HEXA-X-II

A holistic flagship towards the 6G network platform and system, to inspire digital transformation, for the world to act together in meeting needs in society and ecosystems with novel 6G services

Deliverable D6.3
Initial Design of 6G Smart Network Management Framework

Co-funded by the European Union

6GSNS

Hexa-X-II project has received funding from the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101095759.

Date of delivery:	07/11/2025	Version:	1.1
Project reference:	101095759	Call:	HORIZON-JU-SNS-2022
Start date of project:	01/01/2023	Duration:	30 months

Document properties:

Document Number:	D6.3
Document Title:	Initial Design of 6G Smart Network Management Framework
Editor(s):	G. Landi (NXW)
Authors:	R. Vilalta, R. Muñoz, P. Alemany, Ll. Gifre, C. Manso, B. Ojaghi (CTT), C. Ayimba, A. Calvillo (UC3), G. Landi, P.G. Giardina, M. De Angelis, P. Piscione (NXW), W. Tavernier, J. Miserez (IMEC), S. Rodríguez, X.R. Sousa, F. Lamela (OPT), T. Dimitrovski, N. Toumi (TNO), V. Lamprousi, S. Barmpounakis, P. Demestichas, A. Karaolanis, V. Tsekenis, C. Karousatou (WIN), E. Lluesma Marti, L.F. González (ATO), I. Labrador Pavón (ASA), R. Nicolicchia (TID), I. Tzanettis, Grigorios Kakkavas, and A. Zafeiropoulos (ICC).
Contractual Date of Delivery:	30/06/2024
Dissemination level:	PU
Status:	Public version
Version:	1.1
File Name:	Hexa-X-II_D6.3_v1.1

Revision History

Revision	Date	Issued by	Description
1.0	28.06.2024	Hexa-X-II WP6	First public version
1.1	07.11.2025	Hexa-X-II WP6	Broken links fixed in Section 8.2.3 (Interfaces of the Management Capabilities Exposure framework).

Abstract

This document describes the initial design and implementation of the enablers for *6G Smart Network Management*, contributing to the 2nd iteration of the Hexa-X-II end-to-end system blueprint. The deliverable provides an initial overview of the Smart Network Management framework defined in the project. It analyses its contributions to the Hexa-X-II architecture design principles and lists the technical enablers for the management and orchestration functionalities, highlighting the main enhancements and innovations with respect to the Hexa-X architecture.

The categorization of the enablers has been updated with respect to D6.2, to provide an improved and unified picture of their role and contribution in the Hexa-X-II *Smart Network Management* system. The enablers cover the functionalities and objectives of a 6G network Management & Orchestration (M&O) system, providing the technical foundations for programmable network configuration and monitoring, network capability exposure, synergetic resource orchestration across the continuum, and zero-touch network automation. All these functionalities are supported by advanced techniques for security and trustworthiness, AI/ML algorithms, and network digital twins.

For each enabler, the document illustrates the design and presents the ongoing implementation, providing early validation results and discussing the enablers' contributions to KPIs and KVIs. The role of the enablers in Hexa-X-II PoCs is presented as well, with initial integration and evaluation results.

Finally, as key contribution towards the design of the end-to-end Hexa-X-II system, the document analyses the alignment of the M&O enablers with the end-to-end system blueprint, identifying the mapping with the current components and potential gaps to be filled with new elements or interfaces. This will feed the next iteration of the end-to-end Hexa-X-II system design, under development in WP2, following the overall methodology defined in the project.

Keywords

Smart Network Management, Management and Orchestration, Network Programmability, Monitoring, Telemetry, Network Exposure, Security, Continuum Orchestration, Artificial Intelligence and Machine Learning, Trustworthy AI, Network Digital Twins, Zero-Touch Closed Control Loops

Executive Summary

The Smart Network and Services Joint Undertaking (SNS JU) 6G Flagship project Hexa-X-II leads the way to next generation 6G end-to-end (E2E) system design, based on integrated and interacting technology enablers. The project will continue the track of the previous 5G-PPP Hexa-X project [HEX24], which has laid the foundation for the global communication network of the 2030s by developing the 6G vision and basic concepts. As a continuation of Hexa-X, the interaction between the cyber-physical world and human world will be further evolved with the advancement of information, communication, and computation technologies towards a pervasive human centred cyber-physical world in 2030. To reach this 6G vision, Hexa-X-II has set up the goals to design the system blueprint of a sustainable, inclusive, and trustworthy 6G platform, which will require novel enablers regarding Smart Management & Orchestration (M&O) functionalities.

In this direction, this report is the second public deliverable produced by the Hexa-X-II "Smart Network Management" Work Package, documenting the evolution in the design of the envisioned enablers and their first implementation, with preliminary evaluation results. The deliverable has the primary objective of contributing to the 2nd iteration of the Hexa-X-II end-to-end system blueprint, providing a well-structured and consolidated analysis of the M&O technical enablers defined in the project. An initial overview of the Smart Network Management framework highlights the contributions to the Hexa-X-II architecture design principles and the mapping with 6G stakeholders, explaining the main enhancements and innovations introduced in the Hexa-X-II M&O framework, especially with respect to the Hexa-X architecture.

The categorization of the enablers has been updated with respect to the previous deliverable D6.2, to provide an improved and unified picture of their role and complementary contribution in the Hexa-X-II *Smart Network Management* system. The whole set of enablers covers, jointly, the functionalities and objectives of a 6G network Management & Orchestration (M&O) system. They provide the technical foundations for programmable network configuration and monitoring, network capability exposure, synergetic resource orchestration across a multi-domain continuum, and zero-touch network automation. All these functionalities are supported, in a transversal manner, by advanced techniques for security and trustworthiness, Artificial Intelligence (AI) and Machine Learning (ML) - AI/ML algorithms, and network digital twins. The re-structuring of the M&O enablers classification has identified 8 main enablers, where some of them are organized in additional sub-enablers for specific aspects or functionalities. Moreover, their naming and terminology has been updated to highlight their applicability to concrete elements or systems of the future 6G Smart Network Management system. The updated list of proposed enablers and sub-enablers is the following:

- Enabler 1: Network Programmability Framework
- Enabler 2: Monitoring and Telemetry Framework
- Enabler 3: Management Capabilities Exposure Framework
- Enabler 4: Security and trustworthiness Framework
  - Sub-enabler 4.1: Resource controllability for 3rd parties
  - Sub-enabler 4.2: User-centric service provisioning
  - Sub-enabler 4.3: Trust Management System
- Enabler 5: Synergetic Orchestration Mechanisms for the Computing Continuum
  - Sub-enabler 5.1: Multi-agent systems for multi-cluster orchestration
  - Sub-enabler 5.2: Decentralised orchestration system
  - Sub-enabler 5.3: Federated orchestration system
- Enabler 6: AI/ML Algorithms
  - Sub-enabler 6.1: AI/ML-based control algorithms for sustainability
  - Sub-enabler 6.2: Trustworthy AI/ML-based control algorithms
- Enabler 7: Network Digital Twins Creation Mechanisms
- Enabler 8: Real-time Zero-touch Control Loops Automation and Coordination System

For each enabler, the document presents: (i) the technical approach; (ii) the ongoing implementation; (iii) the early validation results, and, (iv) their impact to relevant Key Performance Indicators (KPI) and Key Value Indicators (KVI). At this stage of the activities, most of the M&O enablers have been contextualized and

integrated in software prototypes within two Hexa-X-II Proof of Concepts (PoC): PoC#A.1 and PoC#B.1. These allow to validate and demonstrate the feasibility of the conceptual ideas with concrete scenarios, well-defined workflows and testing environments.

Finally, as key contribution towards the design of the end-to-end Hexa-X-II system, the document analyses the alignment of the M&O enablers with the end-to-end system blueprint, identifying the mapping with the current components and potential gaps to be filled with new elements or interfaces. This will feed the next iteration of the end-to-end Hexa-X-II system design, under development in WP2, following the overall methodology defined in the project. In this direction, the document constitutes an intermediate output towards the design and implementation of Hexa-X-II Smart Network Management system, with preliminary software prototypes, early validation results and initial integration in Hexa-X-II PoCs. The next deliverable D6.4 will further consolidate these assets, providing the final and fully validated version of the M&O system enablers.

Table of Contents

1 Introduction 21
1.1 Objective of the document 21
1.2 Structure of the document 21

2 Smart Network Management 23
2.1 M&O technical enablers 26
2.2 Mapping with Hexa-X-II architecture design principles 26
2.3 Mapping with the 6G stakeholders 31
2.4 Innovations in Hexa-X-II M&O framework 34
2.4.1 Enhancements to the Hexa-X architecture 38

3 Enablers for Management and Orchestration 41
3.1 Enabler 1: Network programmability framework 41
3.1.1 Enabler design 41
3.1.1.1 Internal Architecture of the system components 42
3.1.1.2 Workflows 43
3.1.2 Preliminary implementation and early validation results 44
3.1.2.1 TeraFlowSDN using data-plane in-a-box 44
3.1.2.2 SmartNIC Transceiver support using OpenConfig extensions 45
3.1.2.3 Time Sensitive Networking and Deterministic Networking SDN controller 47
3.1.2.4 Automated transport network re-configuration 49
3.1.2.5 MEC Bandwidth Management service integration with SDN controller 51
3.1.2.6 Integration of TM Forum APIs with ETSI TFS NBI 53
3.1.3 Impacted KPIs and KVIs 55
3.2 Enabler 2: Monitoring and telemetry framework 55
3.2.1 Enabler design 56
3.2.1.1 System components 56
3.2.1.2 Workflows 57
3.2.2 Preliminary implementation and early validation results 58
3.2.2.1 TeraFlowSDN event-driven monitoring 58
3.2.2.2 Energy monitoring 60
3.2.2.3 Monitoring platform for integration in closed loop 62
3.2.3 Conceptual solutions 64
3.2.3.1 Passive/in-band and active telemetry for TSN/DetNet networks 64
3.2.3.2 Data fusion for signals correlation and remediation actions 65
3.2.4 Impacted KPIs and KVIs 66
3.3 Enabler 3: Management capabilities exposure framework 67
3.3.1 Enabler design 67
3.3.1.1 System components 68
3.3.1.2 Internal Architecture of the system components 69
3.3.1.3 Workflows 72
3.3.2 Preliminary implementation and early validation results 73
3.3.2.1 Apache Kafka cluster structure 73
3.3.3 Impacted KPIs and KVIs 74
3.4 Enabler 4: Security and trustworthiness framework 75
3.4.1 Sub-enabler 4.1: 3rd party resource control separation enabler 76
3.4.1.1 Sub-enabler design 76
3.4.1.2 Impacted KPIs and KVIs 82
3.4.2 Sub-enabler 4.2: User-centric service provisioning system 82
3.4.2.1 Sub-enabler design 82
3.4.2.2 Impacted KPIs and KVIs 86
3.4.3 Sub-enabler 4.3: Trust Management System 87
3.4.3.1 Sub-enabler design 87

3.4.3.2 Preliminary implementation and early validation results 88
3.4.3.3 Impacted KPIs and KVIs 89
3.5 Enabler 5: Synergetic orchestration mechanisms for the computing continuum 89
3.5.1 Sub-enabler 5.1: Multi-agent systems for multi-cluster orchestration 90
3.5.1.1 Sub-enablers design 90
3.5.1.2 Preliminary implementation and early validation results 97
3.5.1.3 Impacted KPIs and KVIs 102
3.5.2 Sub-enabler 5.2: Decentralised orchestration system 102
3.5.2.1 Sub-enabler design 103
3.5.2.2 Preliminary Implementation and early validation results 110
3.5.2.3 Impacted KPIs and KVIs 112
3.5.3 Sub-enabler 5.3: Federated orchestration system 113
3.5.3.1 Enabler Design 113
3.5.3.2 Preliminary implementation and early validation results 115
3.5.3.3 Impacted KPIs and KVIs 116
3.6 Enabler 6: AI/ML algorithms 116
3.6.1 Sub-enabler 6.1: AI/ML-based control algorithms for sustainability 117
3.6.1.1 Sub-enabler design 117
3.6.1.2 Preliminary implementation and early validation results 118
3.6.1.3 Impacted KPIs and KVIs 133
3.6.2 Sub-enabler 6.2: Trustworthy AI/ML-based control algorithms 133
3.6.2.1 Sub-enabler design 133
3.6.2.2 Preliminary implementation and early validation results 134
3.6.2.3 Conceptual solutions 137
3.6.2.4 Impacted KPIs and KVIs 140
3.7 Enabler 7: Network digital twins creation mechanisms 140
3.7.1 Enabler design 140
3.7.1.1 System Components 140
3.7.2 Preliminary implementation and early validation results 141
3.7.2.1 Applicable datasets / environment 142
3.7.2.2 Modelling virtualization effects 142
3.7.3 Impacted KPIs and KVIs 143
3.8 Enabler 8: Real-time Zero-touch control loops automation and coordination system 144
3.8.1 Enabler design 144
3.8.1.1 Internal Architecture of the system components 145
3.8.1.2 Workflows 150
3.8.2 Preliminary implementation and early validation results 153
3.8.2.1 Closed Loop Governance and Coordination Functions 154
3.8.2.2 Specialized CLs: Automation of Transport Network Slices 156
3.8.2.3 Penalty-based management of concurrent service CLs 158
3.8.3 Conceptual solutions 161
3.8.3.1 Specialized CLs: TSN/DetNet control 161
3.8.3.2 Specialized CLs: Dependencies in AI/ML models and NDTs 162
3.8.4 Relation with other enablers 163
3.8.4.1 Specialized CLs for the decentralized orchestration system 163
3.8.4.2 Specialized CLs: Service autoscaling in the continuum 165
3.8.5 Impacted KPIs and KVIs 166

4 Alignment of M&O enablers with the E2E system blueprint 167
4.1 Enabler 1: Network programmability framework 167
4.2 Enabler 2: Monitoring and telemetry framework 168
4.3 Enabler 3: Management capabilities exposure framework 169
4.4 Enabler 4: Security and trustworthiness framework 170
4.5 Enabler 5: Synergetic orchestration mechanisms for the computing continuum 171
4.5.1 Sub-enabler 5.1: Multi-agent systems for multi-cluster orchestration 171

4.5.2 Sub-enabler 5.2: Decentralised Orchestration System 172
4.5.3 Sub-enabler 5.3: Federated Orchestration System 173
4.6 Enabler 6: AI/ML Algorithms 173
4.6.1 Sub-enabler 6.1: AI/ML-based control algorithms for sustainability 173
4.6.2 Sub-enabler 6.2: Trustworthy AI/ML-based control algorithms 174
4.7 Enabler 7: Network digital twins creation mechanisms 175
4.8 Enabler 8: Real-time zero-touch control loops automation and coordination system. 175

5 Contributions to dissemination activities 177

6 Conclusions 179

7 References 180

8 Appendix 188
8.1 Information models 188
8.1.1 Information model for Trustworthy 3P 188
8.1.2 Information models for Resource Orchestration in the continuum 190
8.1.3 Information model for Closed Loop Descriptor 194
8.2 Interfaces 197
8.2.1 Interfaces of SDN Orchestrator and SDN controllers 197
8.2.1.1 E2E SDN Orchestrator 197
8.2.1.2 Technological-Domain SDN controller 198
8.2.2 Interfaces of the Monitoring and Telemetry framework 198
8.2.3 Interfaces of the Management Capabilities Exposure framework 199
8.2.4 Interfaces of CL Governance and CL Coordination functions 200
8.3 Overview of TM Forum, IETF, and ETSI Framework Alignments 202
8.3.1 NBI TMForum 202
8.3.2 Perspective on TM Forum APIs and IETF Standards 202
8.3.2.1 TM Forum - Interoperability and Standardization 202
8.3.2.2 TM Forum alignment with IETF Standards 203
8.3.2.3 Intent-Based Management and Autonomous Networks 203
8.4 Related technologies 206
8.4.1 The compute continuum and the convergence with the edge 206

List of Tables

Table 2-1: WP6 Enablers vs. Hexa-X-II Design Principles. 28
Table 3-1: Management Capabilities Exposure Framework components. 68
Table 3-2: Management capabilities exposure framework interface mapping. 71
Table 3-3: AuthN/AuthZ – protocols comparison. 80
Table 3-4: MSE values of the original model after applying adversarial attacks. 137
Table 3-5: Comparison of the MSE values of the original and robust model after applying FGSM attack.. 137
Table 3-6: Comparison of the MSE values of the original and robust model after applying BIM attack. 137
Table 3-7: CL components. 145
Table 3-8: CL Coordination internal components. 149
Table 3-9: Mapping between software components and CL functions 159
Table 4-1: Alignment of the enablers with E2E system blueprint. 167
Table 5-1: Contributions to papers. 177
Table 5-2: Contributions to demonstrations. 178
Table 8-1: “Identity” class attributes. 188
Table 8-2: “Role” class attributes. 188
Table 8-3: “AccessRule” class attributes. 189
Table 8-4: “Permission” dataType attributes. 189
Table 8-5: E2E SDN Orchestrator exposed interfaces. 197
Table 8-6: Technological-Domain SDN controller exposed interfaces. 198
Table 8-7: Subscription API. 199
Table 8-8: Listing API. 200
Table 8-9: Security API. 200
Table 8-10: CL Governance and Coordination interfaces. 201
Table 8-11: Intent Standards Classification. 206

List of Figures

Figure 2-1: WP6 and its relationship with other WPs. 23
Figure 2-2: Extension of E2E System Blueprint provided from WP2. 24
Figure 2-3: WP6 enablers re-definition. 26
Figure 2-4: Roles in 5G provisioning systems [5GP21]. 32
Figure 2-5: Roles in 6G ecosystem. 33
Figure 3-1 Design of Enabler 1: Network programmability framework. 41
Figure 3-2: ETSI TeraFlowSDN internal architecture. 43
Figure 3-3 Network programmability framework end-to-end workflow for provisioning IETF slice. 44

Figure 3-4: TeraFlowSDN SBI driver gNMI/OpenConfig + Dataplane-in-a-box based on ContainerLab. ... 45
Figure 3-5: TFS integration with Management capabilities exposure framework. ... 45
Figure 3-6: Proposed scenario for providing Anomalous Behaviour Profiling. ... 46
Figure 3-7: Anomalous Behaviour Profiling workflow. ... 47
Figure 3-8: Multi-Segment DetNet Network Architecture. ... 48
Figure 3-9: Local actions and EW interactions between CNCs. ... 48
Figure 3-10: TSN/DetNet SDN controller early validation results. ... 49
Figure 3-11: TFS ZSM-aligned Monitoring-Analytics-Automation Loop. ... 50
Figure 3-12: Sequence diagram: operation of the Monitoring-Analytics-Automation Control Loop. ... 51
Figure 3-13: TeraFlowSDN and MEC integration proof-of-concept. ... 52
Figure 3-14 Sequence diagram of integration of BWM services and ETSI TeraFlowSDN. ... 53
Figure 3-15: TMF API integration within ETSI TFS NBI interface. ... 54
Figure 3-16: High-level architecture of monitoring and telemetry framework. ... 56
Figure 3-17: Monitoring and telemetry framework internal architecture. ... 57
Figure 3-18: Monitoring and Telemetry Framework sequence diagram. ... 58
Figure 3-19: TeraFlowSDN event-driven monitoring. ... 58
Figure 3-20: Prometheus visualisation of data topic. ... 60
Figure 3-21: Energy Monitoring Platform implementation high-level diagram. ... 61
Figure 3-22: Dashboard for energy monitoring visualization. ... 61
Figure 3-23: 6G Monitoring Platform in a zero-touch closed-loop instance. ... 62
Figure 3-24: Monitoring platform dashboard showing battery data collected from cobots. ... 64
Figure 3-25: Monitoring for TSN/DetNet network segment. ... 65
Figure 3-26: Observability signals data collection and fusion. ... 66
Figure 3-27: High level architecture of Management Capabilities Exposure Framework. ... 67
Figure 3-28: Management capabilities exposure framework main components and interactions overview. ... 70
Figure 3-29: Management capabilities exposure framework interfaces. ... 70
Figure 3-30: Registration to Management capabilities exposure framework. ... 72
Figure 3-31: Management capabilities exposure framework topic listing. ... 72
Figure 3-32: Management capabilities exposure framework single topic information retrieval. ... 73
Figure 3-33: Enabler communication with Management capabilities exposure framework. ... 73
Figure 3-34: Overview on Apache Kafka cluster structure. ... 74
Figure 3-35: Security and trustworthiness framework. ... 75
Figure 3-36: Access control governance framework for enabler 4.1. ... 76
Figure 3-37: Conceptual model for the mechanisms in enabler 4.1. ... 77
Figure 3-38: Trustworthy 3P information model. ... 78
Figure 3-39: Design time – workflow. ... 79
Figure 3-40: 3P onboarding – system view. ... 79

