AEIT International 2026 Annual Conference
118th edition
Roma ♦ Italy ♦ 8/10 September 2026
Sapienza ♦ University of Rome

Learning-to-Optimize as the Missing Architectural Layer of AI-Native Networks

Giambattista Amati, Federica Mangiatordi, Pierpaolo Salvo, Emiliano Pallotti, Simone Angelini

Fondazione Ugo Bordoni
Rome, Italy

FUB Fondazione Ugo Bordoni
Research, Innovation, Strategy

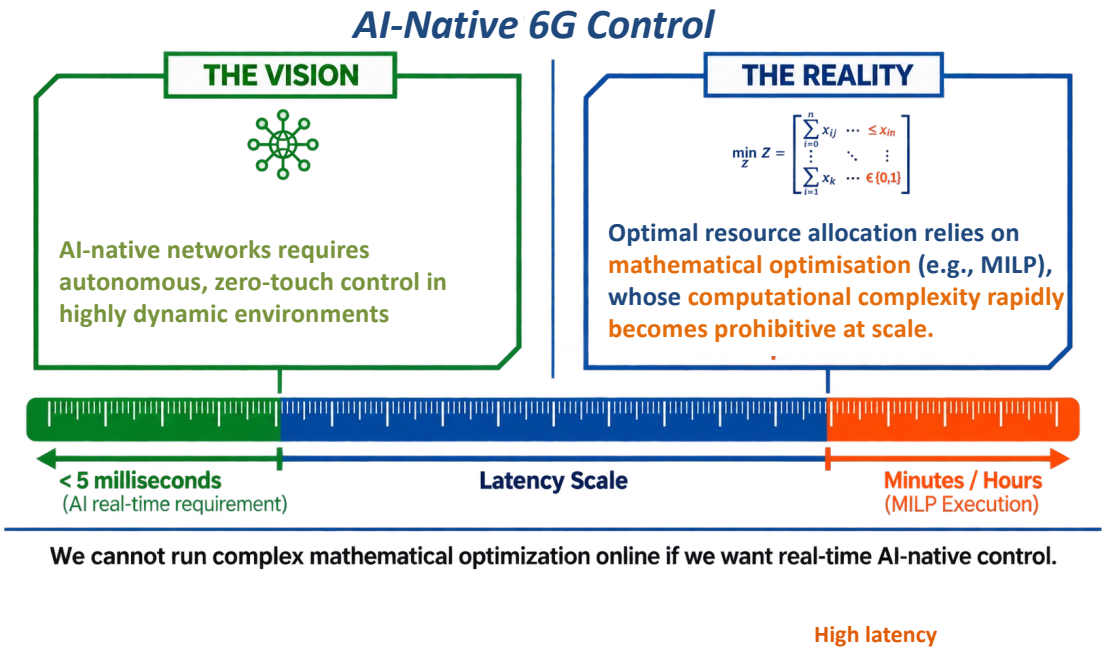
Motivation: AI-Native Networks Need Real-Time Intelligence

From AI-Assisted to AI-Native Networks

- AI as **intrinsic component** of future 6G networks.
- However, many network functions are still formulated as **complex optimisation problems**.

Core question

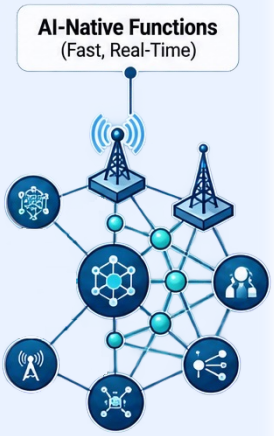
How can we preserve optimisation quality while enabling millisecond-level decisions?



