Project GAINS

(Generative AI for Network Sustainability)

Deliverable D2.2

Technical Report on Agent Integration and Proof-of-Concept Evaluation

Giordano, A. – Basile, L. – De Giacomo, A. – Haider, B. – Fruncillo, D. – McGann, O.

2026-01

Work Package: WP2 System Design, Agent Integration and Proof-of-Concept Validation

Status: Final | Version: 1.0

Classification: internal project use; controlled sharing with partners as needed.

Abstract

This Technical Report documents the end-to-end integration of GAINS system agents and proof-of-concept (PoC) evaluation as a complete automation flow and decision support in an emulated telco/cloud context, with extensions on real signals in a controlled segment of a real telco network directly connected to the research environment built in the datacenter. The aim is to demonstrate that the interaction between Network Monitor, Knowledge Manager and Network Orchestrator produces outputs that are operationally usable, verifiable and consistent with technical constraints, while maintaining operational efficiency, scalability and minimal governance safeguards.
The scope of validation includes functional and stress tests on model scenarios, verification of the correctness and consistency of the artifacts produced, configurations, change and rollback plans, validation reports, explainability packages, and evaluation of the accuracy and repeatability of environmental analyses, all anchored in a canonical data model and versioned API contracts.
The PoC is built to be replicable in a Docker environment and to be validated by a third-party team without privileged access to proprietary infrastructures: the runtime components are containerized, dependencies are made explicit and configured through environment variables and secret management, and the entire flow is orchestrated by an API-first Control Plane which enforces RBAC, end-to-end traceability and an append-only audit trail.
Validation is designed to be "black-box by contract": the user verifies behavior and quality by observing only the published endpoints, the canonical payloads and persisted artifacts, without the need to modify the code. This approach reflects a premise of industrial transferability: individual agents can evolve or be replaced, as long as contracts and the canonical model remain invariant and the chain (trace_id, job, artifacts, audit, export manifest) remains reconstructable.
Integration is based on three distinct and intentionally decoupled operational responsibilities.
(i) The Network Monitor operates in read-only mode and transforms telemetry and history into verifiable evidence, deterministic baselines, anomalies and structured indicators. In the PoC, the source of observability is anchored to an NMS already in stock, so as to accelerate the end-to-end without altering architectural invariants and contracts. (ii) The Knowledge Manager implements a knowledge operation-oriented plane, in which documents, policy and artifacts are loaded in the database with deterministic pipelines and return queries with hybrid retrieval and applicability-binding filters. The

operational responses are grounded and accompanied by verifiable quotes, with fail-closed mechanisms or controlled degradation when the evidence is insufficient. (iii) The Network Orchestrator translates an intent into candidate configurations contextualized and generates a job governed by explicit states, producing versioned and hashed artifacts and introducing human-in-the-loop gating for approval or rejection in operational risk scenarios.

Sustainability is not treated as a retrospective metric, but as a canonical item of the workflow. The PoC validates the production and use of two key objects: Footprint Inputs, generated from the Monitor as a proxy load attributable by device and combinable for PoP, and Footprint Estimate, generated in orchestration as a comparison of as-is/to-be/delta with declared method (metered, model-based, hybrid), consistent time basis, explicit assumptions and confidence level. The end-to-end verification checks that estimates degrade in a deterministic and auditable way when signals or dependencies are missing, and that each output exhibits references to the evidence used, making the chain verifiable like the technical chain. The rating of the PoC is structured in three complementary levels.

The first level is contractual compliance: verification that every critical endpoint respect schema, mandatory fields, minimum requirements, standard error model and propagation of the trace_id, guaranteeing consistency between canonical payloads and persistent artifacts.