Figure 3-41: 3P onboarding – workflow. 80
Figure 3-42: Operation time – REST/OpenAPI workflow. 81
Figure 3-43: Operation time – Netconf/YANG workflow. 81
Figure 3-44: User-centric service provisioning – internal architecture. 83
Figure 3-45: Service fulfilment workflow. 84
Figure 3-46: Service activation – workflow. 85
Figure 3-47: Service assurance – workflow. 86
Figure 3-48: High-level architecture of trust evaluation functions. 87
Figure 3-49: Trustworthiness measurements of the metaheuristic functionality allocation mechanism. 89
Figure 3-50: High-level view of multi-cluster orchestration components for the computing continuum. 91
Figure 3-51: Multi-agent interactions. 91
Figure 3-52: RL-driven autoscaling mechanisms for multi-cluster deployments. 93
Figure 3-53: Network Service Mesh solution over Kubernetes clusters [NET24a]. 95
Figure 3-54: REC-EXEC workflow for platform monitoring. 95
Figure 3-55: REC-EXEC workflow for service deployment. 96
Figure 3-56: REC-EXEC implementation. 97
Figure 3-57: RL-driven autoscaling implementation. 100
Figure 3-58: RL autoscaling early results. 101
Figure 3-59: High level view of the communications for the functionality allocation mechanism. 102
Figure 3-60: ETSI NFV MANO Framework. 104
Figure 3-61: Proposed update aligned with the ETSI NFV MANO Framework. 104
Figure 3-62: Common Infrastructure Management and Services Provisioning System. 105
Figure 3-63: Network Service definition and deployment example. 107
Figure 3-64: NS deployment in the decentralised orchestration approach [HEX223-D32]. 108
Figure 3-65: Deployment Node GUI example [HEX223-D33]. 109
Figure 3-66: Infrastructure Layer Emulator (ILE). 111
Figure 3-67: S/W Design of DLT based federation. 114
Figure 3-68: Sequence diagram of service federation using DLT. 115
Figure 3-69: Initial implementation of DLT federation. 115
Figure 3-70: Average duration of service federation steps. 116
Figure 3-71: High-level architecture for AI/ML-based M&O. 117
Figure 3-72: End-to-end energy optimization from power system to node to network [VHI+21]. 119
Figure 3-73: CVAE Structure. 120
Figure 3-74: Energy savings and KPI impact according to the CVAE model. 120
Figure 3-75: DRL model training environment. 121
Figure 3-76: Reward and performance statistics. 123
Figure 3-77: Flowchart of Functionality Allocation algorithm based on genetic algorithm paradigm. 125

Figure 3-78: Energy consumption measurements of the functionality allocation mechanism. 126
Figure 3-79: Actions use case diagram within the MLOps framework for vendors and operators. 127
Figure 3-80: High level architecture for sustainability measurements 129
Figure 3-81: Example of carbon production equivalent measurements over a node, in gCO_2. 129
Figure 3-82: Example of power consumption measurements over a node in mW. 129
Figure 3-83: Example of power consumption measurements over different ML Artifacts. 130
Figure 3-84: Schematic of decentralised learning. 131
Figure 3-85: (a) 500 samples and (b) 980 samples. 131
Figure 3-86: QMIX application on service auto-scaling. 132
Figure 3-87: High-level architecture for trustworthy AI/ML-based control. 134
Figure 3-88:Topology in our in-house network emulator 135
Figure 3-89: Adversarial attack using FGM method. 136
Figure 3-90: Adversarial attack using BIM method. 136
Figure 3-91: General Overview of privacy protection framework. 138
Figure 3-92: XRL integration with RL-based control. 139
Figure 3-93: Single- and multi-agent XRL in network parameter optimization. 139
Figure 3-94: IETF proposed Reference Architecture of Network Digital Twin. 141
Figure 3-95: Experimental test-bed setup. 143
Figure 3-96: Effect of container-based virtualization on packet loss and round-trip delay 143
Figure 3-97: CL functions and CL Governance during CL provisioning. 146
Figure 3-98: CL functions and CL Governance during CL runtime. 147
Figure 3-99: CL coordination functions. 148
Figure 3-100: Workflow for CL Provisioning. 150
Figure 3-101: Workflow for CL Runtime. 151
Figure 3-102: Conflict Management Workflow - sequence diagram. 152
Figure 3-103: Conflict Management Workflow - Activity Diagram 153
Figure 3-104: Closed-Loop Governance Function software architecture. 155
Figure 3-105: Closed-Loop for automation of transport network slices. 157
Figure 3-106: Intent manager system. 159
Figure 3-107: Network emulator used in the experiments. 160
Figure 3-108: Experiment observations – conversational video QoE, URLLC packet loss and penalty. 161
Figure 3-109: Specialized Closed Loop for TSN/DetNet Control. 162
Figure 3-110: Inter-Control Loop dependencies between AI/ML models and NDTs. 163
Figure 3-111: CLs deployment and governance. Alignment with sub-enabler 5.2. 164
Figure 3-112: CLs coordination. Alignment with sub-enabler 5.2. 164
Figure 3-113: Service autoscaling in the continuum. 165
Figure 4-1: Overall alignment between Enabler 1 and the E2E system blueprint. 168

Figure 4-2: Overall alignment between Enabler 2 and the E2E system blueprint. 168
Figure 4-3: Overall alignment between Enabler 3 and the E2E system blueprint. 169
Figure 4-4: Integration Fabric – high-level architecture 170
Figure 4-5: Overall alignment between sub-enabler 4.3 and the E2E system blueprint 170
Figure 4-6. Overall alignment between Sub-enabler 5.1 and the E2E system blueprint. 171
Figure 4-7: Overall alignment between Sub-enabler 5.3 and the E2E system blueprint – new interfaces 173
Figure 4-8. Overall alignment between Sub-enabler 6.1 and the E2E system blueprint. 174
Figure 4-9. Overall alignment between Sub-enabler 6.2 and the E2E system blueprint. 174
Figure 4-10. Suggested approach for aligning Enabler 7 with the E2E system blueprint. 175
Figure 4-11: Suggested approach for aligning Enabler 8 with the E2E system blueprint. 176
Figure 8-1: Data model for resources. 190
Figure 8-2: Data model for application graph. 191
Figure 8-3: Resource Specification data model. 191
Figure 8-4: Service Specification data model. 192
Figure 8-5: Node data model. 193
Figure 8-6: Data model for service application. 194
Figure 8-7: Data model used for functionality allocation system. 194
Figure 8-8: Information model for CL Descriptor (1). 195
Figure 8-9: Information model for CL Descriptor (2). 196
Figure 8-10: Information model for CL Descriptor (3). 197
Figure 8-11: Integration between Monitoring and telemetry framework and other enablers. 198
Figure 8-12: Overview of standards related to NBI for autonomous networks. 202
Figure 8-13: TM Forum's APIs for Domain Automation – an example from [TMF-IG1230]. 204
Figure 8-14: IETF Network Slicing Framework and TMF Service Management Approach intersection. 204
Figure 8-15: TM Forum, ETSI and IETF NBI alignment. 206

Acronyms and abbreviations

Term	Description
3GPP	3rd Generation Partnership Project
3P	3rd party
ABP	Anomalous Behaviour Profiling
ACL	Access Control List
ADT	Application Data Transfer
AGV	Automated Guided Vehicle
AI	Artificial Intelligence
AM	Attack Mitigator
AMR	Autonomous Mobile Robot
API	Application Programming Interface
B2B	Business to Business
B2B2C	Business to Business to Consumer
B2B2X	Business to Business to Everything
BGP	Border Gateway Protocol
BSS	Business Support System
BWM	BandWidth Management
CAD	Centralized Attack Detector
CFS	Customer Facing Service
CI/CD	Continuous Integration/Continuous Deployment
CIM&SPS	Common Infrastructure Management and Services Provisioning System
CL	Closed loop
CM	Configuration Management
CN	Core Network
CNC	Centralized Network Controller
CoAP	Constrained Application Protocol
COP	Capability Operator
CPU	Central processing unit
CQF	Cyclic Queueing and Forwarding
CSMF	Communication Service Management Function
CSP	Communication Service Provider
CVAE	Conditional Variational Autoencoder

DB	Database
DC	Data Centre
DCSP	Data Centre Service Provider
DDPG	Deep Deterministic Policy Gradient
DetNet	Deterministic Networking
DL	Downlink or Decentralized Learning
DLT	Distributed Ledger Technology
DMS	Deployment Management Services
DN	Data Network or Distinguished Name or Deployment Node
DNN	Data Network Name
DNS	Domain Name System
DPU	Data Processing Unit
DRL	Deep Reinforcement Learning
DS	Deployment Service
DSM	Digital Service Manager
DSP	Digital Service Provider
DT	Digital Twin
E2E	End-to-End
eMMC	embedded MultiMedia Card
EMP	Energy Monitoring Platform
ENI	Experiential Networked Intelligence
ESG	Environmental, social and governance
ETSI	European Telecommunications Standards Institute
EW	East-Westbound
GAN	Graph Attention Network
GIN	Graph Isomorphism Network
GNN	Graph Neural Network
GPU	Graphics Processing Unit
GST	Generic network Slice Template
GSW	Cell-Site Gateways
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
HW	Hardware
IbM	Intent-Based Management

IdP	Identity Provider
IETF	Internet Engineering Task Force
ILE	Infrastructure Layer Emulator
IMF	Intent Management Function
IMS	IP Multimedia Subsystem
IoT	Internet of Things
IP	Internet Protocol
IRN	Infrastructure Registry Node
IRS	Infrastructure Registry Service
IRTF	Internet Research Task Force
ISG	Industry Specification Groups
ISPS	Infrastructure Status Prediction Service
ISPN	Infrastructure Status Prediction Node
ITU	International Telecommunication Union
JSON	JavaScript Object Notation
K8s	Kubernetes
KPI	Key Performance Indicator
KVI	Key Value Indicator
L2VPN	Layer 2 Virtual Private Network
LCM	Life Cycle Management
LDAP	Lightweight Directory Access Protocol
LoTAF	Level of Trust Assessment Function
LSTM	Long Short-Term Memory
LTS	Long Term Support
M&O	Management and Orchestration
MARL	Multi-Agent Reinforcement Learning
MDAF	Management Data Analytics Function
MEC	Multi-access Edge Computing
MIB	Management Information Base
MIP	Mixed Integer Programming
ML	Machine Learning
MNO	Mobile Network Operator
MQTT	Message Queuing Telemetry Transport
MSE	Mean Squared Error

NBI	NorthBound Interface
NB-IoT	NarrowBand IoT
NDT	Network Digital Twin
NEF	Network Exposure Function
NF	Network Function
NFV	Network Functions Virtualization
NFV-NS	NFV Network Service
NFVI	Network Function Virtualization Infrastructure
NFVO	Network Function Virtualization Orchestrator
NIC	Network Interface Card
NM	Network Modelling
NMI	Network Management Interface
NOS	Network Operating System
NPN	Non-Public Network
NRM	Network Resource Model
NS	Network Service
NSaaS	Network Slice as a Service
NSM	Network Service Mesh
NSMF	Network Slice Management Function
NSSMF	Network Slice Subnet Management Function
NSSAI	Network Slice Selection Assistance Information
NTN	Non Terrestrial Network
NWDAF	Network Data Analytics Function
OAM	Operations, Administration, and Maintenance
OIDC	OpenID Connect
ONF	Open Networking Foundation
O-RAN	Open Radio Access Network
OS	Operating System
OSM	Open Source MANO
OSS	Operations Support System
OTLP	OpenTelemetry Protocol
OTN	Optical Transport Network
OTT	Over the Top
PCF	Policy Control Function

PDL	Policy Description Language
PDU	Packet Data Unit
PM	Performance Management
PMF	Privacy Management Function
PoC	Proof of Concept
POF	Privacy Operation Function
QoE	Quality of Experience
QoS	Quality of Service
QT	Quantitative Target
RAM	Radio Access Memory
RAN	Radio Access Network
RAW	Reliable and Available Wireless
RBAC	Role-Based Access Control
REC-EXEC	REsource orchestrator for Continuum across EXtreme-edge, Edge and Cloud
REST	Representational state transfer
RF	Radio Frequency
RFC	Request for Comments
RFS	Resource Facing Service
RL	Reinforcement Learning
RNN	Recurrent Neural Network
ROADM	Reconfigurable Optical Add-Drop Multiplexers
RPC	Remote Procedure Call
SAI	Securing Artificial Intelligence
SBA	Service Based Architecture
SBI	SouthBound Interface
SBMA	Service Based Management Architecture
SCI	Security Control Interface
SDN	Software Defined Network
SDO	Standard Development Organizations
SLA	Service Level Agreement
SLO	Service Level Objective
SMF	Session Management Function
SMO	Service Management and Orchestration
SNMP	Simple Network Management Protocol

SNS-JU	Smart Networks and Services Joint Undertaking
SON	Self-Organizing Network
SoTA	State of the Art
SP	Service Provider
SQL	Structured Query Language
SRN	Service Registry Node
SRS	Services Registry Service
SW	Software
SWV	Software Vendor
TAPI	Transport API
TAS	Time-Aware Shaping
TE	Traffic Engineering
TEAS	Traffic Engineering Architecture and Signalling
TFS	TeraFlowSDN
TMF	TM Forum
TLA	Trust Level Agreement
TN	Transport Network
TLS	Transport Layer Security
TSN	Time-Sensitive Networking
UAV	Unmanned Aerial Vehicle
UDM	Unified Data Management
UE	User Equipment
UI	User Interface
UL	Uplink
UML	Unified Modelling Language
UPF	User Plane Function
URLLC	Ultra-Reliable Low Latency Communication
URSP	UE Route Selection Policy
VIM	Virtual Infrastructure Manager
VISP	Virtual Infrastructure Service Provider
VM	Virtual Machine
VMAF	Virtualized MEC application function
VNF	Virtual Network Function
VPN	Virtual Private Network

WAN	Wide Area Network
WP	Work Package
XR	eXtended Reality
XRL	eXplainable Reinforcement Learning
ZSM	Zero touch network and Service Management

1 Introduction

1.1 Objective of the document

This document aims to describe the design, initial implementation and early validation results of the enablers for the Hexa-X-II Smart Network Management framework. The global objective is to provide feedback for the design of the Hexa-X-II End-to-End (E2E) 6G System Blueprint, in its 2nd iteration. The enablers presented in this document constitute an evolution of the initial set identified in D6.2 [HEX223-D62], proposing a restructuring of their classification and introducing their implementation and contributions to Hexa-X-II Proof of Concepts (PoC) implementation.

This deliverable can be considered as an intermediate milestone in the overall process of defining the Management and Orchestration (M&O) system for the future 6G networks, providing a preliminary version of M&O enablers' design and implementation. However, it plays a crucial role in the methodology planned in the project for the definition of the Hexa-X-II E2E 6G System Blueprint. This methodology is based on an inter-work-package collaboration structured in multiple cycles of (i) initial bottom-up contributions from the technical Work Packages (WP) focused on specific aspects of the 6G system architecture, which are then (ii) consolidated in progressive iterations of the end-to-end blueprint. This is in turn (iii) analysed by the technical WPs in order to provide feedback and contribute to the refinement of the next iteration of the E2E system blueprint.

In this process, this deliverable will provide feedback to an intermediate iteration of the Hexa-X-II E2E 6G System Blueprint contributing to its 2nd official iteration (see section 2 for further details). The perspective provided in this deliverable is particularly relevant because it captures lessons learnt from early implementation and validation activities, going well beyond the initial conceptual design provided in D6.2 [HEX223-D62]. The initial integration of M&O enablers in Hexa-X-II Proof-of-Concepts (PoC) allows to consolidate the design of the enablers in concrete frameworks and systems, offering an opportunity to evaluate and compare different architectural choices in real scenarios.

In this direction and following an evolutionary approach, the enablers initially identified in D6.2 [HEX223-D62] have been re-organized in a short list of 8 enablers, with some of them further structured in sub-enablers. For each of them, the document presents: (i) the technical approach; (ii) the ongoing implementation; (iii) the early validation results, and (iv) their impact to relevant Key Performance Indicators (KPI) and Key Value Indicators (KVI), showing as well how they have been integrated in the project PoCs. Feedback to the E2E system blueprint design is presented at the end of the document following on a per-enabler approach and collecting the results of the latest interactions.

1.2 Structure of the document

The document is structured as follows:

- Section 1 introduces the document with an overview of its content and structure, defining the main objectives of the deliverable together with its position and main contribution to the Hexa-X-II project.
- Section 2 provides a global view of the Smart Network Management framework presenting the refined list of M&O technical enablers and their classification. This section analyses the alignment and the contribution of the framework to the design principles of Hexa-X-II architecture, as well as the mapping of the proposed system with the 6G stakeholders. Finally, it highlights the main innovations introduced in the Hexa-X-II M&O framework, with particular reference to the enhancements with regards to the original Hexa-X architecture.
- Section 3 is focused on the M&O enablers and sub-enablers, constituting the core of the document. For each enabler, this section reports the current status of design, implementation and evaluation results, discussing the contributions to KPIs and KVIs. It should be noted that each enabler may have several implementations, e.g., focusing on particular sub-components, examples of algorithms or customization of workflows and configurations for a given target scenario or objective. For some enablers, conceptual solutions not yet at the implementation level are also reported and, where applicable, the relation with other technologies or other enablers are documented.

- Section 4 discusses the alignment of M&O enablers with the ongoing version of the E2E system blueprint used as reference for feedback on smart network management aspect. For each enabler, this section analyses the mapping with layers, components and interfaces of the E2E system blueprints, identifying possible gaps and requirements for architectural updates, refinements or even new components and interfaces. This is the main outcome of the document in what regards its contribution to the end-to-end Hexa-X-II system design.
- Section 5 reports the contributions to dissemination and demonstrations.
- Section 6 provides the conclusions and a roadmap towards the finalization of the Smart Network Management framework.
- The Appendixes provide additional details on information models and interfaces.

2 Smart Network Management

The Hexa-X-II project approaches the overall design of the 6G E2E System in the context of its WP2, which is intended to harmonize the E2E design principles towards producing the 6G SNS platform blueprint. In this context, WP6 (to which this deliverable belongs) receives inputs from WP2 regarding the overall platform design and, combining it with the information from other work packages (WP1 describing the use cases, and WP3 addressing the 6G Architecture design), provides outputs to WP2 regarding the specific WP6 smart network management topics to be integrated as part of the end-to-end system. Figure 2-1 outlines this information flow between WP6 and the other related WPs mentioned here (interactions between WP2 and WP3 are not shown in the picture for simplicity).

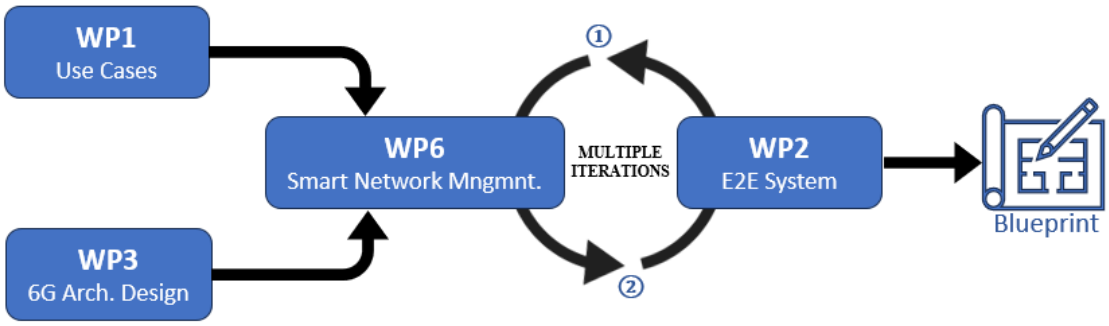


Figure 2-1: WP6 and its relationship with other WPs.

To address this flow of information towards WP2, WP6 has defined a framework considering: (i) the contributions to the Hexa-X-II architecture design principles, (ii) the mapping with the envisaged 6G stakeholders, (iii) the definition and the alignment of the WP6 M&O technical enablers (already introduced in the previous Deliverable D6.2 [HEX223-D62]) with the initial blueprint provided from WP2 (through the flow ① in Figure 2-1), and (iv), the envisaged contributions of those M&O enablers towards the future 6G smart networks (flow ② in the figure). This Section contextualizes the M&O framework in the E2E system blueprint, initially introducing the M&O technical enablers (Subsection 2.1) and discussing their contribution to the architecture design principles (Subsection 2.2). The 6G stakeholders, as defined in WP2, are briefly reported in Subsection 2.3 highlighting their interaction with the M&O framework. Besides, Subsection 2.4 introduces the main innovations envisaged in the Hexa-X-II M&O framework considering the Hexa-X M&O architecture [HEX22-D62] as baseline.

