

Smart Adaptive Computing Across the Continuum

LLMs in IoT-Edge-Cloud resource management

Antonino Vaccarella

PhD Student · University of Pisa · CNR-ISTI

What this talk is about

Wang and colleagues recently surveyed more than one hundred Deep Reinforcement Learning (DRL) systems for the Cloud Continuum. None of them uses a Large Language Model (LLM).

In the last two years, several papers have started to combine LLMs with DRL — but the literature has no shared way to describe what kind of integration each paper actually does.

I propose a third axis for the taxonomy. It splits the new literature into four levels, and it makes the open research cell visible.

The setting — IoT-Edge-Cloud resource management

The Continuum is a layered infrastructure:

- IoT devices generate data and run small tasks
- Edge nodes do near-real-time processing
- Cloud data centres handle heavy computation and storage

Resource management decides, at every moment:

- where each task runs
- which container scales up or down
- how traffic is routed between the tiers

These decisions must respect Service Level Objectives (SLOs) — for latency, availability, energy, and cost.

Why DRL is a natural fit

The Continuum is highly dynamic.

Workloads, network conditions, and user demand change continuously. Hand-tuned policies cannot keep up.

Deep Reinforcement Learning offers:

- online adaptation from experience
- control under uncertainty
- a principled way to optimise long-term objectives

This is why over a hundred recent papers use DRL here.

Wang et al. organise this literature along two axes

Axis 1 — Control scope

Single-agent (SARL): one global policy

Multi-agent (MARL): several agents, independent or jointly trained

Axis 2 — Training paradigm

Standard: full access to training data

Federated: data stays local; only models or signals are exchanged

	Standard	Federated
SARL	>50 papers	~12 works
MARL	>30 papers	~8 works

the Wang et al. corpus

Every paper in the survey fits one of the four cells. None of them uses an LLM.

Wang et al. close their survey with an open challenge

LLMs are starting to appear next to DRL in network and resource-management papers. The way they are integrated is not consistent across works:

- In some papers the LLM picks the action
- In others it gives high-level goals and lets agents act
- In others it stays in the loop and reacts to feedback

Can we organise LLM-augmented DRL systems along a single structural axis, and use it to see which architectures are already explored, and which are not?

Two simple yes/no criteria are enough to do this.

Two yes/no criteria

Criterion 1 — Delegation

Does the LLM execute actions itself, or does it delegate to autonomous agents?

Criterion 2 — Feedback

Do those agents report execution outcomes back to the LLM, closing the loop?

Closing the loop means the LLM receives structured feedback that has passed through the agent layer — making the entire agentic system, LLM and downstream agents alike, the feedback-aware entity.

The agents translate environment state into a form the LLM can interpret, and the LLM uses that feedback to generate its next prompt or directive.

This is distinct from any feedback loop internal to the agents themselves.

Two binary criteria → four levels.

Four levels of LLM augmentation

L0

DRL only

No LLM. The entire Wang et al. corpus sits here

L1

Direct LLM actor

The LLM picks the action itself. No agents. Open loop.
Note: a closed loop is possible in principle at L1, but no paper in this talk demonstrates one.