The Architectural Gap

AI-Native Architectures

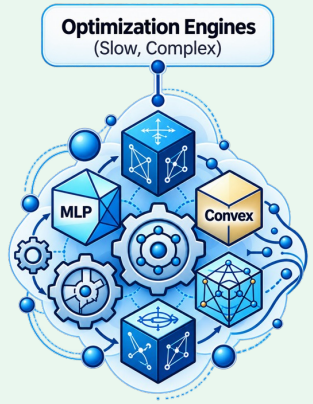
- ✓ Manage AI models
- ✓ Support training and inference
- ✓ Monitor, retrain and update models
- ✓ Assume models already exist



Missing bridge

Learning-to-Optimise

- ✓ Learns from optimisation solutions
- ✓ Builds surrogate models
- ✓ Accelerates expensive algorithms
- ✓ Usually treated as an algorithmic pipeline



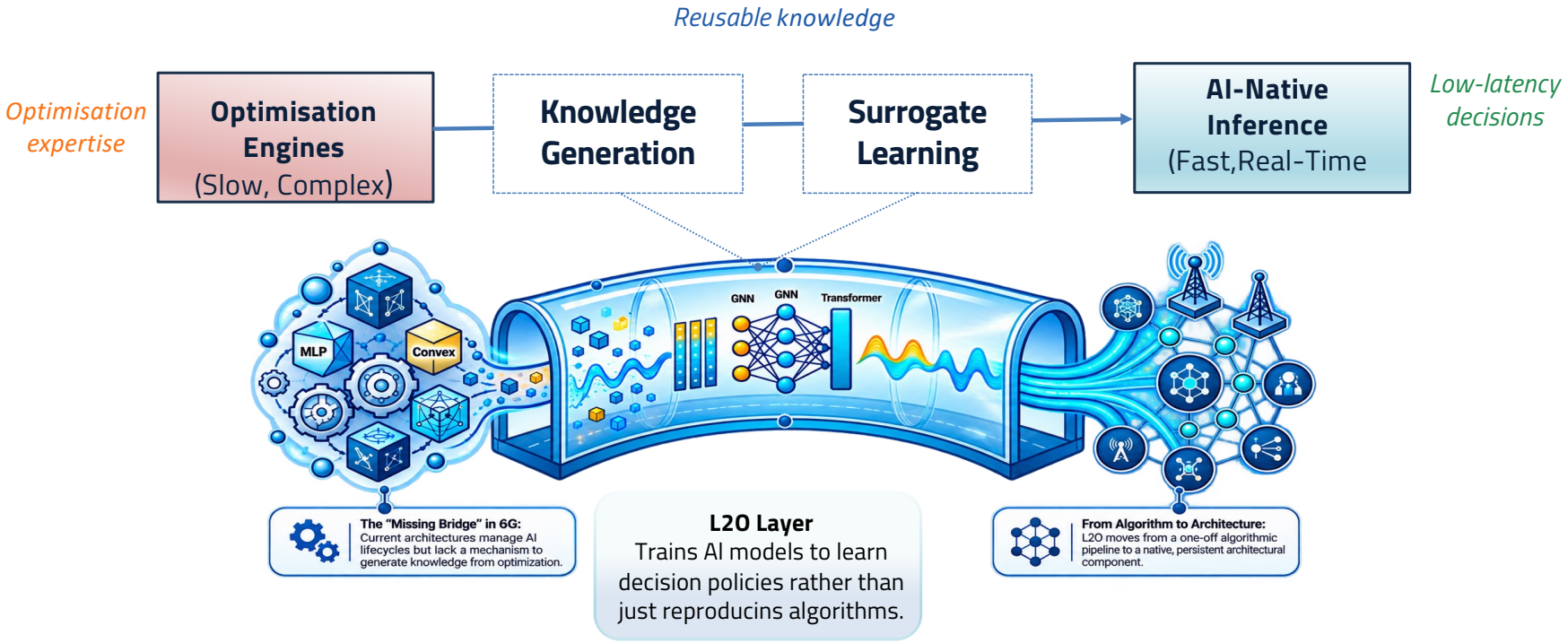
Traditional AI Lifecycle
Focused on deployment, retraining and monitoring of pre-existing models

Structural Disconnection
No architectural component currently links optimization engines with AI lifecycles to systematically create, reuse and evolve knowledge.

Traditional L2O Focus
Focused on accelerating individual algorithms, not system architecture

Learning-to-Optimise as an Architectural Layer

Instead of using optimisation only as an online decision engine, we use it as an offline knowledge generator.