Figure 2-2 depicts the current version (at the time this document is being edited) of the E2E System blueprint provided by WP2. It represents the main building blocks for each Mobile Network Operator (MNO) that would aim to align its network design with the future envisaged 6G E2E System. The design is basically a four-layer stack (left) consisting of an Infrastructure layer (bottom), a Network Functions layer (middle), a Network-centric Application Layer (top), and an Application Layer (on top of the previous one). Besides these four layers there is also a so-called Pervasive Functionalities block (right).

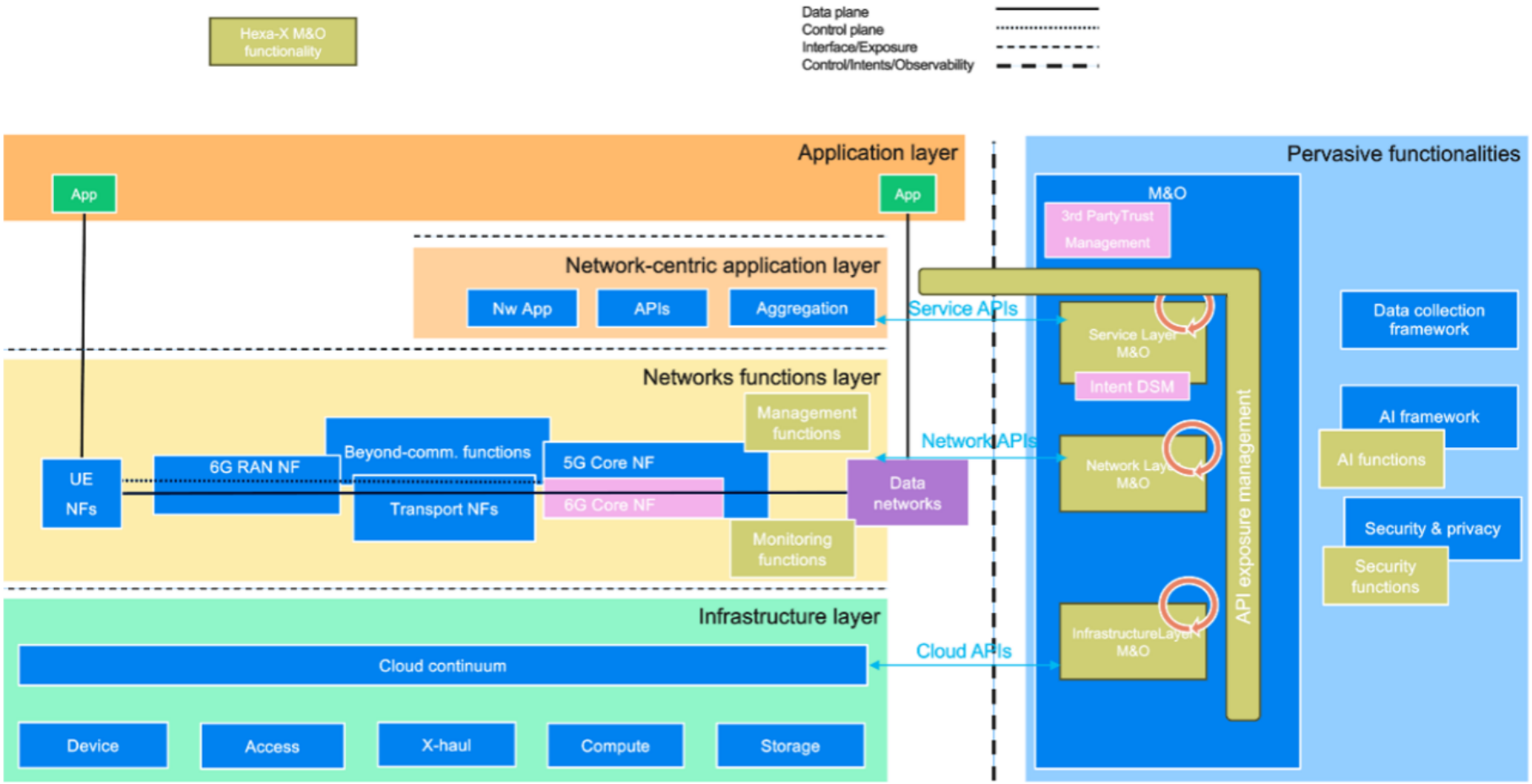


Figure 2-2: Extension of E2E System Blueprint provided from WP2.

- Networks Functions Layer: consists of User Equipment (UE) and subnetwork functions (concerning the UEs protocol stack and their associated subnetworks), Radio Access Network (RAN) functions, Core Network (CN) Functions, Transport Network functions, Beyond communication functions (concerning functions beyond the communications topic itself, e.g., novel Radio Frequency (RF)-based sensing, Artificial Intelligence (AI)-related, 6G positioning and mapping, etc.), and the Data network (as defined in the 3GPP architecture, targeting those functionalities related to services provided by the operator, the Internet access, or 3rd party services).
- Network-centric Application layer: consists of network-centric applications, i.e., applications that are more centred on the network capabilities, possibly depending on network features and that can interact with the control plane of the Network Functions Layer to provide enriched services to vertical applications. Moreover, these network-centric applications can provide certain actions/updates to the network functions and thereby provide quick ways to modify the network functionality, enabling additional operation and optimization of the network (e.g., rApp in O-RAN).
- Application layer: represents the applications that may have certain service expectations from the network itself. These expectations may depend on the application requirements, such as bandwidth, latency, etc. The Network functions layer and the Network-centric applications layer expose network information and network control Application Programming Interfaces (API) through the interface/exposure towards this application layer. This enables application developers to specify their requirements to the network as well as to adjust the expectations of the applications based on network conditions. Operations APIs for the service M&O can also be exposed to the application ecosystem actors (e.g., to vertical providers) through developer-friendly intent-based abstractions and human-oriented intent-based APIs.
- Pervasive Functionalities block: refers to those functionalities and frameworks that operate at the various layers of the 6G E2E system blueprint, i.e., at the infrastructure, network function, and application levels, and can interact with all the different layers, as well.

The M&O functionalities are part of the Pervasive Functionalities block, represented by the "M&O" block on the right-hand side of the diagram. Besides, there are also other additional "Management Functions" within the "Networks Functions Layer" (middle). For the pervasive M&O block [HEX223-D21] provides the following definition:

"*The management and orchestration functionality will cope with novel and more complex services combining both network capabilities, and beyond communications capabilities such as sensing and computing. Smart network management will provide a uniform orchestration across a continuum of resources from extreme edge to edge to central clouds. Automation functionality will provide more and more degrees towards autonomy with fully automated closed-loop control, supported by intent-based management and AI/ML techniques. It will also comprise the Continuous Integration/Delivery (CI/CD) intrinsically associated to the development of software components".*

Beyond this, in the most updated version of the blueprint (the one in Figure 2-2), this pervasive M&O block has been enriched with the following internal elements:

- A "Service Layer M&O" block, intended to provide specific M&O functionalities for the so-called Network-centric Application Layer through a set of Service APIs ("Service APIs" blue arrow).
- A "Network Layer M&O" block, with a functionality similar to the previous block, but with respect to the Network Functions Layer.
- An "Infrastructure Layer M&O" block, with a similar role to the two previous blocks, but focusing on the Infrastructure Layer.
- A "3rd Party Trust Management" block (top), intended to offer multi-tenancy support in resource sharing environments, in terms of (i) resource controllability separation (providing tenants with segregated yet customized management spaces), (ii) user-centric network management (defining policies for tenant subscribers in relation to service delivery and consumption), and (iii) Service Level Agreement (SLA) enforcement, assurance and verifiability.
- An "Intent Digital Service Manager (DSM)" block, associated to the Service Layer M&O block, intended to allow tenants (with or without technical knowledge) to interact with the M&O system and to request a desired service without specifying how it needs to be deployed, leaving the system to find

the most suitable option in an autonomous and efficient way. (Intent based management is addressed in WP2 [HEX223-D22], which has defined an Intent-based management automation framework and the related enablers).

- An "API Exposure Management" block, designed to manage and supervise the use, security, and access of APIs that are exposed to internal or external blocks of the Pervasive Functionalities. So, it entails building methods to promote communication and interactions between the blocks described on previous bullet points. It guarantees, as well, a single-entry point for managing external communication flows, which will activate the Pervasive Functionalities.

On the other hand, those Management Functions within the Network Functions layer correspond to generalized Management Functions defined by 3GPP in [28.533] as logical entities playing the role of Management Service consumer and/or Management Service producer.

As can be noted this blueprint is still a very high-level approach, which is provided as an initial proposal from WP2 to receive feedback from WP6. Based on that, and on the information received from other technical work packages in the project, this blueprint will be evolved and refined in the context of WP2.

2.1 M&O technical enablers

To address the implementation of the M&O functionality, and to study its alignment with the blueprint described in the previous section, WP6 has drafted a set of technical enablers. From the WP6 perspective these enablers are the most relevant technical artifacts that should be provided to implement the M&O functionality for the network and the network services (NS) deployed on it. An initial list of those enablers was already provided in the previous Deliverable D6.2 [HEX223-D62]. However, since the D6.2 release, that initial list has been redefined, as shown in Figure 2-3.

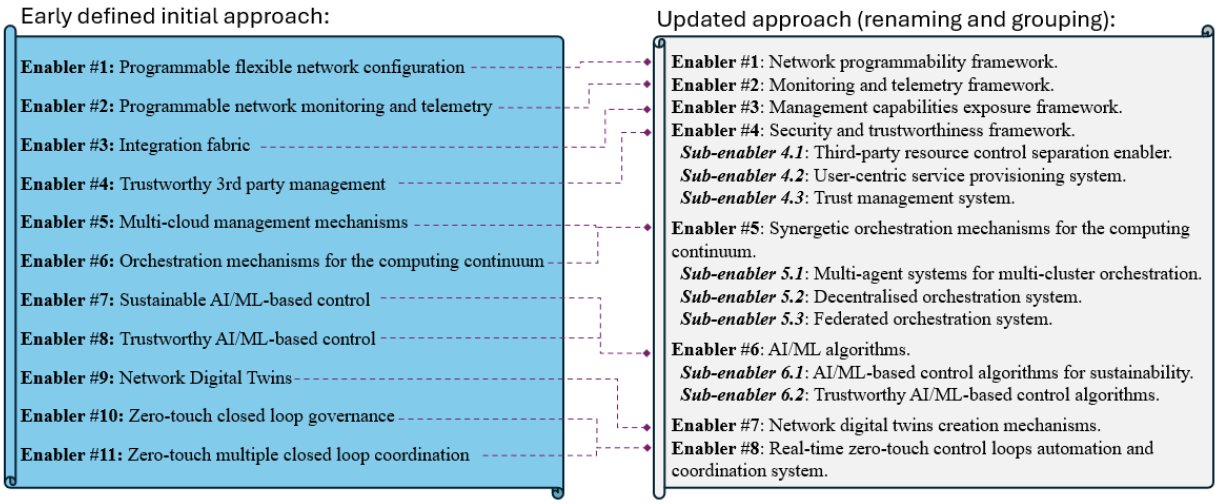


Figure 2-3: WP6 enablers re-definition.

The technical approach of this new set of enablers is a bit more structured than the previous one in D6.2 [HEX223-D62] by aggregating enablers providing complementary features in the same technical area and introducing the concept of "sub-enablers", which are used for those enablers covering various technical aspects that can be clearly distinguished. Moreover, some enablers have been renamed, trying to highlight that they are *artifacts*, *mechanisms*, *systems*… intended to enable or provide certain functionality.

All these WP6 enablers will be described in detail through this document (in Section 3). The following subsection describes how these enablers align with the Hexa-X-II architectural design principles.

2.2 Mapping with Hexa-X-II architecture design principles

The Hexa-X-II architecture design principles are defined in [HEX223-D21], and are summarized below:

- **Principle 1 - Support and exposure of 6G services and capabilities**. The design of the 6G system shall enable efficient incorporation of 6G digital services. The architecture solution shall also be able to expose new and existing capabilities to E2E applications. This can for example be used for predictive M&O.
- **Principle 2 - Full automation and optimization**. The architecture should support full automation of network and service management operations, utilizing distributed AI/ML agents to manage and optimize the system without human interaction. Key features include observability and analytics as well as intent-based management. Trust and explainability aspects related to AI functionality are also important to guarantee. The system should also support continuous orchestration over multiple stakeholders and domains, in an E2E approach.
- **Principle 3 - Flexibility to different network scenarios**. The architecture should be designed to support digital inclusion, by adapting to network-of-network topologies such as subnetworks, non-public networks, autonomous networks, mesh ad hoc networks, new spectrum, and Non-Terrestrial Networks (NTN), without degradation of performance. Addition of service capabilities and new service endpoints can be done at run-time without changes to existing E2E services. The network should support increased application awareness and adaptive Quality of Service (QoS) and Quality of Experience (QoE).
- **Principle 4 – Network scalability**. The system architecture needs to be scalable to different networks and deployments. The architecture should support very small to very large-scale deployments, by scaling up and down network resources based on mobility and time-varying traffic needs and by utilizing the underlying shared cloud platform. The architecture should be scalable to also support multi-stakeholder deployments.
- **Principle 5 - Resilience and availability**. The architecture should allow mobile network operators (MNOs) to build deployments with high resilience and availability. The architecture shall support separation of control and user planes, and resilient mobility solutions as a method to provide service availability. Furthermore, the architecture should support subnetwork resilience e.g., if a subnetwork loses connectivity it should connect with another subnetwork to remove single point of failures.
- **Principle 6 - Persistent security and privacy**. The 6G system should be able to address current as well as future threats in a resilient manner and incorporate security fundamentals in its design. Furthermore, the 6G system should inherently support the preservation of privacy and allow different levels of anonymity also for future services.
- **Principle 7 - Internal interfaces are cloud optimized**. Network interfaces should be designed to be used in a cloud-native environment, utilizing state-of-the-art cloud technologies, design patterns, platforms, and IT tools in a coherent and consistent manner. Care should be taken to design proper service separation to maximize the potential for service reuse and allow for service innovation with minimal integration efforts (plug-and-play).
- **Principle 8 - Separation of concerns of network functions.** The network functions should have bounded context (separation of concerns) and all dependencies among services should be through their APIs, with minimal dependency on other network functions. This allows network functions to be developed and replaced more independently from each other and also allows efficient scaling of network resources. The functionality of network functions should not be duplicated.
- **Principle 9 - Network simplification in comparison to previous generations**. The network architecture shall be streamlined to reduce the complexity of RAN and CN functions with fewer (well-motivated) parameters to configure and fewer external interfaces, to maximize innovation, and to reduce time to market.
- **Principle 10 - Minimizing environmental footprint and enabling sustainable networks**. The design of the 6G system should ensure that the environmental footprints can be justified, and any increased footprint should be clearly motivated with added value, cost efficiency, and societal benefits. The architecture design should enable to disable or deactivate functions that are not in use, either short-term or long-term, to ensure zero power at zero load. It should also facilitate circular practices through higher modularity of components to limit the replacement to only the degraded part or compatibility with software update to extend the lifetime.

Based on this, and on the WP6 enablers in the previous Subsection 2.1, the following matrix represents the mapping between these enablers and each of the principles listed above:

Table 2-1: WP6 Enablers vs. Hexa-X-II Design Principles.

WP6 Enablers ↓ / Design principles →	Principle 1	Principle 2	Principle 3	Principle 4	Principle 5	Principle 6	Principle 7	Principle 8	Principle 9	Principle 10
1: Network programmability framework		◆	◆	◆	◆		◆			
2: Monitoring and telemetry framework		◆	◆							◆
3: Management capabilities exposure framework	◆		◆				◆			
4: Security and trustworthiness framework										
4.1: Third-party resource control separation enabler					◆	◆		◆		
4.2: User-centric service provisioning system			◆			◆				
4.3: Trust management system						◆				
5: Synergetic orchestration mechanisms for the computing continuum										
5.1: Multi-agent systems for multi-cluster orchestration	◆	◆	◆				◆			◆
5.2: Decentralised orchestration system	◆	◆	◆	◆	◆		◆	◆		◆
5.3: Federated orchestration system	◆	◆	◆							
6: AI/ML algorithms										
6.1: AI/ML-based control algorithms for sustainability		◆	◆							◆
6.2: Trustworthy AI/ML-based control algorithms					◆	◆				
7: Network digital twins creation mechanisms	◆	◆								
8: Real-time zero-touch control loops automation and coordination system		◆		◆						

The rationale regarding this alignment for each of the WP6 enablers is presented below:

- Enabler 1 (Network programmability framework) is aligned with the following principles:
  - Principle 2, allowing the network automation through the autonomous control of the transport network connectivity services.
  - Principle 3, in the sense that multiple backhaul connectivity can be provided to support different network splits and scenarios.
  - Principles 4 and 5, as this enabler can be easily scaled, and multiple replicas can provide resilience and scalability.
  - Principle 7, since this enabler has been designed for cloud native deployments.
- Enabler 2 (Monitoring and telemetry framework) is aligned with the following principles:
  - Principle 2, because it contributes to provide the necessary data to fuel automation and optimization lifecycles.
  - Principle 3, as different network scenarios can be covered since the enabler can be fully configured.
  - Principle 10, as this enabler also considers energy consumption metrics monitoring, helping to minimize environmental footprint and enabling sustainable networks.
- Enabler 3 (Management capabilities exposure framework) is aligned with the following principles:
  - Principle 1, because this enabler constitutes a connector, offering a set of communication channels that unlocks the support and the exposure of 6G capabilities playing the role of interoperation medium between API producers and consumers. The main aim is to offer connectivity/reachability within the M&O system and provide a single-entry point for external incoming communication and address them to the enabler involved.
  - Principle 3, because the design rationale behind the Management capabilities exposure framework (cloud native, plug-and-play) makes it configurable to adapt to different network scenarios, without any performance degradation. Internal and external interactions with the integration fabric will be asynchronous and event-driven such that the addition of new services and the exposure of particular QoS/QoE can be done at runtime.
  - Principle 7, since this enabler is fully developed using a cloud-native technology, which is fully compliant with the rationale of design principles of service separation, reutilization and cloud compatible interfaces. The reusability and service separation are made possible by the containerization

of the component of this enabler. The interface offered is cloud compatible because it leverages REST (Representational state transfer) interfaces and an event driven architecture (Apache Kafka [KAF24], produce/consumer asynchronous communication). This results in a plug-and-play compatibility for on-boarding, off-boarding, and communication of the integration fabric with the involved enablers.

- Enabler 4 (Security and trustworthiness framework) is aligned with the following principles:
  a) In what regards sub-enabler 4.1 (Third-party resource control separation enabler):
     - Principle 5, defining management spaces per tenant that allow the operation and control of allocated services, applications and resources.
     - Principle 6, defining what a tenant is authorized to do with their services/resources.
     - Principle 8, controlling and limiting the interaction with the system resources based on the roles and profiles defined per party.
  b) In what regards sub-enabler 4.2 (User-centric service provisioning system):
     - Principle 3, with the provision of personalized services according to specific and customized SLAs per partner.
     - Principle 6, as user-centric service definitions inherently integrates considerations of security and privacy, ensuring that users' trust and confidence are part of the experience.
  c) In what regards sub-enabler 4.3 (Trust management system):
     - Principle 6 is met because within this sub-enabler, security aspects are considered among others (e.g., availability, reliability) for assessing the trustworthiness to the system's compute nodes and establishing trustworthy communication paths holding sensitive data.