L2

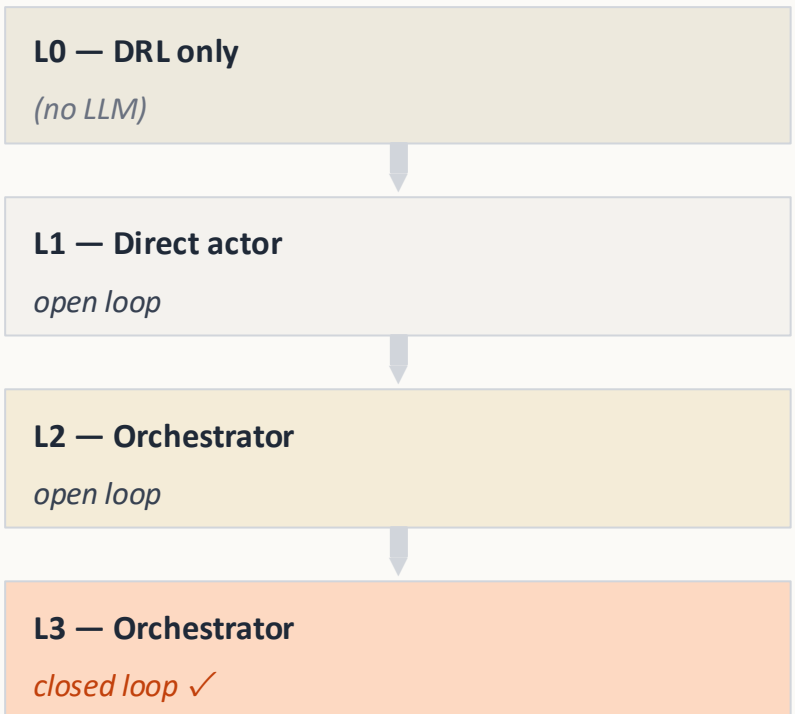
LLM orchestrator

The LLM delegates to agents. No feedback

L3

Closed-loop LLM orchestrator

The LLM delegates, agents act, and report back. Closed loop



The taxonomy becomes 2 × 2 × 4

	<i>scope</i>	<i>Standard</i>	<i>Federated</i>
L0 DRL only	SA	>50 papers	~12 works
	MA	>30 papers	~8 works
L1 Direct LLM actor	SA	IntentContinuum	—
	MA	—	—
L2 LLM orchestrator	SA	—	—
	MA	SALLMA · Habib · AgentEdge	—
L3 Closed-loop LLM orchestrator	SA	N/A — single-agent closed loop has no architectural rationale	N/A — single-agent closed loop has no architectural rationale
	MA	AURA · CyberOps-Bots	<i>open</i>

For each cell: papers that exist, and cells that are empty.
Cells marked N/A are structurally incoherent — a single-agent closed loop offers no coordination benefit.

L1 — IntentContinuum

Domain

Kubernetes-managed edge-cloud cluster, Industry 4.0 image processing

Architecture

GPT-4o reads cluster metrics plus a user-defined latency intent

GPT-4o picks a corrective action from a fixed template (placement, scaling, SDN routing)

A standard Management and Orchestration (MANO) / Software-Defined Networking (SDN) stack applies the action

Nothing is reported back to GPT-4o.

Reported result

85% intent satisfaction, against 43–82.5% for tuned Horizontal Pod Autoscaler (HPA) baselines

Cell: L1 × SARL × Standard.

L2 — Three papers that delegate without feedback

SALLMA — general LLM-MAS architecture

The LLM produces an intent and a workflow; sub-agents execute it

Habib et al. — 5G Open RAN intent automation

BERT classifies the operator intent. A hierarchical deep Q-network (DQN) orchestrates five RAN apps. The LLM exits once the goal is handed off

AgentEdge — edge-cloud service orchestration

Four specialised LLM-agents plus an offline Critic-LLM. The most architecturally mature L2 paper — next slide

All three: open loop. Cell: L2 × MARL × Standard.

AgentEdge — the most mature L2 system in our domain

Domain

Edge-cloud service orchestration — container placement, SLO management, traffic routing across a heterogeneous continuum