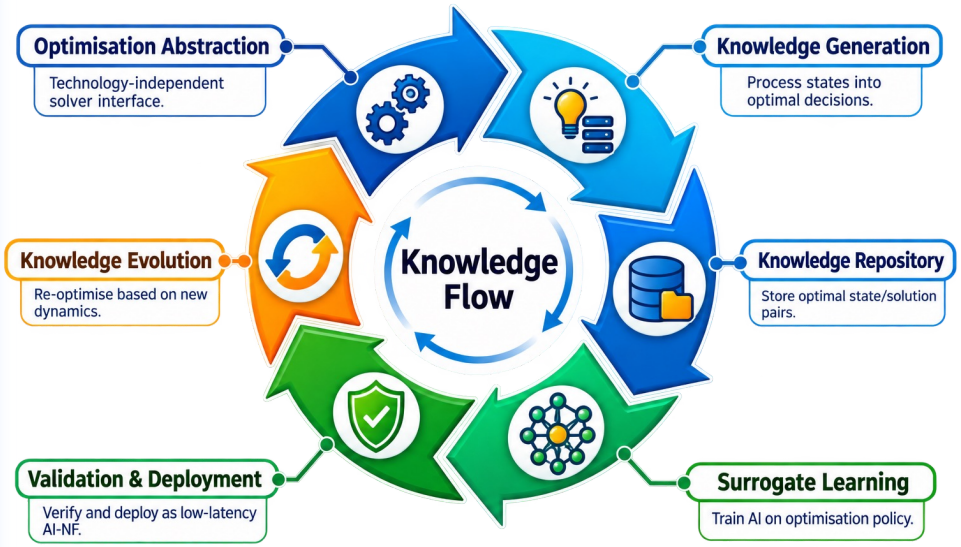


The L2O Layer acts as a persistent architectural knowledge plane, transforming optimisation expertise into reusable AI models for real-time network control.

Functional Components of the L2O Layer

The L2O Layer decomposes the knowledge lifecycle into structured architectural components:

1. **Optimization Abstraction Function:** Provides a technology-independent interface to external solvers (MILP, convex, heuristics, etc.).
2. **Knowledge Generation Function:** Feeds representative network states to solvers to generate optimal decision labels covering target states.
3. **Knowledge Repository Function:** Persistently stores states, metadata, objective functions, constraints and solver labels as a reusable asset.
4. **Surrogate Learning Function:** Trains surrogate models (GNN, DNN, Transformers) on the repository datasets to map states to decisions.
5. **Model Validation & Deployment Function:** Validates accuracy, constraint satisfaction and out-of-distribution robustness before pushing models as AI-native functions (AI-NF).
6. **Knowledge Evolution Function:** Monitors performance drops and drift to trigger new optimization campaigns, continuously enriching the repository.



Knowledge Asset

Optimization results are no longer discarded after one-off use; they become structured, historical datasets in the persistent repository.

Closed-Loop Evolution

As network topologies or traffic statistics drift, new data is collected, solved offline, and the repository is enriched.