- Enabler 5 (Synergetic orchestration mechanisms for the computing continuum) is aligned with the following principles:
  a) In what regards sub-enabler 5.1 (Multi-agent systems for multi-cluster orchestration):
     - Principle 1, allowing the consumption of network management APIs to support the lifecycle management of E2E applications that can be deployed across resources in the computing continuum.
     - Principle 2, allowing the injection of automation and intelligence in orchestration mechanisms, considering the ML-driven interaction and collaboration among multiple agents. This takes advantage of observability stacks for data and analytics processing to assist decision making.
     - Principle 3, allowing the deployment of services over multi-cluster environments and taking advantage of network service mesh approaches for network management.
     - Principle 7, adopting a cloud-native approach in services development and orchestration, considering separation of concerns among the various layers (network function layer, network-centric application layer, application layer) and exposure and consumption of APIs for interaction among network and edge/cloud application providers.
     - Principle 10, allowing the optimal deployment of services across infrastructure in the computing continuum, having energy efficiency as a high-level optimization objective.

  b) Regarding sub-enabler 5.2 (Decentralized orchestration system):
     - Principle 1, in what regards the dynamic exposure of the capabilities of the infrastructure layer towards the resource M&O mechanisms; also, regarding the exposure of the capabilities of the management service components, associated with the SBMA approach.
     - Principle 2, in what regards the automation for the network services instantiation and fulfilment mechanisms; also, in what regards the optimization processes associated to these mechanisms (network services must be deployed and operated considering the optimal set of infrastructure resources in the in the continuum, which can be highly heterogeneous and volatile).
     - Principle 3, in what regards the integration of different networks (Non-Public Networks (NPN), NTNs, mesh ad hoc networks, etc.), as part of the network continuum; also, regarding the possibility to M&O resources and services at run-time without changes to existing E2E services.
     - Principle 4, in what regards supporting small and very large-scale deployments (this is in fact one of the main features targeted by the computing continuum concept itself), and regarding the supporting of multi-stakeholder deployments.
     - Principle 5, in what regards allowing MNOs to build deployments with high resilience and availability, since these are two of the main principles associated to the M&O systems as a whole,

which is also targeted by this specific decentralized orchestration approach. Separation of user and control planes is also performed, as well as high service availability, even considering the volatility of those resources at the extreme-edge domain.
- Principle 7, since the computing continuum concept precisely relies on cloud-native internal (and external) exposed interfaces, as well as in the "plug-and-play" concept to dynamically integrate volatile infrastructure resources.
- Principle 8, since the computing continuum concept would be implemented using the cloud-native microservices-based approach, where the microservices would be treated as independent network functions.
- Principle 10, since the computing continuum concept can also help to minimize the environmental footprint and enabling more sustainable networks: the optimized deployment of network service components on already existing edge and extreme-edge resources can help to prevent the commissioning of additional infrastructure resources in centralized data centres; also, processing big amount of data locally on those edge and extreme-edge resources can help to reduce the energy consumption associated to the data transmission.

c) In what regards sub-enabler 5.3 (Federated orchestration system):
- Principle 1, as the mechanism proposed for establishing service federation leverages an overlay network that allows providers like Capability Operators (COP – see section 2.3) to expose capabilities to partner networks in order to facilitate the migration and scaling beyond the capabilities of the initial provider.
- Principle 2, since this mechanism fully automates the negotiation and handshaking process of establishing SLAs before a service is deployed on a partner network. The latter allows for full zero touch automation of the 6G network.
- Principle 3, by allowing dynamic connections to third party networks with myriad capabilities, this enabler enhances flexibility to different network scenarios and topologies. It also allows for new service endpoints to be established at run-time.

- Enabler 6 (AI/ML algorithms) is aligned with the following principles:

a) In what regards sub-enabler 6.1 (AI/ML-based control algorithms for sustainability):
- Principle 2, since the integration of AI/ML algorithms for control allows for a full zero-touch automation of the orchestration and management operations by removing the need for manual intervention. It also optimizes operation efficiency and KPI metrics through improved quality of decision making.
- Principle 3, since AI/ML algorithms, and particularly Reinforcement Learning can adapt their decision policies to environment changes in the network and services, which leads to increased flexibility, and support for different scenarios.
- Principle 10, as AI/ML-based control algorithms with sustainability goals contribute to minimizing energy consumption and environmental footprint, and enabling a sustainable 6G network.

b) In what regards sub-enabler 6.2 (Trustworthy AI/ML-based control algorithms):
- Principle 5, by protecting the AI/ML decision system against adversarial attacks, the developed defence mechanisms contribute to the system's resilience and availability.
- Principle 6, since AI/ML model privacy defence measures protect sensitive data against theft and leaks, and help protect the system's security and privacy.

- Enabler 7 (Network digital twins creation mechanisms) is aligned with the following principles:
  - Principle 1, by providing a Network Digital Twin (NDT) designed to support the efficient training of AI/ML models for enabling 6G services and capabilities.
  - Principle 2, regarding the full automation of the system by integrating the NDT into the optimization loop of the system and AI/ML decision models, and helps further optimize decision making by implementing a close replica of the network and its behaviour.

- Enabler 8 (Real-time zero-touch control loops automation and coordination system) is aligned with the following principles:

- Principle 2, since it provides the fundamental building blocks for the automation of the network through Closed Loop (CL) functions (monitoring, analysis, decision, execution), together with mechanisms for their provisioning and orchestration in the overall network management system. CL functions can be customized for different domains, technologies and objectives, including self-healing, self-configuration and self-optimization, and they may adopt AI/ML techniques at the analysis and decision stages, reducing human interventions in the network operation. Also, based on the coordination mechanisms incorporated in this enabler, it allows to automatically handle the interdependencies between re-configuration or re-optimization actions with impact on multiple layers, domains, tenants, etc.
- Principle 4, through the capability of the CLs to automate re-configuration and re-optimization actions in complex and large-scale scenarios, which would be too complex for human interventions only. Also, this enabler provides a joint coordination logic, with peer-to-peer or hierarchical CLs able to cooperate together to achieve more scalable and end-to-end optimization.

2.3 Mapping with the 6G stakeholders

This section elaborates on how the multiple stakeholders envisaged for the future 6G networks would interact with the M&O framework. According to the information in [HEX22-D62] and [HEX223-D22], those stakeholders would take the following roles[1]:

a) Those already considered for the 5G system [5GP21], represented in Figure 2-4. A main role in this figure is the Service Provider (SP), noted as (1), which interfaces with the Service Customers (top) and obtains and orchestrates resources from Network Operators (2), Virtualisation Infrastructure Service Providers (VISP) (3) and Data Centre Service Providers (DCSP) (4) (collectively referred to as Infrastructure Providers). The SP comprises also the roles of Communication Service Provider (CSP) (5), in charge of the activities for offering traditional telecom services, Digital Service Provider (DSP) (6), entailing the activities for offering digital services such as enhanced mobile broadband and IoT to vertical industries, and Network Slice as a Service (NSaaS) Provider (7), in charge of the activities for offering a network slice along with the services that it may support and configure. Regarding the Network Operator role (2) it would be in charge of operating the programmable network infrastructure, which was considered a shift towards 5G compared with the role of this actor in previous generations, spanning from the radio and/or fixed access to the edge, transport and core networks, and including also the operation of virtual resources leased by other Infrastructure Providers through appropriate APIs. To this end, clearly distinct new roles defined here were the VISP (3), which offers virtualised network or cloud/edge computing resources available through APIs, and the Data Centre Service Provider (4), which offers raw computing resources. Also, the model considers a high interaction between the so-called pure IT and Systems' roles, namely the roles of hardware (HW) and software (SW) suppliers (13) and Operation Support Providers (12), and the roles of 5G resource provisioning, (1) to (11), as presented in Figure 2-4 [5GP21].

b) Those considered in [HEX22-D62], where a new split was done in the Network Operator category (2) in Figure 2-4, namely:
    a. Public Network Operators, representing the traditional telecommunications companies that provide mobile network services to the general public, including consumers and businesses. This would correspond to what was designated as just Network Operators in Figure 2-4.
    b. Non-public Network Operators, referring to those actors operating mobile networks, but not providing services to the general public. This would refer to those organizations enabling and managing their own mobile networks for internal use rather than offering commercial services to external customers.
    c. Satellite Operators (regular or virtual), leveraging the integration of NTNs in 6G infrastructures. They are envisaged to be involved in the whole value chain by providing an alternative network access, extending the coverage towards remote areas, or providing backup connectivity for the backhaul/midhaul segment to guarantee the reliability of the E2E connection.

[1] Identified roles could be shared between one or more stakeholders, which would assume the management of relevant interfaces at business and technical level.

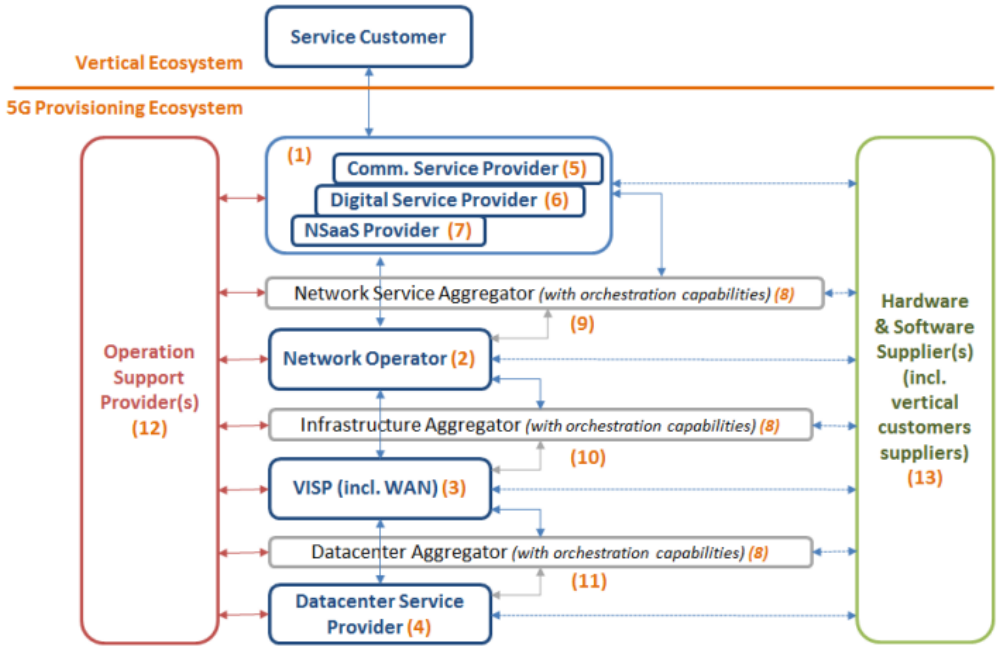


Figure 2-4: Roles in 5G provisioning systems [5GP21].

c) A new role belonging to the Hardware and Software Suppliers category (13) in the previous Figure 2-4, designated as AI/ML Software Provider [HEX22-D62].

d) The new 6G specific roles mentioned in [HEX223-D22], namely[2]:
   a. Capability Operator (COP), as a decomposition of the already existing Network Operator role that would oversee the E2E service management within the operator administrative domain, integrating the different available capabilities (e.g., Network, Cloud, AI, applications, etc.). COPs are envisaged to operate those entities/elements in charge of managing the available domain resources, as well as handling the control and management elements, such as a Network Slice Management Function (NSMF), or other domain specific managers.
   b. Resource Provider, another decomposition of the same Network Operator role that would offer different domain resources, such as applications, AI, network (including RAN, transport, and core), and cloud resources (extreme-edge, edge, and cloud).
   c. TechCos [STL23], considered a step forward with respect to regular Tier-1 telco operators (referred as just 'Telcos' in [HEX223-D22]), and having a wider scope on service offerings, as well as further flexibility on the composition and operation of managed resources. TechCos are expected to articulate their systems with much more granularity, breaking functions into its fundamental components (microservices), each representing stand-alone capabilities that can be individually programmed and chained on a per service/use case needs. Also, their service offering is envisaged not to be limited to communication services, but to other digital services (e.g., Web3, big data, or security services) and beyond communication services (e.g., AI, or applications) offered to new customers (e.g., aggregators) through APIs, in line with the DSP (Digital Service Provider) stakeholder definition already in [5GP21]. However, in the new 6G context, the DSP is also envisaged as the stakeholder in charge to receive intent-based requests from the Service Customers and manage them to achieve the right translation into a set of specific requests for a selected set of the available Capability Operators.
   d. In line with the TechCos approach, Service Customers already in [5GP21] are renamed as Digital Service Customers (i.e., tenants), which in turn are divided into three categories:
      i. Aggregator Tenants, which would include actors such as Hyperscalers, Marketplaces, Telco Consortium, etc., following the Business-to-Business-to-X (B2B2X) model and offering intent-based services to other tenants.

[2] [HEX223-D22] also refers some of the roles already in Figure 2-5, namely: Network Operator (NOP), Communication Service Provider (CSP), and Digital Service Provider (DSP).

ii. Business-to-Business (B2B) Verticals. This type of tenants would group either vertical industries following a B2B model (e.g., Extended Reality (XR) providers) or Application Service Providers following a Business to Business to Consumer (B2B2C) model.
iii. Other Verticals having direct access to the offered intent-based services.

Considering all this, the roles in the 5G system presented in Figure 2-4 would be updated to include the new roles envisaged towards 6G, as shown in Figure 2-5.

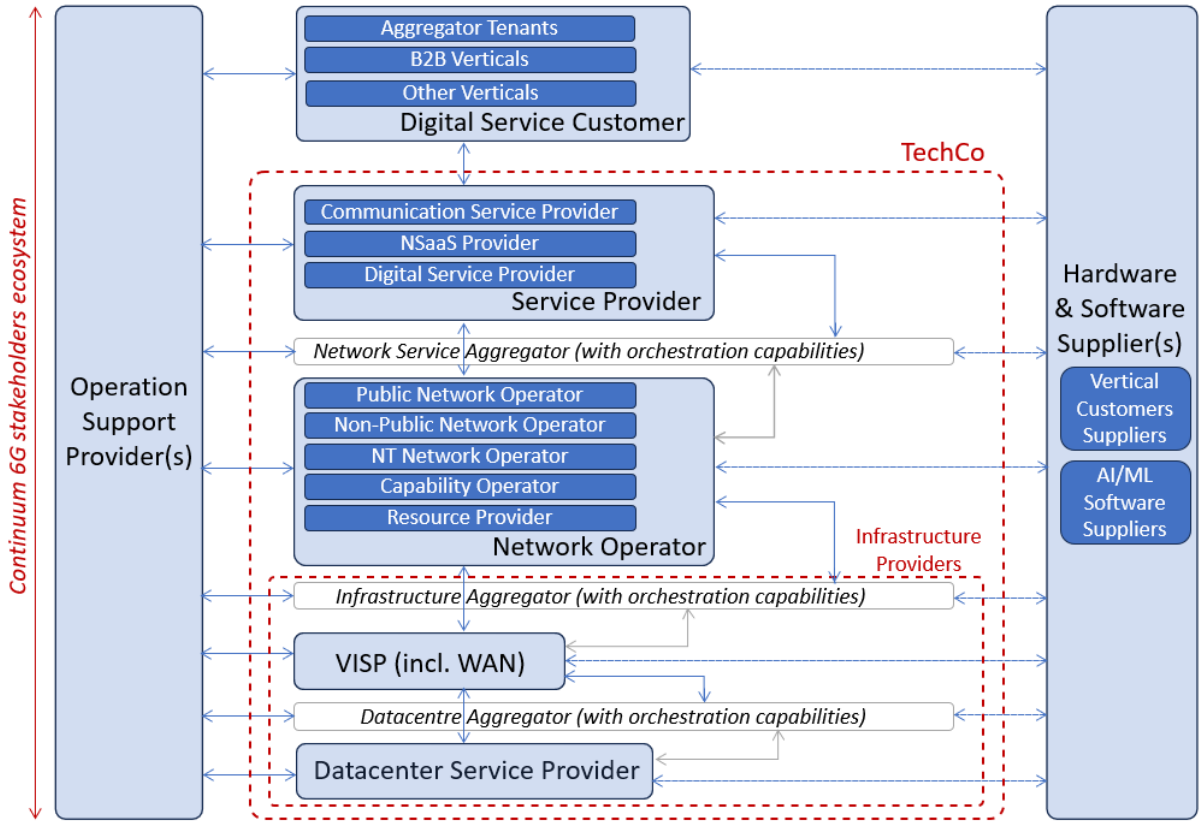


Figure 2-5: Roles in 6G ecosystem.

Besides aggregating the different roles mentioned above, the main update in this figure compared to Figure 2-4 is the deletion of the "boundary" line separating the Service Customers in the Vertical Ecosystem scope from the Provisioning Ecosystem. This is intentional, trying to illustrate the "continuum" orchestration concept towards 6G. The network operator (designated as *TechCo* now) is seen now as part of the network continuum, just like the other stakeholders in the ecosystem. The idea behind this is those vertical industries, or even end-users, will no longer be mere network service consumers, but also able to perform some of the roles that in previous generations were exclusive to the network operator, even including the deployment and the operation of the network services. Besides, those network services can be also decomposed to be deployed on heterogeneous network resources made available by different infrastructure providers, spanning on different technical and administrative domains across the network continuum (core, edge, and extreme-edge), and using cloud-native techniques for this purpose.

Below some examples considering possible interactions among different stakeholders based on this approach:

a) A Software Supplier could automatically deploy (using cloud-native Continuous Integration and Continuous Deployment (CI/CD) techniques) a network service (or a subset of network service components) on the infrastructure resources (e.g., in a specific staging environment) provided by a TechCo and/or other Infrastructure Providers. The Software Supplier could also monitor certain parameters once the service is deployed, to automatically trigger its re-deployment (e.g., based on certain monitoring parameters) or to perform other service life cycle management actions (e.g., the service re-configuration).
b) A Software Supplier could request data to an AI/ML Data Supplier and to a TechCo, in order to train an AI/ML model that would be part of a network service to be deployed on the infrastructure belonging to that TechCo.

c) A TechCo could deploy a network service consisting of multiple microservices on infrastructure resources belonging to different network domains: on its own domain (e.g., on its core network), on certain infrastructure resources provided by a vertical industry (acting as both, Infrastructure Provider and Digital Service Customer), and other additional infrastructure resources at the extreme-edge belonging to end-users (also acting as Infrastructure Providers).
d) A Vertical Industry could act not only as a Digital Service Consumer, but also as Software Provider: certain microservices deployed on its own domain could be part of a network service that, as a whole, would be deployed through the network using the resources of different Infrastructure Providers, and that would be operated by the TechCo, or by a third-party Operation Support Provider. This use case is considered especially relevant to integrate the vertical industries with the operators, without this meaning the usage of specific software provided by the operator.
e) A hyperscaler could act as both Service Provider and Infrastructure Provider, and could request the deployment of RAN service components on a TechCo that would act as Infrastructure Provider as well.
f) Complementary to the previous, a TechCo could request the deployment of certain AI/ML service components on the infrastructure provided by a hyperscaler (e.g., to enrich a network service with certain image recognition or natural language processing capabilities).

The examples list could continue. However, the key idea here is that, as can be appreciated, contrary to what occurred in previous generations, the role-stakeholder relationship is not so strictly defined. Taking advantage of the cloud-native technology, network services can be decomposed and deployed throughout the entire network continuum using agile DevOps techniques, enabling each stakeholder (including the operator) to perform different roles, both in terms of the network services provisioning and their execution.

2.4 Innovations in Hexa-X-II M&O framework

The main innovations envisaged for the Hexa-X-II M&O framework are those derived from the WP6 enablers presented in the previous Section 2.1. Below are the main innovations associated with each of these enablers.

Enabler #1 Network programmability framework.
Broadly speaking, this enabler is about including Software Defined Networking (SDN) technology in the 6G architecture. Although the SDN technology itself is not something new (it is a key enabler already for the 5G technology) it is considered that SDN should continue being part of the next generation of the mobile technology as well, in order to make it possible to interact with the network infrastructure in a programmable and flexible manner, and also based on well-defined standards. It is in this regards that R17 3GPP documents consider Transport Network as part of xHaul network and a dedicated sub-slice manager is introduced for the Transport Network. Thus, some innovations associated with this SDN technology are also considered towards 6G. The main ones are:

- Integration of Transport Networks in 6G through Transport Network - Network Slice Subnet Management Functions (TN-NSSMFs).
- Its alignment with the cloud-native approach, in line with the overall architectural design approach already posed in the previous Hexa-X project.
- The design of interfaces for new devices, in order to support network disaggregation, where network elements such as routers are divided between whiteboxes and network operating systems (NOS). New support for control and management of multiple NOS through OpenConfig or gNMI (gRPC Network Management Interface) protocols.
- To align the SDN architecture with the cloud continuum concept, starting with ease of integration of transport network in Multi-Access Edge Computing (MEC), considered as an example of edge architecture.

Enabler #2: Monitoring and telemetry framework.

The goal of this enabler is to develop an innovative Monitoring and telemetry framework architecture for future 6G networks that is both scalable and driven by data and events. This architecture focuses on advanced monitoring and telemetry systems and also includes features for tracking energy usage. Additionally, it supports the integration of diverse data sources, facilitating data fusion. This approach aims to enhance the

efficiency and effectiveness of network operations, enabling more sophisticated management and optimization of resources in 6G environments.

While Network Data Analytics Function (NWDAF) in 5G focuses on network-specific data analytics, the proposed 6G enabler expands this concept by introducing the following innovations:

- Enhancing scalability
- Diverse data types integration (including energy usage data and extreme edge monitoring and telemetry).
- Data and event driven architecture to provide instant access to alarms and event alerts.
- Ease of integration with closed loop mechanisms for network automation.

This evolution reflects a shift towards more comprehensive network management tools that can adapt to the increasing complexity and demands of future telecommunications networks. The move towards an event-driven architecture in 6G suggests a response to real-time changes and conditions, which is critical for managing the dynamic environments expected in next-generation networks.

Enabler #3: Management capabilities exposure framework.