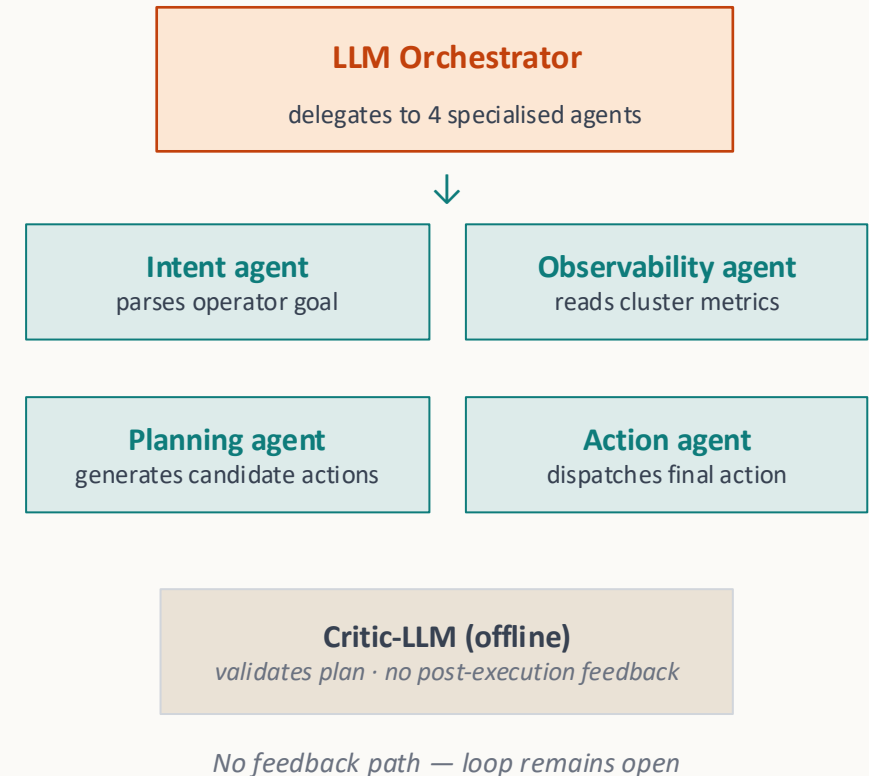
Architecture

Four specialised LLM-agents, each responsible for one stage of the orchestration pipeline

An offline Critic-LLM validates the plan before execution and rejects unsafe proposals

Result: 78.3% orchestration success vs 21.7% for a single-agent baseline

Loop remains open.



L3 — AURA: what changes when you close the loop

Domain

NextG cellular network. Problem: handover management — deciding when and where to hand a mobile user between base stations, in real time.

Architecture

A multimodal LLM reads traffic logs, location data, and GIS maps, converted into language-aligned tokens.

MARL agents at each base station execute and report back.

Trust mechanism: agents weight LLM advice by past benefit.

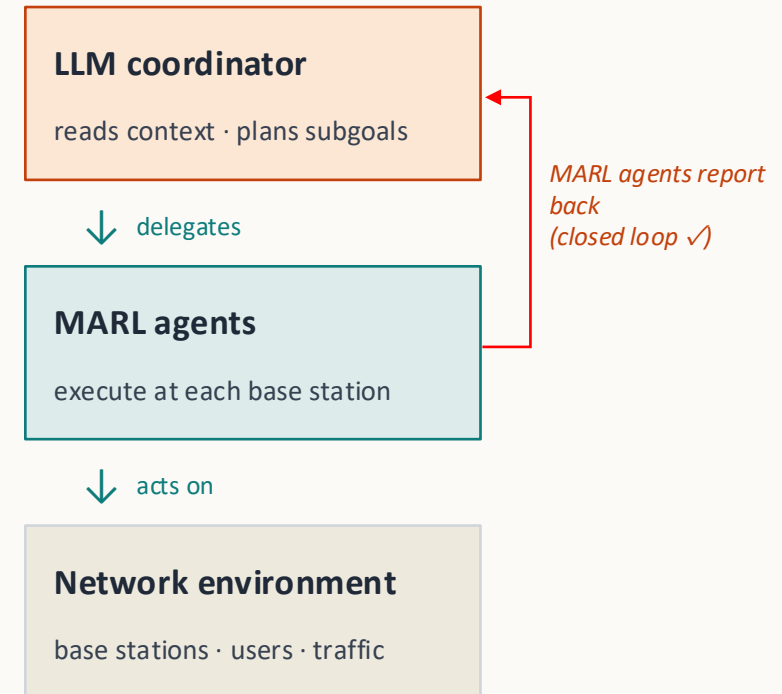
Low trust reduces LLM influence; it does not silence it.

After each cycle, agents report their actions, local conditions, and outcomes back to the LLM.

The LLM updates its understanding before the next planning round.