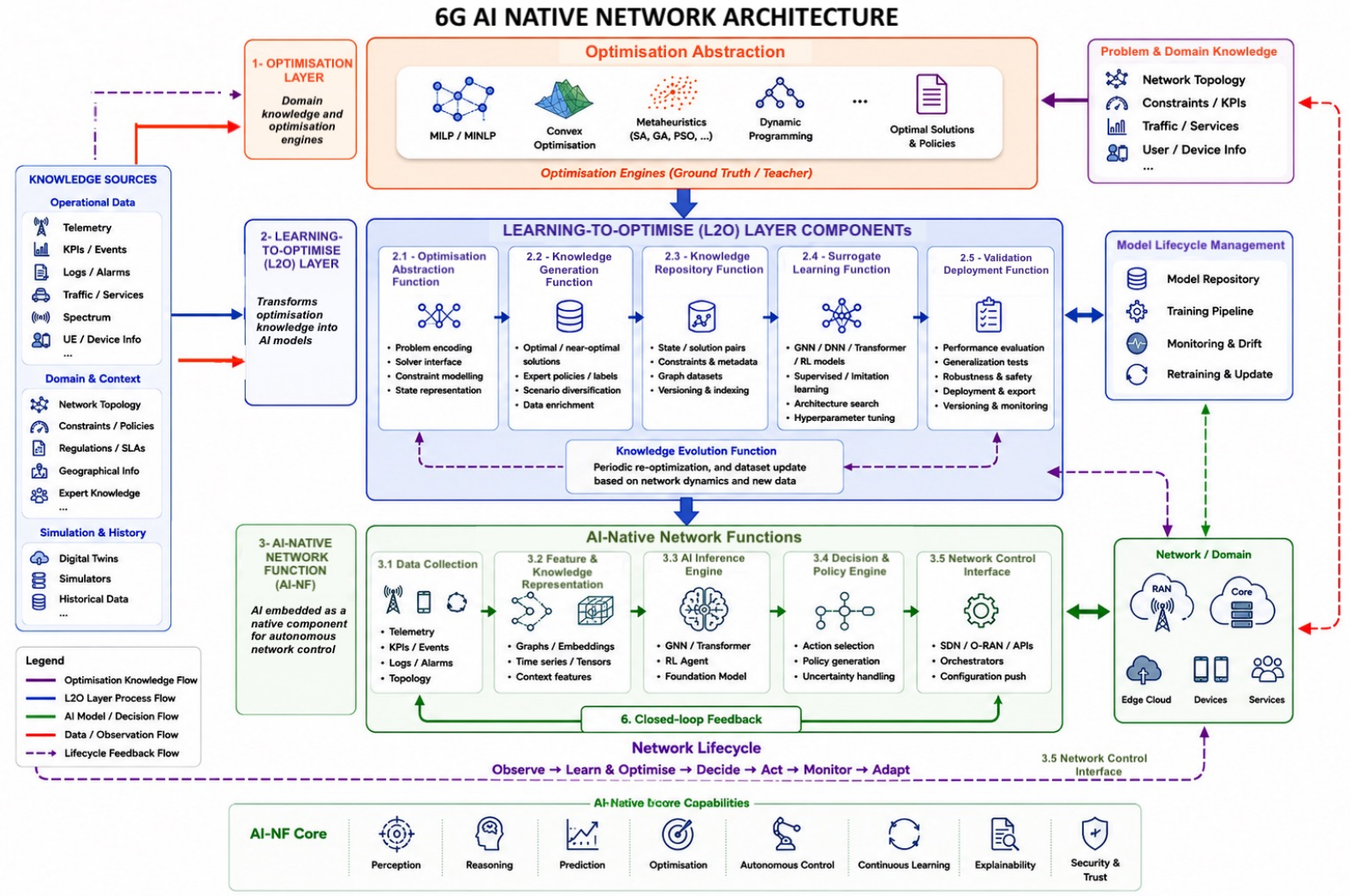
Architectural Comparison

Beyond Conventional Learning-to-Optimise

\PABILITY	AI- NATIVE ARCHITECTURES	CONVENTIONAL L2O	PROPOSED L2O Layer
Main Goal	AI lifecycle management	Optimisation acceleration	**Optimisation knowledge lifecycle**
Optimisation Role	Runtime support	Oracle	**Persistent knowledge generator**
Knowledge	Model repository	Optimisation datasets	**Optimisation knowledge repository**
Lifecycle	Deploy & retrain	Offline learning	**Generate → Learn → Deploy → Evolve**
Architectural Role	AI management	Algorithmic workflow	**Native architectural component**

The main novelty is architectural, not only algorithmic.

L2O Layer Architectural Integration



- ✓ The L2O Layer is positioned between optimisation engines and AI-native network functions.
- ✓ It manages optimisation-derived knowledge as a persistent network asset.
- ✓ It supports model training, validation, deployment and continuous evolution.