This enabler proposes an implementation of the Integration Fabric inspired by the ETSI ZSM specification [ZSM-002]. It basically provides two main innovations:

- Dynamic registration and discovery of network elements or services, following a plug-and-play concept[3]. This innovation streamlines the operational aspects by automating the process of registering and discovering new network elements or services within the system. By adopting a plug-and-play approach, it reduces the need for manual intervention, thus minimizing operational overhead and allowing a quick integration of new services and functionalities. This not only enhances operational efficiency but also facilitates rapid service innovation, a crucial aspect in the dynamic landscape of 6G deployments.
- The potential of being implemented in cloud-native environments, especially in scenarios involving several stakeholders. This feature considers the changing operational requirements of 6G networks, particularly in situations where a variety of stakeholders operate together. The enabler provides interoperability with cloud-native architectures, which guarantees resource efficiency, scalability, and flexibility in operations. It supports dynamic workload orchestration and resource allocation, giving operators the ability to manage resources across distributed settings with efficiency. This feature is necessary to maximize resource utilization and optimize operating expenses, which will improve the overall operational performance.

This enabler was present in the Hexa-X architecture [HEX22-D62]. Its tasks included handling exposure blocks and API administration. Hexa-X-II comprises a prototype implementation and extends the use of the previously proven concepts to multi-site settings.

Enabler #4: Security and trustworthiness framework

A security and trustworthiness framework is essential for the successful deployment and operation of 6G technology. It will ensure the protection of data, maintain the integrity of communications, support regulatory compliance, and foster trust among users and stakeholders. This enabler is split into 3 sub-enablers, namely:

- Sub-enabler #4.1: Third-party resource control separation enabler. This sub-enabler is intended to define the scope and impact of the “tenancy” concept in an M&O system. The proposed innovation goes beyond the static, manually configured Role-Based Access (RBAC) and assisted with Lightweight Directory Access Protocol (LDAP) solutions that are being used. This sub-enabler deals with the definition of a granular access control solution that targets authentication, authorization and auditability, ensuring that tenants can be provisioned with tailored management spaces where the permissions that characterize the readable/writable attributes are built in the model of the managed resources and services, avoiding conflict between tenants on resource sharing environments.

[3] The term 'plug and play' is used to describe an architectural approach where software components or modules can be added, removed, or replaced within a system without requiring significant modifications to the existing codebase [WAC08].

- Sub-enabler #4.2: User-centric service provisioning system. This sub-enabler covers the tenant subscribers provision based on a personalized service definition according to their preferences. This sub-enabler will mean innovations in service fulfilment and service activation stages (with the definition of customized policies based on UE Route Selection for accessing subscribed services and keeping protection of user's data), and also in service assurance stage (enriching Service Level Agreement (SLA) with trustworthiness KVIs and exploring how closed loop automation solutions can be reused and adapted to fulfil SLA assurance activities).
- Sub-enabler #4.3: Trust management system. This sub-enabler proposes a trustworthiness estimation procedure of compute nodes and infrastructure components through trust evaluation function. It also aims to estimate the trustworthiness level of communication paths that hold sensitive data workloads as they transit the network. Cloud orchestration engines use the output of the trust evaluation functions, among other factors, to allocate workloads into compute nodes, targeting maximum trustworthiness and efficiency.

Enabler #5: Synergetic orchestration mechanisms for the computing continuum

The term computing continuum refers to resources that span different parts of the programmable infrastructure from Internet of Things (IoT) devices to extreme edge devices to edge and cloud computing infrastructure [DPD23]. The term orchestration mechanisms for the computing continuum refers to solutions for management of the deployment and operation of service graphs across various resources in the continuum [DPD23] [PDM23]. Solutions can be centralized or federated, depending on the applied orchestration scheme. Various types of synergies are considered, either between agents managing various parts of the continuum, or between different stakeholders within a 6G ecosystem (e.g., network providers, cloud providers, OTT players).

This enabler is split into 3 sub-enablers, each of them targeting the orchestration mechanisms for the computing continuum from a different perspective. Below the innovations associated to each one:

- Sub-enabler #5.1: Multi-agent systems for multi-cluster orchestration. This sub-enabler focuses on the proper abstraction and management of multi-cluster resources that may span the computing continuum. With the term multi-cluster resources, we refer to compute resources that can host virtualized network and application functions in the various parts of the continuum. In a 6G E2E system, the deployment of distributed services and the provision of distributed applications in multiple clusters is envisaged to better satisfy QoS, privacy and security needs. Integration of IoT and edge computing technologies is examined, where computational resources may be offered by IoT/edge devices, while end-to-end network management has to be supported. The sub-enabler also introduces mechanisms that follow a "system of systems" approach in orchestration, considering hierarchical decision-making principles. With the term "system of systems" we refer to an approach where the overall responsibility is assigned to a centralized entity, while the control is distributed across various entities [NLF15]. The objective is to introduce distributed intelligence and autonomy characteristics and enable the optimal management of virtualized compute resources in cases where the deployments are made over multi-cluster environments. The combination of techniques coming from multi-agent systems and ML is envisaged for the development of synergetic orchestration mechanisms. For instance, multiple agents may collaborate for managing autoscaling of functions for an overall application/service graph, taking advantage of Reinforcement Learning techniques. Synergies may be also applied between different stakeholders in a 6G ecosystem. For instance, interaction between network providers and over the top (OTT) players such as Service Consumers and Digital Service Providers.
- Sub-enabler #5.2: Decentralised orchestration system. This sub-enabler considers the M&O mechanisms for the computing continuum, focusing mainly on integrating the extreme-edge domain. The targeting of this extreme-edge domain was already introduced in the previous Hexa-X [HEX22-D62] project, meaning that domain beyond the MNO own domain, and including even the end-user devices. The objective is to integrate that domain as part of the network continuum, deploying network service components on it, which is considered quite challenging, since the infrastructure resources in that domain can be highly heterogeneous, they can be asynchronously connected or disconnected (they are not in well controlled premises), they can be mobile devices, or devices with limited computing capabilities. Besides, the extreme-edge domain can be huge, with millions of devices, on a cloud native scale. This sub-enabler is associated with the work regarding virtualisation and the cloud transformation studies, already introduced in [HEX223-D22], [HEX223-D32], and [HEX224-D33].

- Sub-enabler #5.3: Federated orchestration system. This sub-enabler provides enhanced flexibility in the provisioning of services across domains by automating the establishment of dynamic SLAs between them. Services can therefore be scaled up beyond the installed capacity of a provider leveraging the resources of allied providers communicating over a secure blockchain network. The corresponding negotiation and service extension or migration procedures would be handled by aptly designed smart contracts.

Enabler #6: AI/ML algorithms.

This enabler builds on top of the AI/ML framework already provided in the previous Hexa-X project [HEX22-D62] but provides specific algorithms targeting sustainability and trustworthiness, which are considered two of key KVIs. The enabler is split into two sub-enablers, targeting each of these aspects. Below are the innovations related to each of these sub-enablers.

- Regarding sub-enabler 6.1 (AI/ML-based control algorithms for sustainability), this enabler brings several innovations over the 5G technology, namely:
  - Energy-efficient configuration recommendations based on high-level requirements related to energy efficiency that are provided as input, then translated into decisions and configurations to be implemented by the 6G system in order to satisfy the expressed requirements. This helps further automate the M&O of the 6G system compared to 5G by adding an abstraction layer for the consumer that only has to provide high-level requirements instead of detailed configurations.
  - Sustainability targets in optimization processes of network and service M&O, where AI/ML algorithms are trained to optimize energy consumption and carbon footprint while satisfying the performance requirements. This helps realize a performant but also sustainable and energy-efficient 6G system, when AI/ML optimization in 5G systems was mostly focused on performance.
  - Adding energy spent in ML training to the overall energy consumption and optimization process. Indeed, AI/ML model training consumes energy, particularly when the model is complex and computation intensive. Thus, tracking and optimizing energy consumption during all phases of the MLOps pipeline can help to further reduce the overall energy usage for orchestration and management operations in 6G.
  - Energy footprint driven federation of orchestration domains, by optimizing the clustering of federated edges to reduce the communication load and energy cost, which minimizes the overall energy footprint.
- Regarding sub-enabler 6.2 (Trustworthy AI/ML-based control algorithms), the main innovation regarding this enabler is to improve the robustness, privacy levels and explainability of AI/ML models in network and service M&O against adversarial attacks meant to influence model decisions for M&O, or privacy attacks for accessing sensitive data. This results in a more trustworthy, robust and resilient AI-based 6G M&O system with assured performance.

Enabler #7: Network digital twins creation mechanisms

This enabler is intended to develop a general framework for the use of NDTs in the network and the service M&O. In 5G systems, AI/ML models for M&O are generally trained using input from datasets, or simulators. However, real data from operational networks is difficult to obtain due to data privacy regulations, while networks simulators often fall short when mimicking real network behaviour, which results in sub-optimal models. This enabler helps bridge that gap by providing mechanisms for generating Network Digital Twins that provide an accurate representation of the network state in real time while taking into account the virtualization aspects, which allows for more efficient AI/ML models for M&O in 6G systems.

Enabler #8: Real-time zero-touch control loops automation and coordination system.

This enabler provides mechanisms for the automated provisioning and the lifecycle management of closed loops (CL) for zero-touch network automation. CLs can be specialized to operate in different layers (infrastructure, network, service) and domains (edge/cloud, access, core, transport) and to control particular aspects of dynamic elements like network slices and services. Closed loops are delivered as a set of cloud native functions, fully programmable and possibly assisted by AI functions. They are managed through the CL Governance as an integrated step of service and network provisioning lifecycle, so that their deployment and configuration is fully associated and jointly managed with the entities they automatically control. Moreover, the same orchestration mechanisms adopted to optimize provisioning of network functions and services can

be applied to CL instances, with resource allocation algorithms, placement strategies and scaling or migration procedures customized for each CL. This approach is aligned with the closed loop governance defined by ETSI ZSM [ZSM-009-1] and an example prototype will be implemented in the project. As a new feature with respect to CL management in 5G networks, the main innovation proposed for 6G is related to the coordination of multiple and concurrent closed loops, that may operate in different time scales and work at different layers, but with interdependencies that impact the same resources or services. CL coordination allows to resolve issues related to potential conflicts among their decisions, to schedule the execution of their actions, to identify possible issues in the combination of actions proposed by different closed loops and to find solutions to arbitrate and negotiate among contrasting decisions. Moreover, coordination techniques among hierarchical closed loops allow more scalable network automation, among different domains, through delegation and escalation procedures. Finally, collaboration among peer closed loops allows to share knowledge and insight on different network domains or services, allowing for more extensive network data analysis and more effective actions via multi-objective decisions.

2.4.1 Enhancements to the Hexa-X architecture

Hexa-X-II is the continuation of previous 5G-PPP Hexa-X project [HEX24], whose main results are also considered towards designing the E2E 6G system design. The main novel capabilities for the future 6G M&O systems identified in that previous project were the following ([HEX22-D62] – Section 5.3):

a) **Unified orchestration across the "extreme-edge, edge, core" continuum**, considering that the end-user devices can contribute to provide additional infrastructure resources which can leverage the deployment of innovative 6G services.
b) **Unified management and orchestration across multiple domains, owned and administered by different stakeholders**, characterised by heterogeneous technologies, platforms and management systems with their own specific interfaces. This feature involves the definition of converging interfaces, as well as the mechanisms to dynamically register and expose the resources and capabilities offered from each domain, including also federation strategies.
c) **Increasing levels of automation** in the functionalities of service and network planning, design, provisioning, optimisation, operation, and control, leveraging on closed-loop and zero-touch solutions that may strongly reduce the required manual interventions. This innovation also involves the continuous monitoring of different aspects of the network and the services performance, so that the M&O system can be able to automatically identify, detect or predict potential issues, failures, bottlenecks, or inefficient configurations, to trigger and coordinate dynamic reactions. This would also be enabled through the extensive programmability of networks and their computing resources.
d) **Adoption of data-driven and AI/ML techniques in the M&O system**, supporting frameworks for distributed and collaborative AI, AI/MLOps, pervasive monitoring of service and network KPIs, with support for scalable data and trained models sharing along the "extreme-edge, edge and core" continuum in multi-stakeholder environments. The scope of AI/ML techniques would also cover several optimisation aspects, as well as the lifecycle actions regarding the services M&O, including the resource allocation and the slices sharing at provisioning time, services composition, scaling, migration, re-configuration, and re-optimisation of the network functions and their related resources.
e) **Intent-based approaches for service planning and definition**. The M&O system would implement automated mechanisms for translating service specifications and commands based on intents, which could be expressed even in natural language.
f) **Adoption of the cloud-native principles in the telco-grade environment**, involving three aspects: (i) the usage of micro-services for implementing the network functions, (ii) the implementation of service meshes, targeting to optimise the communication between applications and to reduce downtimes by using a specific built-in infrastructure layer, and (iii), the enabling mechanisms for the network services to be deployed and updated using DevOps practices, by implementing CI/CD pipelines with a very high automation degree. This third aspect was considered innovative in the telco-grade environment, involving the fact of putting together development and operational teams, which could be challenging in this scope, where services development is typically a collaborative effort involving multiple external software vendors.

In the following we describe how the Hexa-X-II WP6 enablers are related to these innovative aspects identified in the previous Hexa-X project.

Enabler 1: Network programmability framework.

This enabler is related to innovation (c) stated in Hexa-X, and specifically, in what regards the mention about the extensive programmability of networks and their computing resources. However, the work done in Hexa-X regarding this is more specific, targeting the implementation of an SDN controller towards 6G, based on TeraFlowSDN [TER24] controller. So, beyond the theoretical approach outlined in Hexa-X, the work in Hexa-X-II is more specific and tangible, contributing to the project with a specific PoC, and also, to the Open-Source community, since TeraFlowSDN is an Open-Source ETSI hosted project [TFS24].

Enabler 2: Monitoring and telemetry framework.

This enabler is directly related to innovations (c) and (d) stated in Hexa-X, although it can be also related to innovations (a) and (b), regarding the M&O over different technical and administrative domains. In fact, monitoring can be considered a kind of pervasive functionality, in the sense that for the upcoming 6G networks the collection of monitoring data from multiple and heterogeneous sources is considered necessary. Anyway, all these innovative aspects considered in Hexa-X are more oriented to the data gathering and integration problem, while the approach in Hexa-X-II goes beyond data collection, also addressing the immediate reception and the forwarding of network related events, as well as its fusion and processing. Also, a specific scalable architecture is proposed to provide a cloud-scale solution.

Enabler 3: Management capabilities exposure framework.

This enabler is directly related to innovation (b) stated in Hexa-X, and specifically, in what regards the definition of converging interfaces and the mechanisms to dynamically register and expose the resources and capabilities in different network domains. However, although the work performed in Hexa-X already introduced a component called "API Management Exposure" to perform this functionality, the approach there was basically theoretical. Beyond that, the approach in Hexa-X-II is more specific, targeting to implement such functionality by means of a prototype, inspired by the Integration Fabric concept defined in the ETSI ZSM specification [ZSM-002]. This will also convey contributions to ETSI.

Enabler 4: Security and trustworthiness framework.

Although certain security related components were included in the Hexa-X architectural design, as can be seen in the listing at the beginning of this section, that was not considered a main novel capability. In Hexa-X-II the approach consists of introducing three components that are considered key enablers: the third-party resource control separation enabler, a user-centric service provisioning system, and a specific trust management system.

Enabler 5: Synergetic orchestration mechanisms for the computing continuum.

This enabler is mainly related with novel capability (a) from Hexa-X, but also with (b). In this case, the step-forward in Hexa-X-II is different considering the three sub-enablers in this Enabler 5:

- Regarding sub-enabler 5.1 (multi-agent systems for multi-cluster orchestration), the main added value is the development of specific data models and compute resource management solutions targeted to multi-cluster environments. Also, the way that the resources will be modelled will be in accordance with the developed data management schemes defined in WP2. In this way, orchestration mechanisms will be able to consider in a unified way the available resources to schedule the necessary deployment/optimization/re-configuration mechanisms. Synergetic orchestration mechanisms based on the adoption of multi-agent systems and ML techniques will enable the better collaboration among distributed orchestration entities that collaborate towards joint objectives (e.g., assurance of a KPI or SLA for an end-to-end service). This approach has been already introduced in [HEX223-D62].
- Regarding sub-enabler 5.2 (decentralised orchestration system), although it still relies on applying the cloud-native principles (as those mentioned in (f.i) and (f.ii) above), it represents a novel approach to orchestrate resources and services towards 6G, not considered in Hexa-X. It targets the integration of the compute-continuum infrastructure resources for the network services M&O, and with focus on integrating the extreme-edge domain considering its key challenging features (the aggregation of devices beyond the MNO own premises, the diversity of stakeholders in this domain, the high heterogeneity of devices, the

high volatility of those devices, and the size of this domain, that can be huge). In short, the approach consists in (i) delegating the network services provisioning on a new set of distributed network elements, and (ii), to embed as part of the network services themselves the specific set of M&O resources they may need once they are deployed. This approach has already been introduced in [HEX223-D22], [HEX223-D32] and [HEX224-D33], associated to the work regarding virtualisation and the cloud transformation studies.

- Regarding sub-enabler 5.3 (federated orchestration system), this is an innovative approach, not considered in Hexa-X, to the static and slow process of establishing SLAs between service providers is proposed. This allows service continuity in roaming scenarios that may not have been envisaged by providers and as such lack pre-established agreements that in legacy M&O systems led to service disruption.

Enabler 6: AI/ML algorithms.

This enabler is clearly related to innovation (d) in the previous Hexa-X project (Adoption of data-driven and AI/ML techniques in the M&O system). However, while in Hexa-X the approach was in the need of defining a general AI/ML framework to support the M&O functionalities, Hexa-X-II focuses on more specific solutions to address two of the aspects that are considered relevant regarding the usage of AI/ML algorithms: trustworthiness and sustainability. Regarding trustworthiness, this enabler provides specific mechanisms intended to improve robustness, privacy levels, and the explainability of the M&O actions that may be triggered by the AI/ML models. Regarding sustainability, the enabler provides mechanisms to improve certain M&O optimization processes such as resource allocation, function placement, migration and scaling, by adding consideration of energy-related metrics to drive the AI/ML models training processes, among other performance related KPIs.

Enabler 7: Network digital twins creation mechanisms.

This enabler is completely new with respect to Hexa-X, where network digital twins (NDT) were not considered regarding the M&O processes optimization. However, it could be somehow related with innovation (d), i.e., as part of the general AI/ML framework, since those NDTs, as they are addressed in Hexa-X-II, will be created based on AI/ML techniques. The generated NDTs will also be used for training more efficient AI/ML models for optimizing M&O processes. Anyway, all the work regarding this in Hexa-X-II certainly adds value to what was initially stated in Hexa-X.

Enabler 8: Real-time zero-touch control loops automation and coordination system.

This enabler is related to innovation (c) from Hexa-X in what regards to increase the level of automation based on closed-loop zero-touch solutions. However, while in Hexa-X closed loops were considered just as static functionalities, in Hexa-X-II the introduction of the control-loops governance mechanisms allows to deploy the closed-loops in a more customizable and automatic manner over the edge/cloud continuum, as well as adjusting their placement, scaling, and configuration to the dynamicity of the network. On the other hand, Hexa-X did not explicitly investigate the cooperation among multiple closed loops.

In summary, the smart network management enablers in Hexa-X-II are compliant with the main novel capabilities for the future 6G M&O systems identified in the previous Hexa-X project[4], while they go a step further in providing more specific and tangible implementations in some cases, and more elaborated and complete concepts in other cases. Besides, new aspects are also included, such as the usage of network digital twins as part of the M&O processes, the programmability of transport networks through SDN, decentralized orchestration through the network continuum, or more specific and complete security and sustainability related enablers.

[4] Intent management is not reported in this deliverable, since in Hexa-X-II it is addressed in WP2 (ref. [HEX223-D22]). CI/CD is considered out of scope for the project.

3 Enablers for Management and Orchestration

3.1 Enabler 1: Network programmability framework

The solution/enabler for "Network programmability framework" is designed to provide service providers with a flexible, programmable, and scalable approach to network management. This solution is based on several key technologies and approaches, including software-defined networking (SDN) [NG13], application programming interfaces (APIs), and cloud-based network management platforms [VMC+21].

D6.2 [HEX223-D62] presented the SoTA and expected beyond SoTA of this enabler. To this end, work on multiple components was proposed, as well as possible external interfaces. The following subsections present the internal architecture that could be used to implement this enabler, as well as preliminary implementation details, and early validation results.