The second level is functional and scenario-based: on a set of emulated telco/cloud scenarios, it is measured whether the agents cooperate correctly in producing outputs that are consistent, applicable and verifiable, checking in detail (i) applicability of knowledge retrieval with respect to vendor/role/version/service, (ii) presence of robust quotes and absence of unsupported claims in operating mode, (iii) structural fairness and verifiability of change and rollback, (iv) consistency between generated documentation and delivered technical artifacts. The third level is robustness and stress: it exercises concurrency, timeouts, bounded retries, controlled degradation in case of unavailable dependencies and absence of leakage in logs and outputs. The evidence includes parallel jobs, idempotency controls and validations of exportability, immutability and integrity of artifacts.

The deliverable consolidates the experimental results of integration and rating: (i) a PoC environment replicable in Docker in which the three agents interact through canonical contracts and external OpenAI-compatible configurable endpoints, (ii) a test and stress harness that enables automated validation and evidence harvesting with structured reporting, (iii) an export manifest that formalizes inventory and integrity of the evidence package through checksums and environment metadata, (iv) a set of metrics and quality gates to measure retrieval quality and grounding, quality of operational artifacts and reliability of the chain, (v) a synthesis of lessons learned focused on determinism, risk minimization and reduction of ambiguities between intent, policy and technical outputs.

Finally, the validation is designed to be usable also by non-specialist users through an optional demo UI and qualitative stress, translating conversational interactions into calls towards the Control Plane, without becoming a structural dependency of the system and remaining confined to supporting experimentation.

Table of Contents

5. Integration of agents

    5.1 Network Monitor integration: inputs, normalization, structured outputs

    5.2 Knowledge Manager integration: ingest, retrieval, citations, policy gating

    5.3 Network Orchestrator integration: intent-to-plan, job state machine, artifacts

    5.4 End-to-end flow orchestration and dependency management

    5.5 Idempotency, retry bounded, timeout, rate limit and backpressure

    5.6 Human-in-the-loop and approval checks

6. Model Data canonical and Contracts API for the Validation

    6.1 Catalogue of canonical objects and relationships

    6.2 Payload Schemas and Required Fields

    6.3 Validation rules and version compatibility

    6.4 Standardized error model and semantics of error codes

    6.5 End-to-end traceability

    6.6 Export manifest and package of evidence

7. Scenarios Telco Model/Cloud for The Rating

    7.1 Selection criteria and functional coverage

    7.2 Scenario S1: Anomaly detection, diagnosis, operational recommendation

    7.3 Scenario S2: Change on service, generation of config data, validation and rollback plan

    7.4 Scenario S3: Verification of compliance, constraints and standards, generation of technical documentation

    7.5 Scenario S4: Optimization with sustainability constraints and as-is/to-be comparison

    7.6 Scenario S5: Controlled degradation and fail-closed in the absence of evidence

    7.7 Synthetic datasets, seeds and reproducibility of scenario

8. Plan of testing end-to-end

    8.1 Testing Objectives and Overall Strategy

    8.2 Test Matrix

    8.3 Acceptance criteria and quality gates

    8.4 Test & Automation Harness

    8.5 Collection of evidence, audit trail and completeness criteria

    8.6 Non-compliance management, issue taxonomy and triage

9. Testing of stress, Scalability e Reliability

    9.1 Load model and workload profile

    9.2 Concurrency, throughput, latency, and resource saturation

    9.3 Soak test and stability over time

    9.4 Fault injection and resiliency

    9.5 Idempotency and Consistency of Artifacts Under Stress

    9.6 Results and Interpretation of Robustness Metrics

10. Verification of the Fairness of the Configurations e operational safety

    10.1 Syntactic and semantic validations

    10.2 Change plan, rollback plan and operating window consistency

    10.3 Guardrails and policy enforcement

    10.4 Operational Risk Minimization and Access Controls

    10.5 Secret management and reduction of leakage in logs and artifacts

11. Rating of the quality of the Knowledge Plane

    11.1 Grounding and coverage of citations

    11.2 Retrieval metrics and evidence quality

    11.3 Robustness to adversarial prompts and operational hallucinations

    11.4 Fail-closed behavior and uncertainty management

    11.5 Consistency between generated documentation and technical artifacts