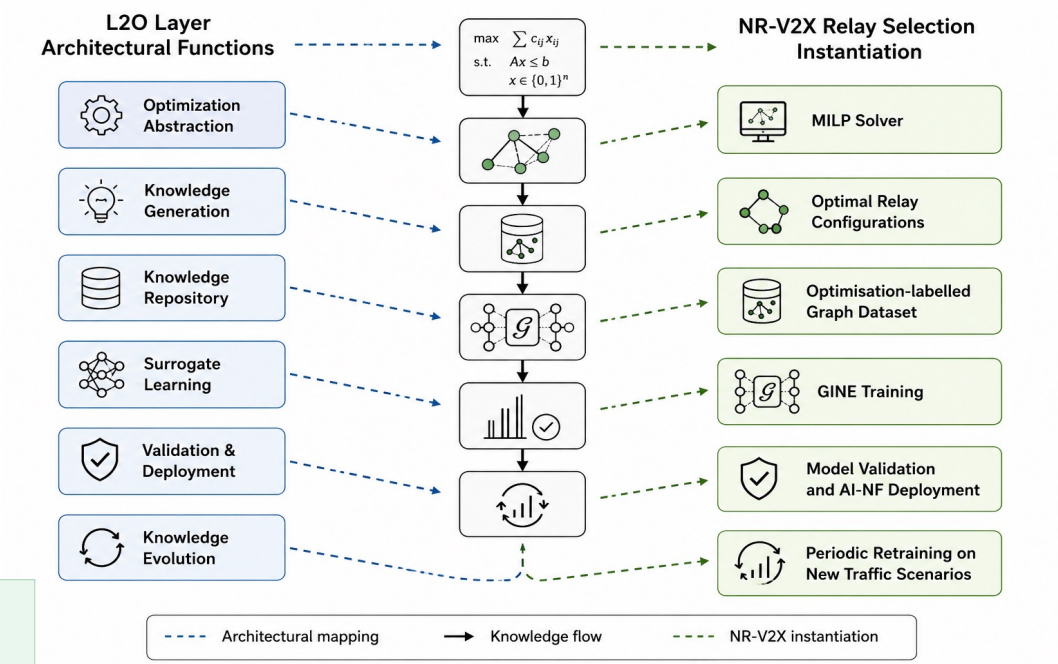
AEIT 2026

FUB Fondazione Ugo Bordoni Research, Innovation, Strategy

NR-V2X Case Study

Dense urban vehicular networks require reliable relay selection under fast topology changes.

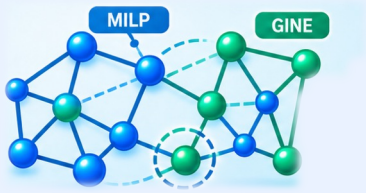
- ✓ Dynamic topology and NLOS may prevent direct RSU connectivity.
- ✓ Relay selection is formulated as a constrained optimisation problem.
- ✓ The L2O Layer turns MILP-generated relay configurations into graph-based training knowledge.



MILP	GINE	AI-NF
optimisation oracle	surrogate model	runtime decision

Experimental Evidence

Can the surrogate reproduce the optimisation oracle?



≈96%

link-level agreement with the MILP decisions

<5 ms

inference latency for all evaluated graph instances

- ✓ The GINE reproduces approximately 96% of the MILP link-level relay decisions, while inference remains below few milliseconds.
- ✓ The solver is removed from the runtime control loop.
- ✓ In hybrid optimisation, the surrogate can prune the MILP search space.

Offline optimisation expertise → real-time network intelligence

Conclusions

Learning-to-Optimise should move from an algorithmic tool to a native architectural component.

- ✓ We introduced an L2O Layer for AI-native networks.
- ✓ We defined a lifecycle for optimisation-derived knowledge.
- ✓ We instantiated the approach through NR-V2X relay selection using MILP and GINE.
- ✓ The same principles apply to routing, resource allocation, spectrum coordination, topology optimisation and cloud-edge orchestration.

Take-away message

Optimisation should not only solve network problems.
It should generate knowledge that AI-native networks can learn, reuse and continuously evolve.

Constraint-aware learning | **Continual evolution**

Digital Twin-assisted optimisation

Thank you
for your attention!