3.1.1 Enabler design

The proposed high-level architecture for Enabler 1 assumes a single administrative domain and it involves a hierarchical approach with an End-to-End (E2E) controller as the parent controller and technological domain controllers as child controllers. Specifically, it mentions an E2E SDN (Software-Defined Networking) orchestrator and technological domain SDN controllers in the IP (Internet Protocol), Optical, Time-Sensitive Networks (TSN) / Deterministic Networks (DetNet) and other domains.

The Network programmability framework, which acts as End-to-End SDN Orchestrator, is the parent controller responsible for managing and orchestrating the entire network infrastructure using Software-Defined Networking. The E2E SDN orchestrator oversees the coordination and control of the network across different technological domains. The E2E SDN orchestrator can rely on existing solutions, such as ETSI TeraFlowSDN (TFS) [TFS24]. Figure 3-1 shows the hierarchical SDN orchestration architecture proposed and also the internal design of Enabler 1: Network programmability framework. TeraFlowSDN and its components has been described at [VMC+21].

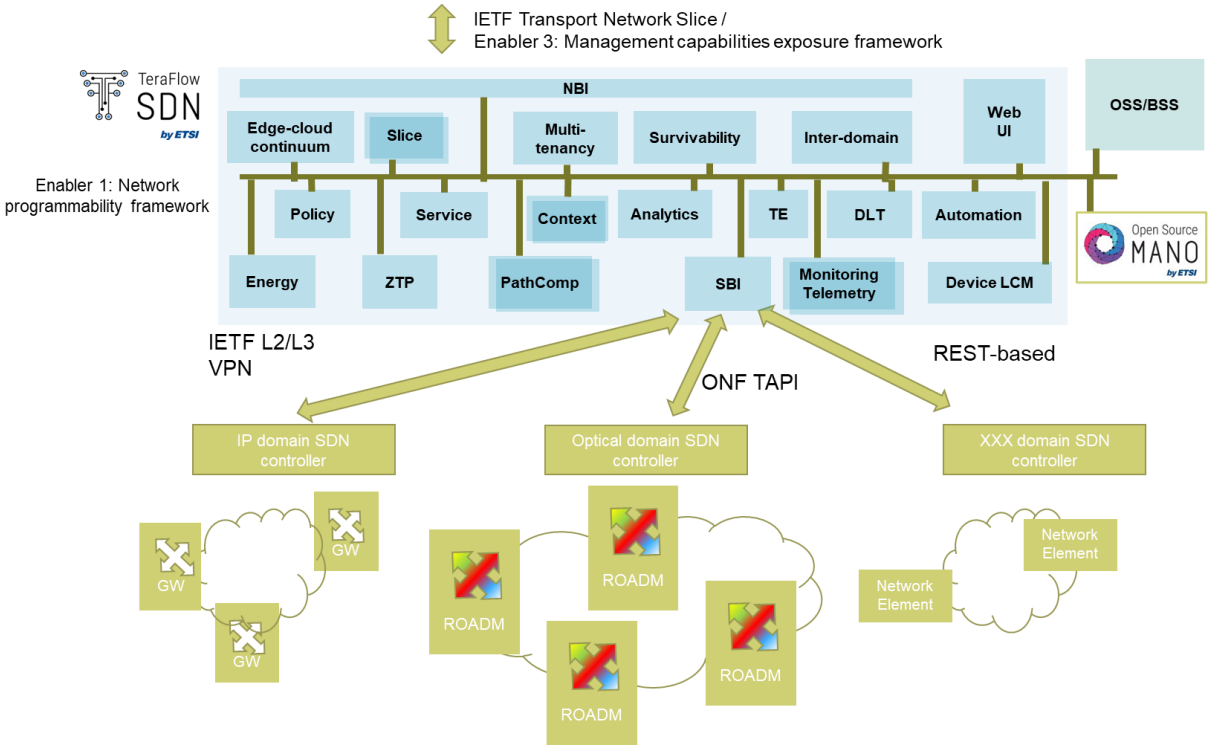


Figure 3-1 Design of Enabler 1: Network programmability framework.

As the parent controller, the E2E SDN Orchestrator assumes the role of overseeing the coordination and control of the network across different technological domains. It acts as a unifying entity that harmonizes the

operations of various network elements, ensuring seamless communication and efficient utilization of resources.

One of the primary objectives of the E2E SDN Orchestrator is to coordinate multi-domain orchestration of network operations. In modern network architectures, different domains, such as data centre networks, wide area networks (WANs), edge networks, and cloud infrastructure, coexist and interact to deliver end-to-end services. The E2E SDN Orchestrator plays a crucial role in integrating and managing network elements from diverse domains. By facilitating the coordination and orchestration of network operations across these domains, the E2E SDN Orchestrator enables consistent policies, optimized resource utilization, and efficient communication throughout the entire network infrastructure.

Furthermore, the E2E SDN Orchestrator supports service-level orchestration. In addition to managing the network infrastructure, it provides mechanisms for defining and managing services that traverse multiple network domains. By considering the end-to-end service requirements, the E2E SDN Orchestrator ensures that services are provisioned and delivered seamlessly across the network. It enables the dynamic allocation and optimization of network resources, allowing services to adapt to changing demands and conditions. The service-level orchestration capabilities of the E2E SDN Orchestrator enhance the overall efficiency, scalability, and agility of the network, providing a flexible and adaptable infrastructure for delivering a wide range of applications and services.

Technological Domain SDN Controllers are child controllers that operate within specific technological domains of the network. In this case, it mentions IP, Optical, Time Sensitive Networks/Deterministic Networks (TSN/DetNet) [MCT22], and other domains, but there may be additional domains depending on the network architecture. Each technological domain controller is responsible for managing and controlling the SDN functions and resources within its respective domain. Each specific technological domain controller might be based on several SNS project solutions, for example PREDICT-6G [PRE24] work could be used for DetNet. For IP SDN controllers, ETSI TeraFlowSDN is also proposed.

Each Technological Domain SDN Controller focuses on a particular domain and is tailored to address the specific requirements and characteristics of that domain. An IP SDN Controller would be designed to manage and control the IP-based network elements, such as routers and switches, within the network infrastructure. It would handle tasks like routing, traffic engineering, QoS enforcement, and network optimization within the IP domain. Similarly, an Optical SDN Controller would specialize in managing and controlling the optical network elements, such as wavelength routers and optical switches. It would facilitate functions like lightpath provisioning, dynamic spectrum allocation, and optical resource optimization. The Optical SDN Controller enables programmability and control of the optical domain, allowing for efficient utilization of optical resources and seamless integration with other domains. Another is Time-Sensitive Networking (TSN) or Deterministic Networking (DetNet). These controllers are responsible for managing and controlling the real-time and time-sensitive communication requirements within the network. They ensure precise and deterministic delivery of critical data, including time synchronization, traffic scheduling, flow registration, resource reservation, and computing end-to-end routes. It remains to be seen whether these controllers can be implemented by building upon the work of research projects such as PREDICT-6G [PRE24]. In addition to the mentioned domains, there can be other technological domains SDN controllers depending on the specific network architecture and requirements. For example, domains like wireless networks, cloud infrastructure, IoT networks, or VNF could have dedicated SDN Controllers to manage and control their respective elements and functionalities. By having technological domain SDN Controllers operating within specific domains, the network architecture becomes more modular and scalable. Each controller focuses on its domain's unique characteristics, allowing for optimized management and control of the associated SDN functions and resources. These controllers work in conjunction with the E2E SDN Orchestrator to achieve a comprehensive and efficient network management framework, providing end-to-end coordination and control across different technological domains.

3.1.1.1 Internal Architecture of the system components

Figure 3-2 presents the architecture of ETSI TeraFlowSDN (TFS), an open-source, microservice-based, cloud-native, and carrier-grade SDN controller. It is designed to integrate with current NFV (Network Functions Virtualization) and MEC (Mobile Edge Computing) frameworks and provides a range of features for flow management at the service layer and network equipment integration at the infrastructure layer. It incorporates

machine learning-based security and PDL (Policy Description Language)-based forensic evidence to support multi-tenancy. The proposed Enabler acts as a parent SDN controller and End-to-End SDN orchestrator. Moreover, the same Enabler can be instantiated to support a specific technological domain as domain SDN controller.

The architecture is structured to support a range of functionalities including Edge-cloud continuum, Slice, Multi-tenancy, Survivability, Inter-domain, Web UI, and OSS/BSS, which are likely to refer to different components or capabilities of the TeraFlowSDN. These include policy management, service orchestration, context awareness, analytics, traffic engineering (TE), Distributed Ledger Technology (DLT), monitoring telemetry, device lifecycle management (LCM), and automation. It also highlights the potential to manage energy consumption across network elements, thus contributing to more energy-efficient network deployments.

The various components of the system interact with external elements such as NFV orchestrator (such as ETSI OpenSourceMANO) [GNG+22], OpenConfig [CVM+19], ONF Transport API [LVU+16], and P4 whiteboxes [FKK+23]. It seems to emphasize a disaggregated service-based architecture that provides smart connectivity services atop advanced programmable networking infrastructures. The diagram also refers to transport network slicing in both the optical and packet domains with associated SLAs and isolation requirements, managed by the Slice component.

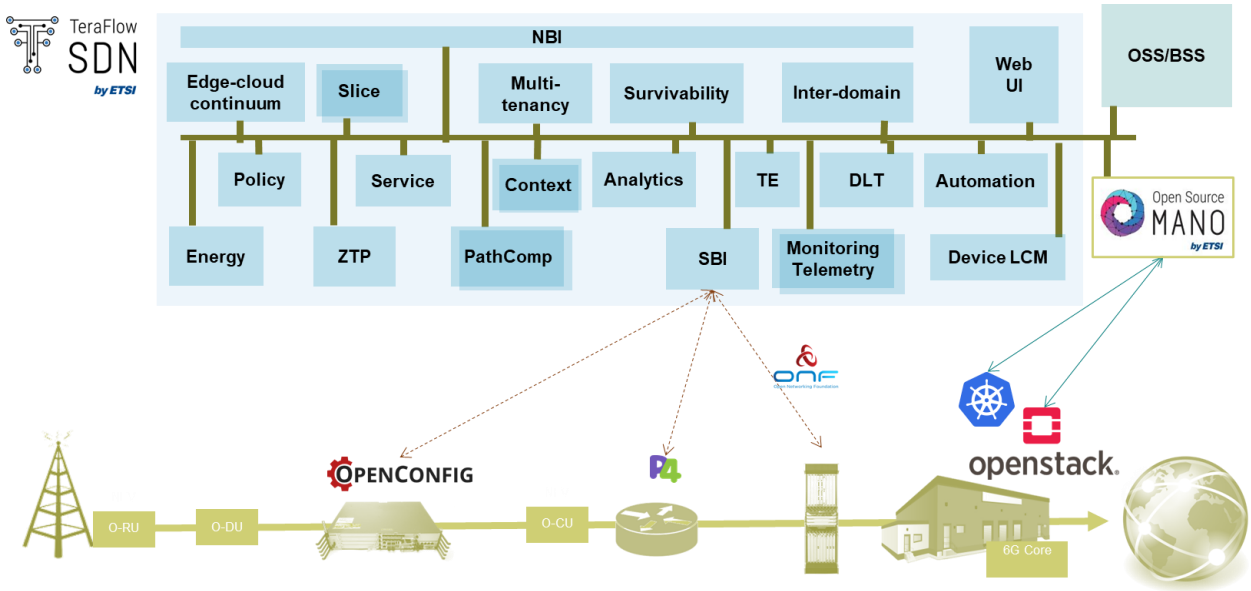


Figure 3-2: ETSI TeraFlowSDN internal architecture.

Due to its cloud-native architecture, core components shall be implemented as microservices (using Java and Python) and might be deployed on a Kubernetes-based environment, using cutting-edge technologies for network management.

3.1.1.2 Workflows

Figure 3-3 outlines a sequence of interactions involving network management components within a software-defined networking (SDN) environment.

The diagram begins with Enabler 3 (management capabilities exposure framework), which initiates a request to TeraFlowSDN (TFS) instructing it to establish a transport network slice. In response, TFS acts as an End-to-End SDN Orchestrator, computes the required multi-layer connectivity services and starts to interact with underlying technological domains SDN controllers in order to provision the requested slice. TFS communicates with the Optical SDN Controller to create a ConnectivityService, which involves iterating over a set of ROADMs (reconfigurable optical add-drop multiplexers) using a REST API or OpenConfig protocol to configure the optical path. Once the ConnectivityService is successfully created, Optical SDN controller informs TFS of the completion. Subsequently, TFS requests the IP SDN Controller (IP) to create an L2VPN service. This process includes configuring network instances and interfaces on various routers through multiple specific commands related to network instances, interfaces, quality of service, and MPLS traffic engineering,

reflected in loops where each router is configured sequentially. After the IP SDN Controller successfully establishes the L2VPN service, it notifies TFS, which in turn informs Enabler 3 that the network slice has been successfully created.

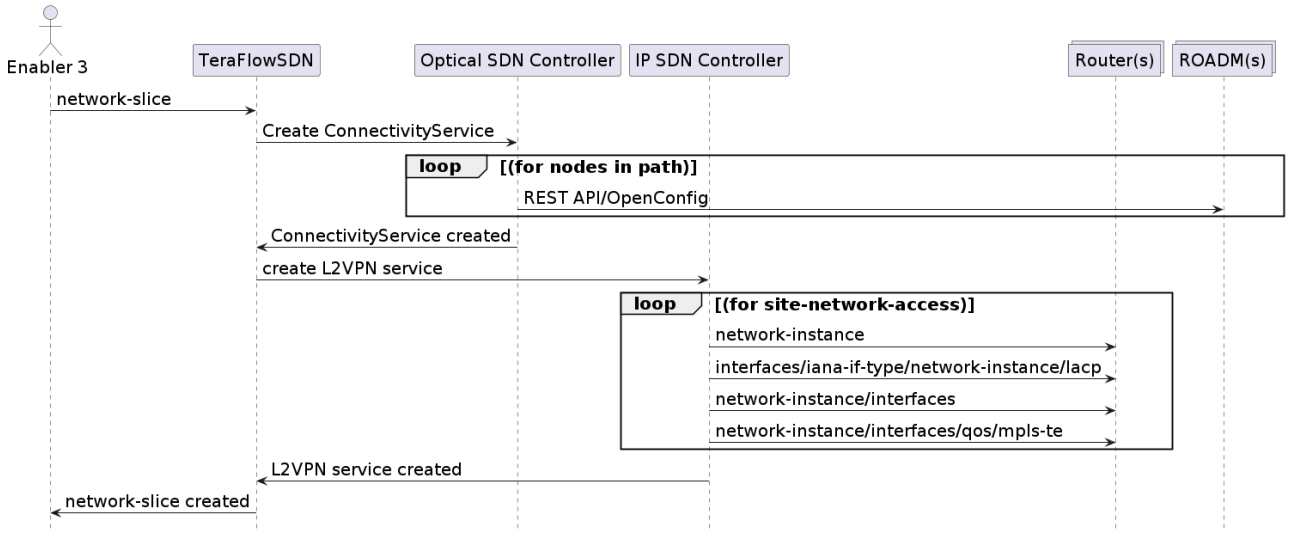


Figure 3-3 Network programmability framework end-to-end workflow for provisioning IETF slice.

The SDN controller interfaces are reported in Appendix 8.2.1.

3.1.2 Preliminary implementation and early validation results

Several preliminary implementations are presented in this section. Firstly, TeraFlowSDN (TFS) using data-plane in-a-box is a novel feature that allows quick demonstration of TFS and its integration as a transport SDN controller with an emulated transport network that can be easily integrated in a system proof-of-concept (section 3.1.2.1). Secondly, SmartNIC Transceiver support with OpenConfig extensions is presented (section 3.1.2.2), in order to provide support for inventory of network interface cards with computing capabilities and support for deployment of specific ML pipelines on top of them. Later, a Time-Sensitive Network and Deterministic Network SDN controller is explored in section 3.1.2.3, followed by automated transport network re-configuration (section 3.1.2.4), and integration of transport network and Multi-access Edge Computing (MEC) (section 3.1.2.5). Finally, section 3.1.2.6 presents the integration of TM Forum APIs with ETSI TFS API.

3.1.2.1 TeraFlowSDN using data-plane in-a-box

TeraFlowSDN (TFS) using data-plane in-a-box is a novel feature that allows quick demonstration of TFS and its integration as a transport SDN controller with an emulated transport network that can be easily integrated in a system proof-of-concept. It will be integrated in System PoCB.1.

Figure 3-4 depicts a schematic diagram of a network architecture integrating various technologies and platforms. At the top left of the diagram, there is a representation of ETSI TeraFlow SDN, indicating a software-defined networking (SDN) controller solution. This connects an "SBI Servicer" on the top right via a gRPC (gRPC Remote Procedure Calls) interface, which is known for connecting services in a language-agnostic way. The SBI Servicer box contains an array of different network control and management APIs (Application Programming Interfaces), such as: OpenConfig, Emulated, gNMI/OpenConfig, P4, Transport API, IETF Network Topology, and IETF L2/L3VPN.

TFS is connected to two emulated network operating system switches, which support OpenConfig interfaces and gNMI (gRPC Network Management Interface) for network operations. CONTAINERlab [CON24] is the proposed lab environment for containerized network functions and might be used here for network simulation or testing purposes.

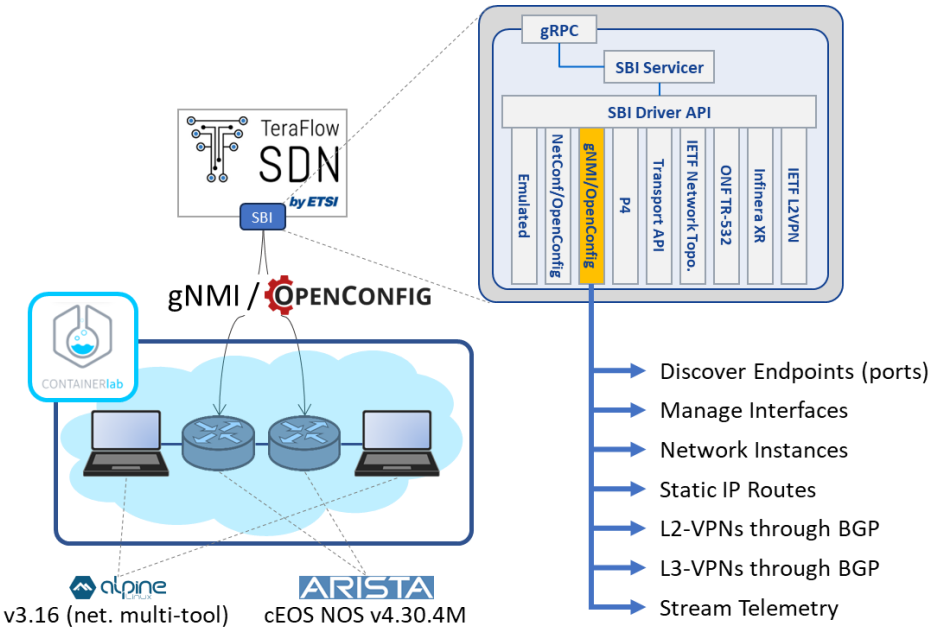


Figure 3-4: TeraFlowSDN SBI driver gNMI/OpenConfig + Dataplane-in-a-box based on ContainerLab.

Figure 3-5 illustrates the high-level architectural diagram of the TeraFlowSDN preliminary implementation to support integration with Enabler 3: Management capabilities exposure framework (described later on in Section 3.3), based on Kafka [KAF24], which is a distributed event streaming platform commonly used for building real-time data pipelines and streaming apps.

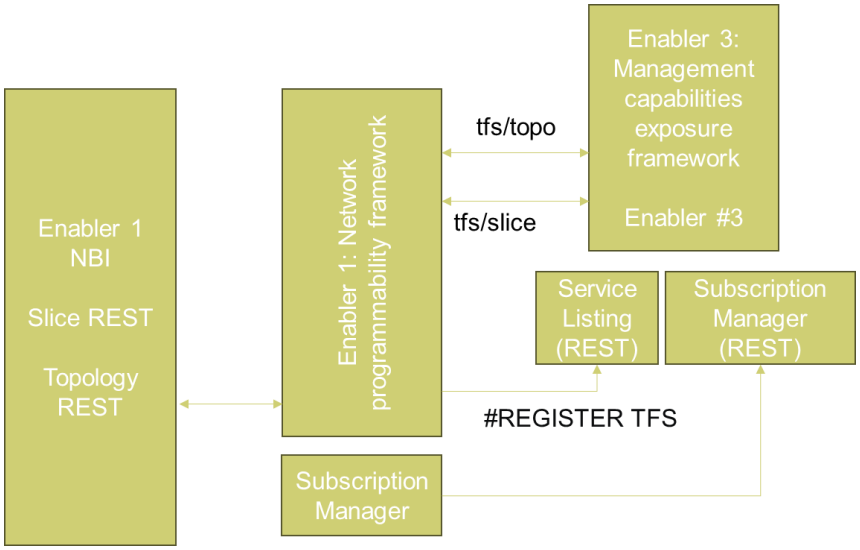


Figure 3-5: TFS integration with Management capabilities exposure framework.