The structural limitation is scope: AURA operates on two base stations in a cellular network.

No multi-tier heterogeneity, no containerised workload placement, no SLO management.



Trust mechanism: agents weight LLM advice by past benefit

Batched feedback closes the loop.

Result: –50% dropped handoffs vs MARL alone.

L3 — CyberOps-Bots: closed-loop cloud cyber-defence

Domain

Cloud cyber-defence. Problem: respond autonomously to Multi-stage attacks (lateral movement, ransomware, DDoS) guided by the MITRE ATT&CK framework. Goal: adapt without retraining as the network changes

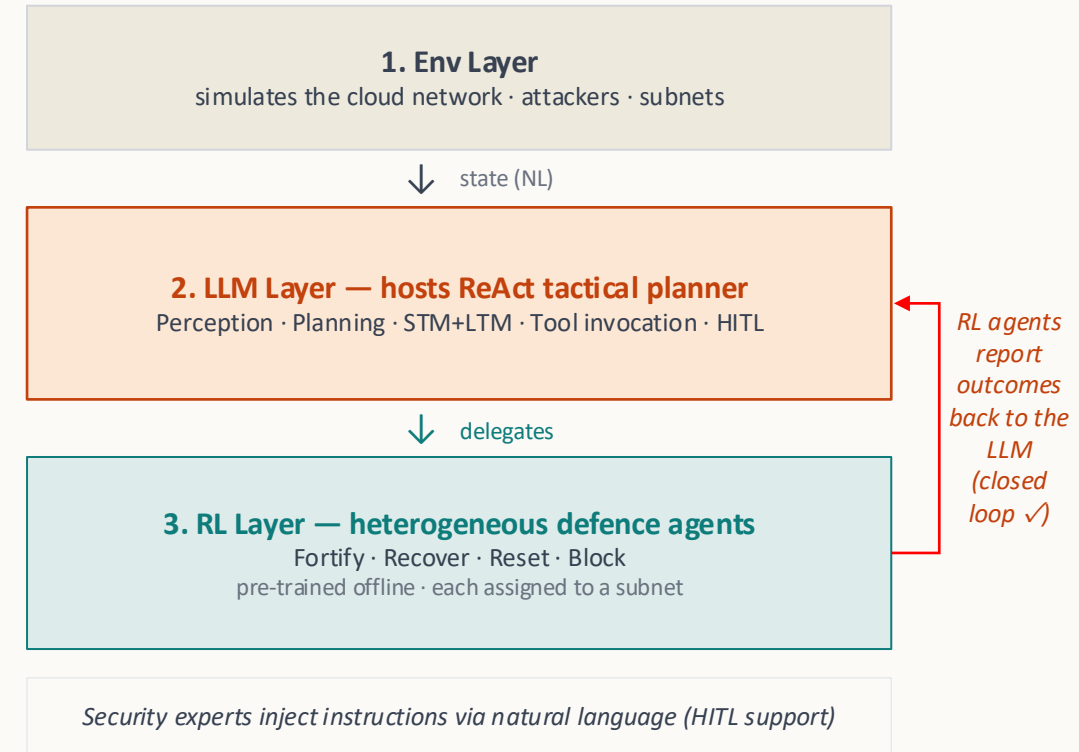
Architecture

Three-layer hierarchy: Env Layer / LLM Layer / RL Layer

LLM uses ReAct (Thought-Action-Observation) with short-term and long-term memory.

Four specialised RL agents (Fortify, Recover, Reset, Block) execute and *report outcomes back. LLM revises the plan at each decision step.*

Result: +68.5% network availability; +34.7% under shifting attack strategies (no retraining).



Any architecture for the L3 cells must answer two structural questions: how many LLMs, and how specialised are the agents?

Dimension A: LLM cardinality

A1 — Single centralised LLM

One Large Language Model. Receives aggregated performance signals from all agents.
Returns subgoals or reward-shaping parameters.
Minimum coordination overhead — easiest to validate experimentally.

