switches along the path (step 5). Similarly as in step 4, this involves several iterations between the specialized AI agent and the SDN controller through MCP. The SDN controller programs the Open-Virtual Switches (OVS) in Mininet using OF (step 6). Once the configuration has been implemented, as confirmed by the SDN controller (step 7), the specialized agent informs the orchestrator AI agent specifying the details, which are returned back to the human operator (step 8).

All the components of the architecture, including the vector DB, the LLM and embedding inference, the MCP server, the agent, and the orchestrator run on premises on an NVIDIA DGX Spark GPU equipment. Components have been implemented in Python v. 3.13. Ollama is used as inference engine for serving the LLM models and ChromaDB has been selected as vector DB. The LLM model is Mistral 7B with 7.3 billion parameters. The MCP server has been implemented using the FastMCP library. The testbed includes a software-defined networking built on Mininet and the Ryu controller framework that runs on an independent VM. The emulated topology consists of a minimal two-host, single-switch network connected through an OpenFlow 1.3 OVS switch that delegates control to a remote Ryu SDN controller. The controller application operates as a standard L2 learning switch, installing MAC-to-port forwarding rules. In addition, it exposes a REST API with three endpoints that enable an external entity (in this case the agent through MCP) to toggle a traffic-isolation policy between the two hosts at runtime.

Figure 3 shows a capture of the exchanged messages for the workflow, following the same numbering as the one used in Figure 2. The workflow is triggered by a user when a prompt is sent to the orchestrator (message 1). The orchestrator accepts natural-language commands such as *"Establish an IP connection from H1 to H2 with 1GB bandwidth"* and translates them into concrete network control actions. Upon receiving a prompt, the orchestrator processes the text to identify the workflow that best matches the user's intent. The matched workflow name is used as a query against a vector DB that stores the system's supported workflows.

Next, the orchestrator determines the specialized agent for every step and hands off its execution. The specialized agent opens a session with the network's tool server using the MCP and retrieves the list of available tools on that specific MCP server. The server replies with a list of tool specifications, each containing a name, a plain-text description, and a JSON schema describing the tool's input arguments. Then, the specialized agent submits a structured chat payload to the LLM, that will generate the action to be executed for the step based on the available tools on the MCP server (message 2 more in detail in Figure 5). First, a path-computation call is delegated to a PCE that loads the network topology and state and calculates the shortest path connecting the source and destination node (message 4). A packet connectivity set up call is sent to the Ryu SDN Controller containing the source and destination nodes (message 5). The Ryu SDN Controller then applies the necessary rules to configure a flow between the selected hosts allowing the connectivity (see the pings in Figure 4). Finally, the specialized agent receives a summary of the applied actions on the DP that will then forward to the orchestrator (message 7). The orchestrator, based on the received

	Source	Destination	Info
①	User	Orchestrator	POST /prompt
	Orchestrator	LLM	POST /plan
②	LLM	Orchestrator	HTTP/1.1 200 OK
	Orchestrator	Agent	POST /execute_plan
	Agent	PCE	POST /compute_route
④	PCE	Agent	HTTP/1.1 200 OK
③	⑤ Agent	SDN	POST /service/ip/create
	⑦ SDN	Agent	HTTP/1.1 200 OK
	Agent	Orchestrator	HTTP/1.1 200 OK
	⑧ Orchestrator	User	HTTP/1.1 200 OK

Figure 3 Workflow messages exchanged

```
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
mininet> h1 ping h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=10.5 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.288 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=0.079 ms
```

Figure 4 Connection establishment on Mininet


```
{'steps': [{'intent': 'compute the shortest route between source and
destination nodes for a candidate connection'}, {'intent': 'provision and bring
up a new IP connection on the SDN-controlled network'}],
'plans': [{'tool': 'compute_route', 'args': {'src_node': 'H1', 'dst_node': 'H2'},
'source': 'llm'}, {'tool': 'establish_ip_connection', 'args': {'src_node': 'H1',
'dst_node': 'H2', 'bitrate_gbps': 1}, 'source': 'llm'}] }                    (2)
```

```
{"trace": [
    {"step": 1,
      "intent": "compute the shortest route between source and destination
          nodes for a candidate connection",
      "tool": "compute_route",
      "args": { "src_node": "H1", "dst_node": "H2"},
      "agent": "IPConnectionAgent",
      "result": {
          "src": "H1", "dst": "H2",
          "path": ["H1", "R1", "R2", "R3", "H2"], "hops": 4} }
    {"step": 2,
      "intent": "provision and bring up a new IP connection on the SDN-
          controlled network",
      "tool": "establish_ip_connection",
      "args": {"src_node": "H1", "dst_node": "H2", "bitrate_gbps": 1},
      "agent": "IPConnectionAgent",
      "result": {
          "connection_id": "H1-H2",
          "status": "UP",
          "message": "Connection H1-H2 established"}}] }                    (8)
```


Figure 5 Selected messages

message from the specialized agent, generates a message that will be returned to the user as a result to the initial input prompt (message 8 more in detail in Figure 5).

4 CONCLUSION

An Agentic AI-based centralized control plane architecture for telecom network has been proposed and demonstrated. By coordinating specialized AI agents through an orchestrator and exposing network tools and the SDN controller through MCP, the proposed architecture combines the planning and reasoning capabilities of LLMs with the determinism and explainability required for real network operation. A workflow showcasing packet connectivity provisioning through an Agentic AI stack has been demonstrated on top of the Ryu SDN Controller and an emulated DP based on Mininet. Running all the components on premises, including the LLM and the full agentic stack addresses the privacy, security, and economic concerns associated with frontier models, while the MCP based integration keeps southbound interfaces toward the DP unchanged.

ACKNOWLEDGEMENT

The research leading to these results has received funding from the European Union's Horizon Europe research and innovation programme under G.A. No. 101092766 (ALLEGRO), the project PID2024-157824OB-I00 funded by MICIU/ AEI/10.13039/501100011033 /FEDER, UE and from ICREA.

REFERENCES

[1] P. Gonzalez *et al*., "Near-Real-Time 6G Service Operation Enabled by Distributed Intelligence and In-Band Telemetry," IEEE/OPTICA J. of Optical Communications and Networking, vol. 17, pp. A247-A258, 2025.

[2] H. Shakespear-Miles *et al*., "Centralized and Distributed Approaches to Control Optical Point-to-Multipoint Systems Near-Real-Time," IEEE/OPTICA J. of Optical Communications and Networking, vol. 16, pp. 565-576, 2024.

[3] M. Devigili *et al*., "Applications of the OCATA Time Domain Digital Twin: from QoT Estimation to Failure Management," IEEE/OPTICA J. of Optical Communications and Networking, vol. 16, pp. 221-232, 2024.

[4] Agent2Agent (A2A) Protocol. [Online] https://a2a-protocol.org/

[5] Model Context Protocol (MCP). [Online] https://modelcontextprotocol.io/

[6] Open Optical & Packet Transport community at TIP. [Online] https://www.telecominfraproject.com/post/open-transport-community -objectives-for-2026-and-beyond

[7] TM Forum, "Autonomous Network Levels Evaluation Methodology v1.2.0 (IG1252)," 2023

[8] Ryu SDN framework. [Online] https://ryu-sdn.org/

[9] Mininet. [Online] https://mininet.org/

[10] ETSI TeraFlow SDN. [Online] https://tfs.etsi.org/