In order to interface with Management capabilities exposure framework, several IETF standards are required to provide the necessary data in JSON and the Kafka topics.:

- YANG Data Model for Traffic Engineering (TE) Topologies (RFC 8795 [RFC8795]).
- A YANG Data Model for the RFC 9543 [RFC9543] Network Slice Service (draft-ietf-teas-ietf-network-slice-nbi-yang-10).

3.1.2.2 SmartNIC Transceiver support using OpenConfig extensions

This implementation has been presented at [VVG+24]. It has been deployed at ADRENALINE Testbed at CTTC premises [MNC+17] and it will be part of future TeraFlowSDN release 4/5. Anomaly detection and mitigation is one of the required novel security features for an SDN controller. To this end, this section presents a proposal for implementation of an SDN-enabled anomaly detection mechanism by extending the SDN controller with support for SmartNICs.

Anomaly detection stands at the forefront of proactive defence strategies, offering the potential to identify subtle deviations from normal network behaviour indicative of malicious activities. Anomaly detection begins with the establishment of a baseline behaviour profile, encapsulating the anticipated patterns and behaviours of data during normal operation [GLZ18]. Anomalous Behaviour Profiling (ABP) involves creating a detailed understanding of deviations from the established normal behaviour. It identifies and characterizes patterns that deviate from the expected, contributing to a comprehensive analysis of anomalies within the incoming data. Then, incoming data undergoes scrutiny. Data points are carefully evaluated to assess their conformity with the expected characteristics outlined in the normal behaviour profile. Any data points that significantly deviate from it are flagged as anomalies. The distributed and programmable nature of network elements requires an orchestrated approach to anomaly detection. For this reason, the SDN controller might need to orchestrate the subsequent mitigation actions across the network infrastructure, once anomaly is detected.

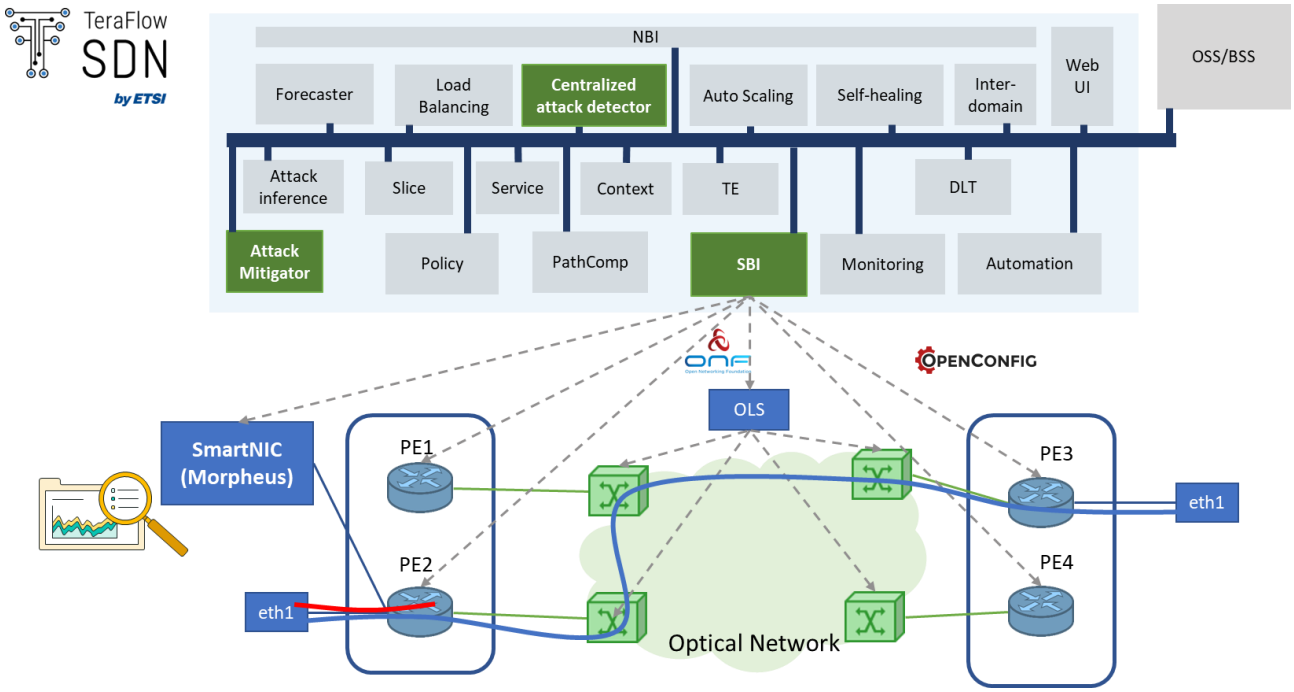


Figure 3-6: Proposed scenario for providing Anomalous Behaviour Profiling.

Figure 3-6 presents the proposed scenario for SDN-based control of SmartNIC and deployment of Anomalous Behaviour Detection using ETSI TeraFlowSDN SDN controller. The threat detection is backed by the NVIDIA Morpheus open application framework [MOR24] that enables cybersecurity developers to create optimized AI pipelines for filtering, processing, and classifying large volumes of real-time data. Firstly, the extension of the TFS context component for SmartNICs incorporates a series of functionalities in ProtocolBuffer format, including information about the manufacturer, model, and serial number, as well as details about their transceivers, Data Processing Units (DPU), and Graphics Processing Units (GPU). For transceivers, their port types and speeds are modelled for each. DPUs include information about their cores, RAM, and memory. Lastly, GPUs are modelled for their architecture, memory, cores, etc. This extended model in TFS context component enables the discovery of the characteristics of each SmartNIC node. After defining the context extension for SmartNICs, an extension of the OpenConfig-Probes data model has been carried out to support the configuration of SmartNICs through the Morpheus agent and Python API. This data model, specified in YANG format, defines a Morpheus pipeline providing various information such as its name, number of threads, pipeline size, input and output files, model name, server URL, and the configuration of various stages within the modelling phase, including deserialization, monitoring, inference, and serialization. This data model has been introduced in the TFS SouthBound Interface (SBI) component in order to directly interact with each SmartNIC. Centralized Attack Detector has been extended to support the necessary closed loop for monitoring anomalous traffic behaviour and react accordingly as described.

Figure 3-7 shows the implemented workflow. The workflow involves 3 main components, the SDN controller, the SmartNIC, and the IP routers that are part of the network. The relevant SDN controller components are the Centralized Attack Detector (CAD), the Attack Mitigator (AM) and the South Bound Interface (SBI). On the

SmartNIC, the two main software components are the Operating System (SmartNIC_OS) managing the SmartNIC and the Morpheus agent carrying on the detection (SmartNIC_M).

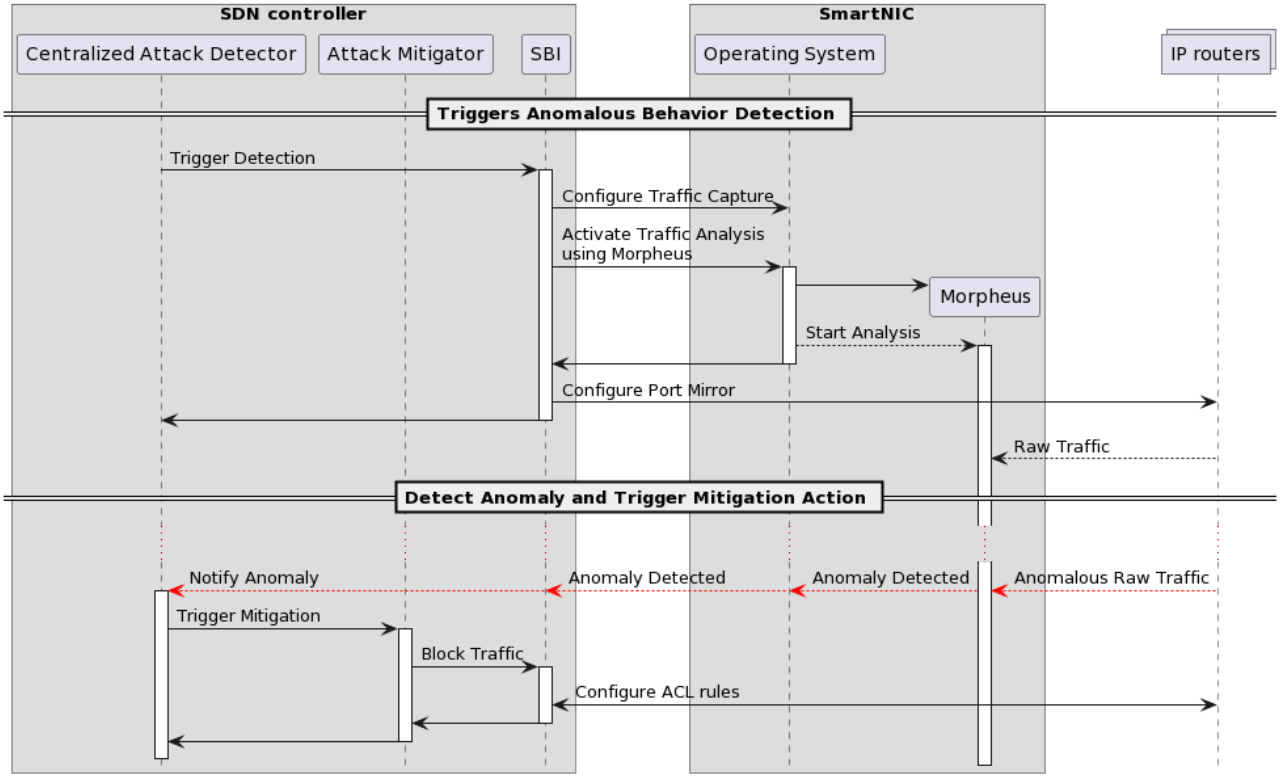


Figure 3-7: Anomalous Behaviour Profiling workflow.

The workflow starts when the CAD, after a new connection has been detected, triggers the configuration of the anomaly detection capacities of the SmartNIC through the SBI component. The SBI component, upon activation, communicates with the SmartNIC_OS to configure traffic capture and subsequently initiates the activation of traffic analysis using the Morpheus module. This involves the creation of the SmartNIC_M and starting the analysis of raw network traffic. Later, the detection of anomalous raw traffic is symbolized by the red-coloured arrows. The SmartNIC_M, detecting an anomaly, raises a notification to the SmartNIC_OS, which, in turn, forwards it to the SBI module. The SBI module promptly notifies the CAD of the detected anomaly, initiating a chain reaction for mitigation. The CAD triggers the Attack Mitigator, which interfaces with the SBI module to block traffic. The SBI module configures new Access Control List (ACL) rules in the IP routers through NETCONF and the IETF ACL data model to fortify the network against the identified threat.

The proof-of-concept has been implemented in ADRENALINE Testbed [MNC+17]. To this end, NVIDIA SmartNIC BlueField-2 has been incorporated in a COTS server interconnected with whiteboxes Cell-Site Gateways (GSW) from EdgeCore using IP Infusion Network Operating System (NOS). Probe traffic has been forwarded from the CSW towards a photonic network of 4 Reconfigurable Optical Add-Drop Multiplexers (ROADM), controlled using an Optical Line System. TeraFlowSDN has been deployed on top, using the necessary extensions for the previously presented architecture. The generated pipeline for the SmartNIC has been deployed using Morpheus and configured to use the XGBoost model. The model is an example of a binary classifier to differentiate between anomalous crypto mining / malware, and normal workflows.

3.1.2.3 Time Sensitive Networking and Deterministic Networking SDN controller

Time Sensitive Networking (TSN) and Deterministic Networking (DetNet) provide bounded latency and zero packet loss in a reliable fashion in respectively L2 and L3 networks. Although their goals look similar, they differ in lots of aspects because they have different scopes: DetNet is an L3 solution that can run on top of a TSN network. Unlike TSN networks, which typically use Ethernet links and rely on high time synchronization accuracy for technologies like Time-Aware Shaping (TAS) and Cyclic Queueing and Forwarding (CQF), DetNet is not restricted to Ethernet links and employs alternative packet scheduling techniques in the absence

of high time synchronization accuracy such as strict priority queueing and tagged Cyclic Queueing and Forwarding (tCQF). Still, a unified network model that incorporates all these forwarding techniques, while also allowing to reserve resources on the end-to-end path, is very complex and difficult to maintain.

This implementation proposes an East-Westbound (EW) control architecture that allows a modular DetNet network using a divide-and-conquer approach. As shown in Figure 3-8, the main idea is that multiple centralized network controllers (CNCs) are employed, where each controller is only concerned about its specific network segment. This segment is a collection of devices that employ the same packet scheduling techniques, e.g., a TSN network segment with TAS-based switches or an L3 network segment with routers employing strict priority queueing. The controllers use the EW protocol to exchange various types of information.

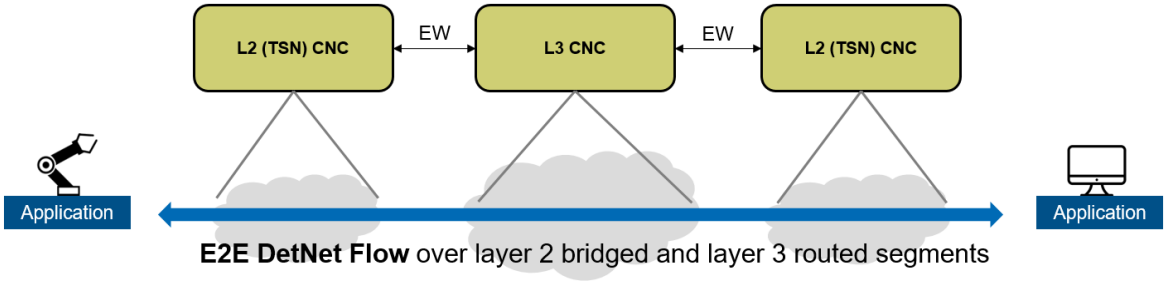


Figure 3-8: Multi-Segment DetNet Network Architecture.

The EW protocol has the following main features:

- It provides discovery of attached neighbouring network segments in a BGP-like fashion.
- It coordinates the routing and resource reservation problem of an end-to-end flow over the different segments. Importantly, it needs to translate a given traffic specification into a representation that is useful at the next segment. Network calculus can be used to derive arrival curves, which can be used as a traffic specification across multiple segments.
- It spreads global configuration along the different segments.

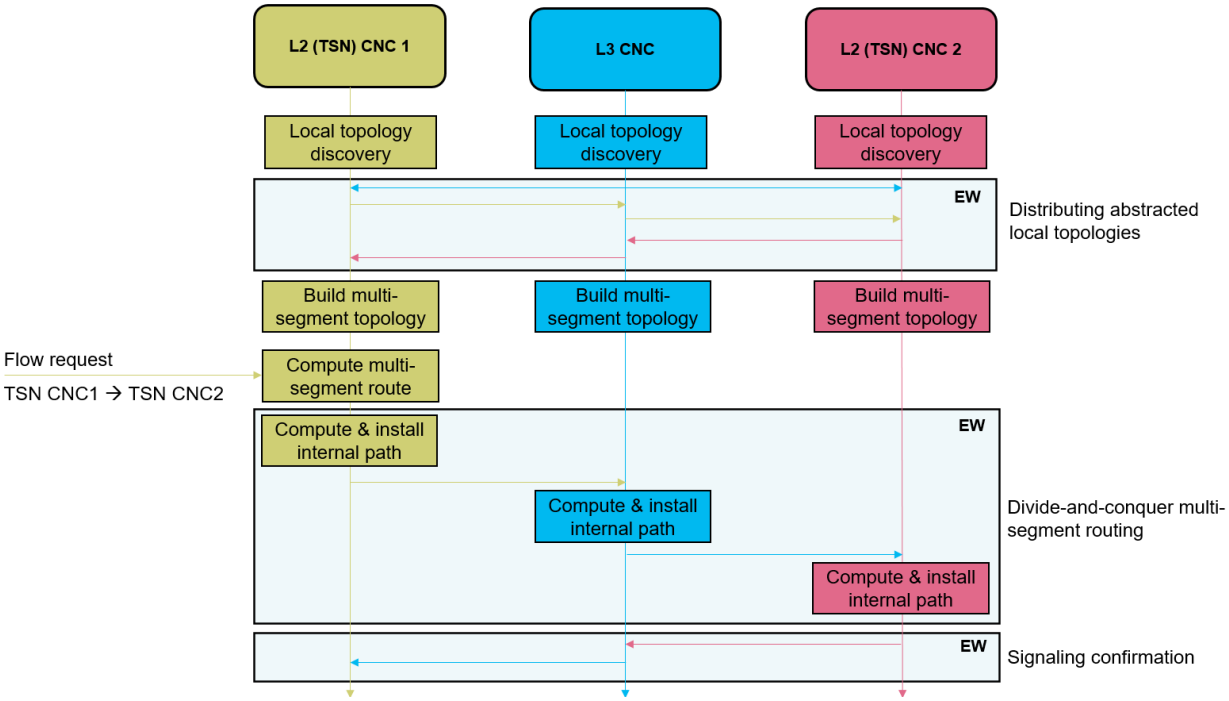


Figure 3-9: Local actions and EW interactions between CNCs.

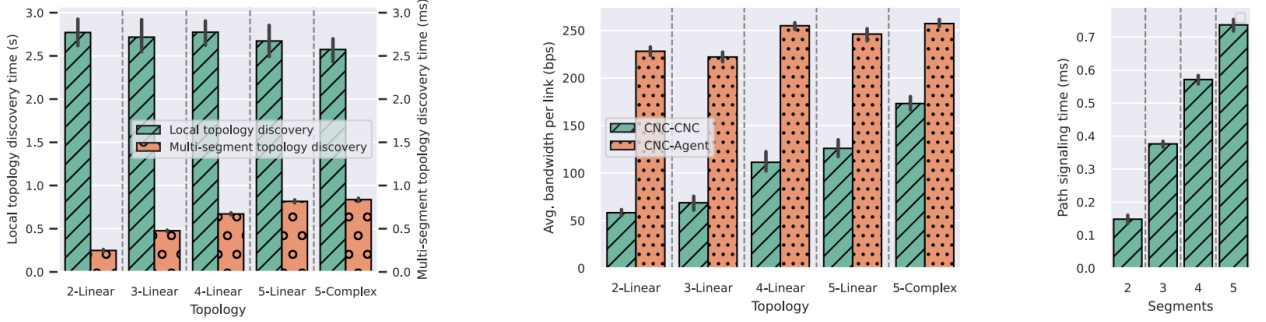
Given the network topology in Figure 3-8, Figure 3-9 describes the different local and EW interactions in order to setup an end-to-end path. It works as follows: first, each controller discovers the topology of its own network segment. Then, it transforms this detailed topology into an abstracted version. This abstract topology consists

of neighbouring segments, local end stations, and additional statistics such as network diameter. The EW protocol is then used to flood this information to the other controllers. By gathering all abstracted topologies, a multi-segment topology can be built at every controller. Then, as a flow request arrives at TSN CNC1, the multi-segment routing algorithm computes a path of segments for an end-to-end flow. After computing this path, the end-to-end delay budget division algorithm divides the end-to-end delay among the different segments. Then, the EW protocol coordinates the end-to-end path setup by computing and installing an internal every involved segment given the assigned local delay budget and translating the traffic specification across segments. Finally, a confirmation is signalled back to the source.

In this initial implementation, the multi-segment routing algorithm computes a multi-segment path that crosses the least number of segments to the destination. The end-to-end delay is divided among the segments proportionally based on the network diameter.

Early Validation Results

The proposed implementation was demonstrated on Linux-based routers/switchers in an emulated network setup. Five different networks of increasing size and complexity were used. As shown in Figure 3-10, operations that are contained to a local segment only, retain constant complexity in all networks. Interactions that require coordination over multiple segments, such as multi-segment topology discovery and path signalling upon a flow request, have increasing complexity, which is also reflected in the increase in control overhead. Additional details on the architecture, the used algorithms, the different interactions, and the experimental setup of this implementation can be found in [MCP+24]. Moreover, in the context of PoC#B.1, this implementation can help to demonstrate the seamless deployment of a latency-sensitive application/flow over a complex/heterogeneous TSN/DetNet network.



(a) Topology discovery time (b) Control overhead (c) Path signaling time

Figure 3-10: TSN/DetNet SDN controller early validation results.