A2 — Hierarchical LLM + per-tier SLMs

A large coordinator at the top handles cross-domain planning.
Small language models deployed locally per tier reason over sensitive data and expose only anonymised KPIs upward.
Relevant when agents handle private usage patterns or proprietary traffic profiles.

A3 — Federated per-domain LLMs

Each administrative domain has its own model.
Only abstract performance representations are shared — no raw observations, no model weights cross domain boundaries.
Fully addresses data sovereignty. Requires a co-designed aggregation protocol. Open problem.

A2 and A3 are not mutually exclusive: a system could use per-tier SLMs locally (A2) within each domain while federating the domain-level coordinators (A3). Their combination is an open research direction.

Dimension B: Agent specialisation

B0 — Homogeneous *(excluded)*

All agents share one policy and one action vocabulary.

Unrealistic for the Cloud Continuum: the scale difference between an IoT microcontroller and a cloud cluster makes a single policy either over-engineered in small settings or under-engineered in large ones.

B1 — Tier-specialised

A distinct policy per tier — IoT, edge, cloud — each with its own action vocabulary.

During training, a centralised critic sees the full system state; at deployment, each agent acts on local observations only.

A natural fit for the Continuum.

B2 — Function-specialised

Agents differentiated by management role: scheduling, placement, routing, privacy.

A privacy-sensitive agent can incorporate a local SLM (A2 pattern) internally. B2 + A2 is the most expressive combination.

B1 and B2 are not mutually exclusive — their combination (tier-specialised agents that are also function-differentiated) is a potential future direction. Adding B3 (role-specialised per SLO type, e.g. latency vs. energy) is also an open possibility.

What the 16-cell map tells us

Four observations.

1. Most LLM-augmented systems are open-loop. Four out of six analyzed papers act once and exit
2. The two closed-loop papers are off-domain. There are no results in the literature demonstrating a control loop closed toward the LLM in the Cloud Continuum.
3. The $L3 \times MA \times Federated$ cell is empty
4. AgentEdge and CyberOps-Bots already discuss federated extensions as future work

Primary open cell: $L3 \times MA \times Standard$ — Cloud Continuum (no paper).
Secondary open cell: $L3 \times MA \times Federated$ — completely empty.

	Standard	Federated
L0	~50 / ~30	~12 / ~8
L1	1 paper	—
L2	3 papers	—
L3	2 papers (off-domain)	open ←

*$L3 \times Federated$ —
secondary open cell*

Open question: what does it even mean to close the loop in a federated setting, where agents never share raw data and models stay local?

Three open engineering challenges in this space

1. Latency mismatch

LLM inference takes seconds to minutes. MARL acts at sub-second granularity. The interface between them must absorb this gap

2. Cost of LLM involvement

Calling the LLM at every decision step is too expensive.

How sparsely can the loop be closed while preserving its coordination benefit?

3. Training stability under non-stationary reward shaping

If the LLM rewrites reward signals online, the MARL training target moves — this is non-stationary reward shaping.

Standard convergence guarantees no longer hold.

A potential development plan

Goal

A closed-loop LLM coordinator for tier-heterogeneous MARL across IoT, edge, and cloud nodes, under SLO constraints

Starting configuration – Direction 1

A1+B1 as entry point: one centralised LLM + tier-specialised agents.

The structurally simplest closed-loop configuration.

Resolves the baseline question: does closing the loop produce measurable benefit over open-loop L2 systems in the Cloud Continuum?

Possible directions

Direction 2 — A2+B2: per-tier SLMs handle sensitive local data; function-specialised agents manage distinct roles.

Resolves the data locality problem: private traffic patterns never leave the tier. Most expressive combination.

Direction 3 — federated variant: each administrative domain has its own LLM coordinator; only abstract performance signals cross domain boundaries.

Resolves data sovereignty constraints. The definition of “closing the loop” in this setting is itself an open problem.

Summary

1. The literature combining LLMs with DRL has grown fast, but it has no shared vocabulary
2. Two yes/no criteria — delegation and feedback — give four levels (L0, L1, L2, L3) and a 16-cell map
3. The Cloud Continuum L3 cell is empty. This is the primary open research target

Thank you