3.1.2.4 Automated transport network re-configuration

This implementation is expected to be demonstrated in ADRENALINE Testbed at CTTC premises [MNC+17]. To this end, preliminary implementation details are provided, but full feature will be available for next D6.4. This new workflow will be part of TeraFlowSDN R4. The new ETSI ZSM-aligned Monitoring-Analytics-Automation loop architecture is being described in Enabler 8 and it is also under development in ETSI TeraFlowSDN, as depicted in Figure 3-11. This architecture may be used as an example to implement closed-loops (see enabler 8 in section 3.8) for the automation of transport networks.

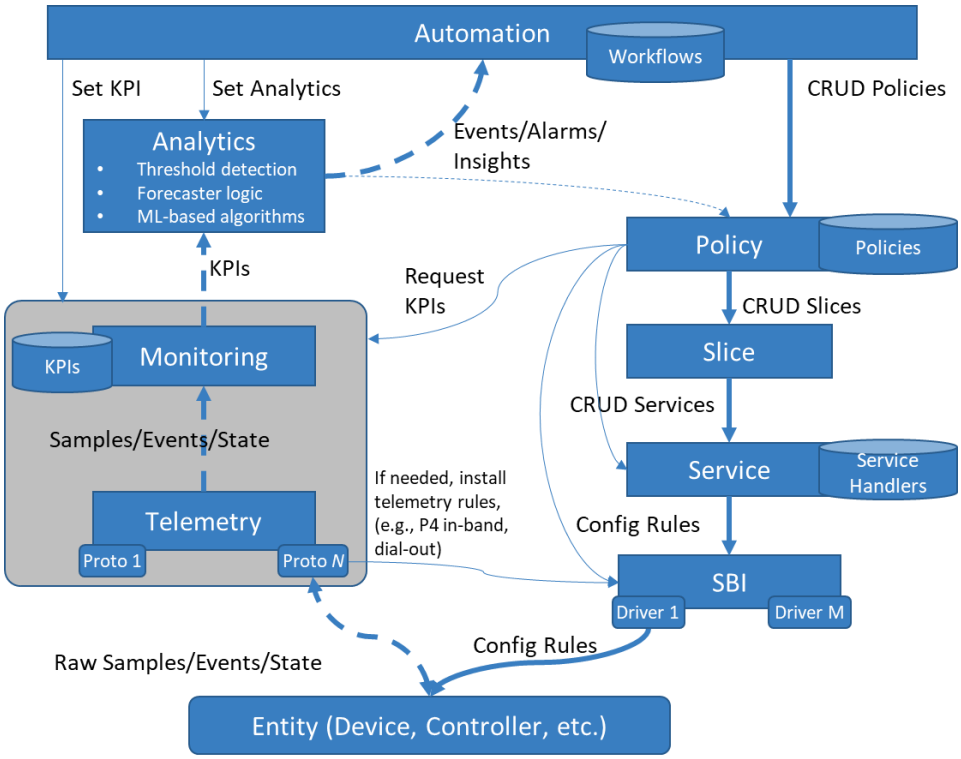


Figure 3-11: TFS ZSM-aligned Monitoring-Analytics-Automation Loop.

The TeraFlowSDN components that are involved in this implementation are:

1. The **Telemetry** data ingestion component is based on plugins (to provide extensibility). It supports polling-based and streaming-based raw telemetry data ingestion, e.g., raw samples, events and state data, from the underlying entities, i.e., network devices and intermediate controllers. The collected data is forwarded to the Monitoring component for further processing.
2. The **Monitoring** component is responsible for managing the KPI descriptors defining the type of data to be collected, the observation point used for interrogating the devices, e.g., the path in the data model used to retrieve or subscribe to receive the data, and the TeraFlowSDN resources associated with the KPI descriptor, e.g., the device, port, link, service, etc. Monitoring also checks and adjusts the timestamps of the collected data, performs the mapping with existing KPI Descriptors, and produces usable KPI Values that are later stored in a TimeSeries DataBase.
3. The **Analytics** component implements a number of pluggable algorithms including simple threshold-based detectors, forecasters, and AI/ML-based algorithms. It subscribes to Monitoring to receive new KPI Values produced and analyses them according to the configured analyses. As a result, it produces Events, Alarms, and Insight that can be consumed by other components.
4. The **Policy** component implements a set of pluggable Event-Condition-Action—based policies. It consumes outcomes from Analytics and, whenever they fulfil a configured event (e.g., type of notification) and condition (e.g., for a specific resource) it triggers the associated policy (e.g., a set of actions to be sequentially executed). Among others, the actions supported include performing Create/Read/Update/Delete operations and/or configuring explicit configuration rules over TeraFlowSDN-managed transport network slices, services, or devices.
5. The **Slice**, **Service**, and **SBI** components are the existing TeraFlowSDN components used to manage transport network slices, connectivity services supporting the transport network slices, and devices where the configuration rules are installed to implement the connectivity services.
6. Finally, on top of all the components, the **Automation** component is responsible for managing the overall set of components involved in the Monitoring-Analytics-Automation loop. It takes as input the Service Level Agreement constraints defined in newly-configured connectivity services and transport network slices, and extrapolates the required KPIs to be monitored, the analytics algorithms required to process them, and the policies to be triggered when specific events and conditions are met. Given the wide range of scenarios that might arise, the Automation component is based on plugins, shaped

as workflows, to provide the appropriate levels of flexibility. Interestingly, the Automation component can also consume events, alarms, and insight from the Analytics component to self-reconfigure its internal workflows.

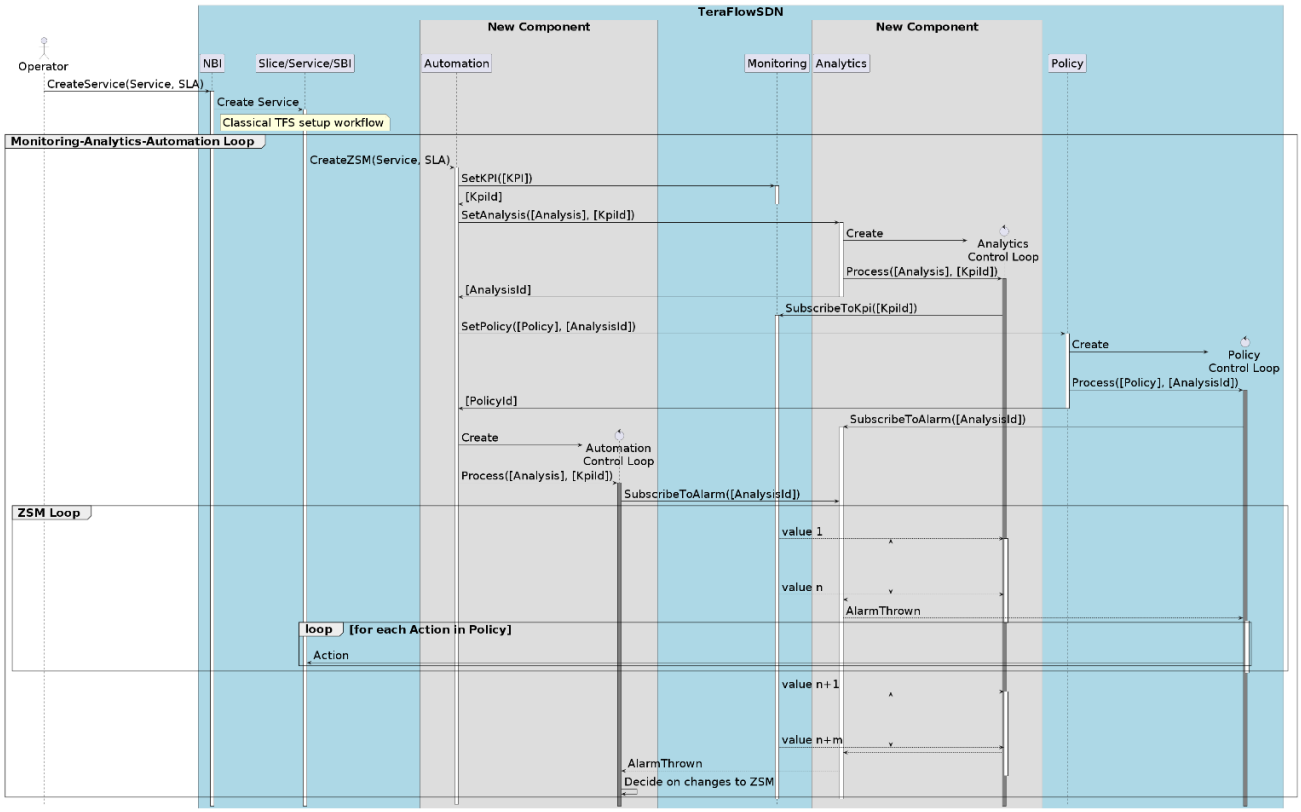


Figure 3-12: Sequence diagram: operation of the Monitoring-Analytics-Automation Control Loop.

The sequence diagram illustrating the operation of the Monitoring-Analytics-Automation Control Loop is depicted in Figure 3-12. The workflow assumes the transport network slices and services are created in the TeraFlowSDN controller and, if the slice/service carries some SLA to be enforced through the utilization of the Monitoring-Analytics-Automation Control Loop, it issues a request to the Automation component. The Automation component first infers the required KPIs, Alarms, and Policies to be used through the pluggable workflows it features. Next, it uses the outcome from the selected workflow to configure:

1. the Monitoring component to perform the telemetry data collection,
2. the Analytics component to perform the analysis over the data,
3. the Policy component to execute the appropriate sequence of actions encoded in the policies, and
4. it self-subscribes to alarms from Analytics to self-reconfigure as needed.

Note that each of Analytics, Policy, and Automation creates an internal self-running Control Loop in order to provide the appropriate levels of scalability and that Control Loop is the one responsible for subscribing to receive data, execute the analysis algorithms, policies, and workflows, respectively. In the bottom part of the figure, two cases are illustrated:

a) the execution of a ZSM loop triggering a reconfiguration of a slice/service/device, and
b) the self-reconfiguration of Automation component, which might trigger changes in the configuration of Monitoring, Analytics, and Policy, based on the alarms, events, and insight produced by the Analytics component.

3.1.2.5 MEC Bandwidth Management service integration with SDN controller

This section explores the synergy between MEC BandWidth Management (BWM) service and TeraFlowSDN (TFS) [TFS24] in dedicating resources for optimal network resource allocation in the gaming domain. This implementation is part of presented ETSI MEC PoC 14: Network resource allocation [MECP014] for Application specific requests using MEC BandWidth Management service and TeraFlowSDN. The source

code has been introduced in Release 3. BWM empowers applications to earmark specific bandwidth quotas (among other quality of service constraints, such as latency) for gaming applications, while TFS orchestrates the management and control of traffic flows. The resulting prioritization of gaming traffic over other data holds the potential to significantly enhance the overall gaming experience for users.

The benefits that MEC BWM and TFS offer in the context of application networking improved application performance, reduced congestion, and increased scalability stand out as key advantages. The prioritization of application traffic, facilitated by BWM and TFS, contributes to latency reduction, thereby elevating the overall quality of the experience. The allocation of designated bandwidth to applications mitigates network congestion, fostering an improved application environment for all users. Crucially, MEC BWM and TFS showcase scalability, effortlessly accommodating the evolving landscape of applications and the surge in user traffic. This scalability is particularly pertinent as applications might continue to gain popularity, necessitating robust solutions that can seamlessly adapt to increasing demands.

In the ever-evolving landscape of network resource management for applications, the integration of MEC BWM service and TFS stands as a groundbreaking innovation. This section delves into the unique aspects of this approach, detailing how it enhances the experience for users.

- Application: MEC BWM and TFS allow application-triggered requests to the network. This allows ensuring a targeted and responsive allocation of network resources for the most bandwidth-demanding applications. The application has been modified so it is able to request the necessary network resources.
- End-to-End Dynamic and Adaptive Bandwidth Allocation: The core innovation lies in the dynamic and adaptive nature of end-to-end (E2E) bandwidth allocation. The integration with TFS enables real-time responsiveness to the changing demands for the packet-optical network. This allocation mechanism ensures that resources are optimally distributed in a multi-layer network.
- Application-network interface in TeraFlowSDN: A pivotal innovation lies in the novel API facilitated by TFS to directly allow applications to request bandwidth allocation resources. This dynamic allocation ensures that applications receive dedicated and exclusive network services. TFS can monitor in real-time a departure from static allocation methods, offering a proactive approach to safeguard against potential congestion and maintain an uninterrupted SLA request.
- User-Centric Focus: Ultimately, the innovation in MEC BWM and TFS is driven by a user-centric philosophy. By prioritizing applications and tailoring resource allocation to their specific needs, this approach elevates the gaming experience to new heights. Reduced latency, improved performance, and minimized congestion collectively contribute to a network environment that aligns with the expectations and demands of modern applications.

Figure 3-13 shows the implemented demonstration using a gaming server and client as example applications. It can be observed that ETSI TFS acts as an E2E SDN controller/orchestrator of a multi-layer multi-domain packet optical network. To this end, it controls hierarchically two domain-specific SDN controllers (IP and optical SDN controllers). Also, edge and cloud resources are shown.

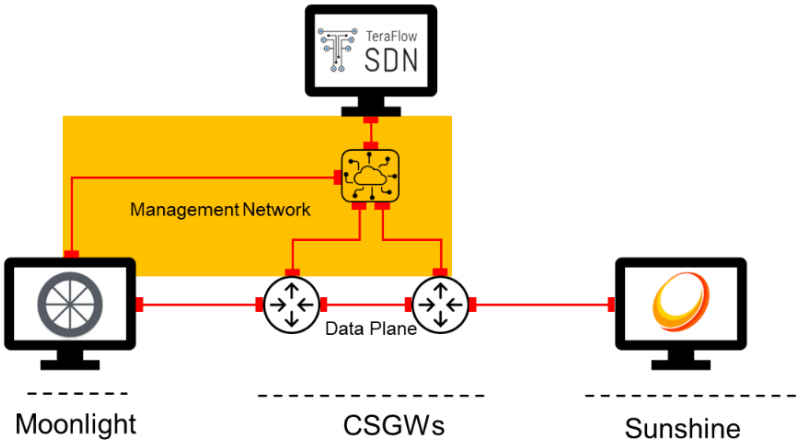


Figure 3-13: TeraFlowSDN and MEC integration proof-of-concept.

Moreover, the figure depicts two applications: Moonlight and Sunshine. Moonlight is a game streaming application client. It is run on the network edge or extreme edge. Moonlight allows to play PC games on almost any device, through game streaming, as it is an open-source implementation of NVIDIA's GameStream protocol [MOO24]. On the other hand, Sunshine is a game streaming application server. It is a self-hosted game stream host for Moonlight, offering low latency, cloud gaming server capabilities with support for AMD, Intel, and Nvidia GPUs for hardware encoding. Sunshine is run in the cloud resources, typically located in a Data Centre (DC). Upon client application request through MEC 015 BWM allocation request [MEC015], a certain degree of bandwidth is granted from end-to-end (E2E) perspective by ETSI TFS. To this end, TFS interacts with underlying SDN controllers and Network Elements, such as whiteboxes (IP routers) and Reconfigurable Optical Add-Drop Multiplexers (ROADMs) in order to deploy constrained connectivity service.

Figure 3-14 presents the implemented sequence diagram for a complex network architecture with E2E communication involving various components such as E2E SDN controllers, IP and optical SDN controllers, whiteboxes, ROADMs, and cloud resources.

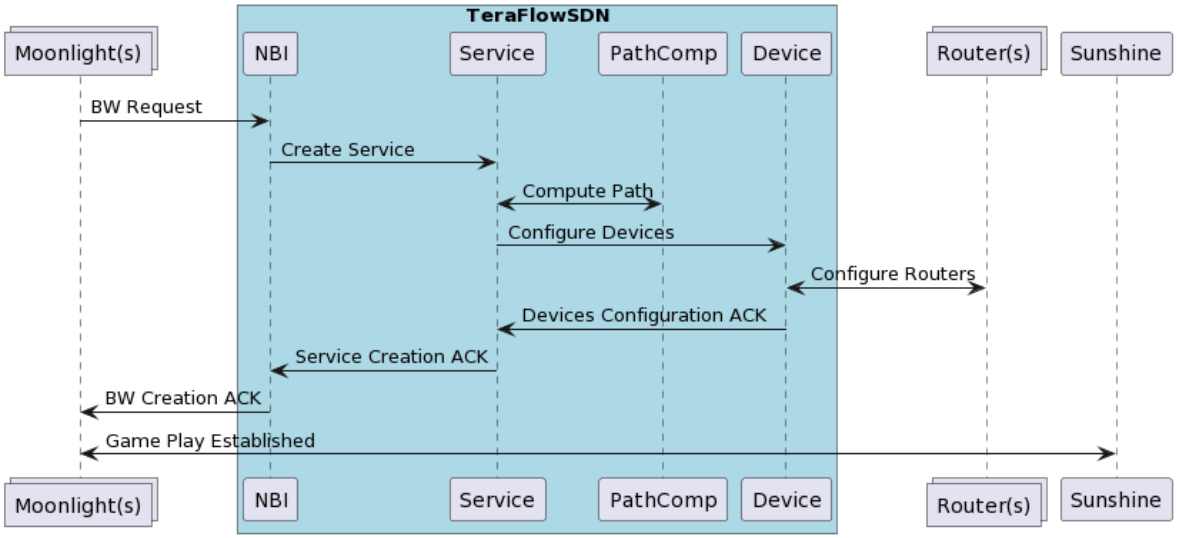


Figure 3-14 Sequence diagram of integration of BWM services and ETSI TeraFlowSDN.

3.1.2.6 Integration of TM Forum APIs with ETSI TFS NBI

The evaluation and detailed exposition on the intersection and integration of TM Forum, IETF and ETSI, focused on enhancing network management through API standards is analysed in Appendix 8.3, while this section focuses on the concrete example of TM Forum APIs integration in ETSI TFS NBI. The TM Forum experience in proving guidelines, best practices and Open APIs from service providers and their suppliers, as well the vision that has in autonomous networks and aligned with the focus on increasing automation and flexibility in network management from ETSI's perspective, is an opportunity to integrate and test the alignment between ETSI's framework concept of intents and TM Forum's approach where network operations are driven by high-level customer requirements rather than detailed technical configurations. With a common technology-agnostic approach from both TM Forum and ETSI we can purse a generic solution between all the parties, allowing independence of the underlying network technology and allowing the specification of customers need without any concern to the specifics of the technology.

To achieve a holistic solution, it is needed to integrate the TM Forum APIs [TMF-OA] into the existing NBI interface of ETSI's TeraFlow SDN Controller, offering to network service providers the best of both worlds: the robust, telecom-focused business processes and service models from TM Forum, coupled with the technical and architectural strengths of ETSI's network management frameworks. This can lead to more dynamic, efficient, and customer-centric network operations.

This integration involves several aspects and it is implemented under OPTARE's testbed (two cloud instances, Linux based, with 4 CPU each, one having 16GB RAM and the other 7,5 GB RAM and 60GB and 250GB for storage). The integration involves the following aspects:

- **Service Abstraction:** TM Forum APIs, such as TMF640 (Service Activation and Configuration) [TMF-640] and TMF664 (Resource Function Activation and Configuration) [TMF-664], are used to abstract network functions and services in a way that they can be consumed by business applications.
- **Standardization:** TM Forum APIs are standardized and widely accepted in the telecom industry, which means they can help ensure that the integration into the ETSI TFS NBI will be compliant with industry practices, allowing for easier interoperability and communication among systems.
- **Functionality Mapping:** Functions that the TM Forum APIs perform, such as activating a service or allocating a resource, are being mapped to the corresponding functionalities within the ETSI TFS NBI.
- **API Extension or Adaptation:** There is the need to extend the TM Forum APIs or adapt them to fit into the specific requirements and capabilities of the ETSI TFS NBI.
- **Protocol and Data Model Alignment:** TM Forum APIs need to work with the protocols and data models supported by the ETSI TFS NBI. The TM Forum APIs translate their service requests into the YANG models understood by the network controller, so a data transformation is being implemented that can translate the protocols and data models into YANG models and the ones defined by ETSI TFS NBI.
- **Workflow and Orchestration:** The service orchestration and workflow capabilities provided by TM Forum APIs complement the orchestration logic of the ETSI TFS NBI, allowing a seamless lifecycle management of services and resources and fully integrated within ETSI architecture.

Figure 3-15 represents a sequence diagram that outlines the process of integrating TM Forum APIs into the ETSI TFS NBI to facilitate dynamic network management. It features several key elements: TMF640 and TMF664, representing Service Activation & Configuration API and Resource Function Activation & Configuration API, respectively. These interfaces interact through the ETSI TFS NBI. The diagram illustrates the flow of service and resource activation requests from the TM Forum side, which are then translated and forwarded by the ETSI TFS NBI to manage network resources. It also shows feedback loops where the network resources provide status updates and metrics back through the ETSI TFS system, which aggregates and translates these updates.

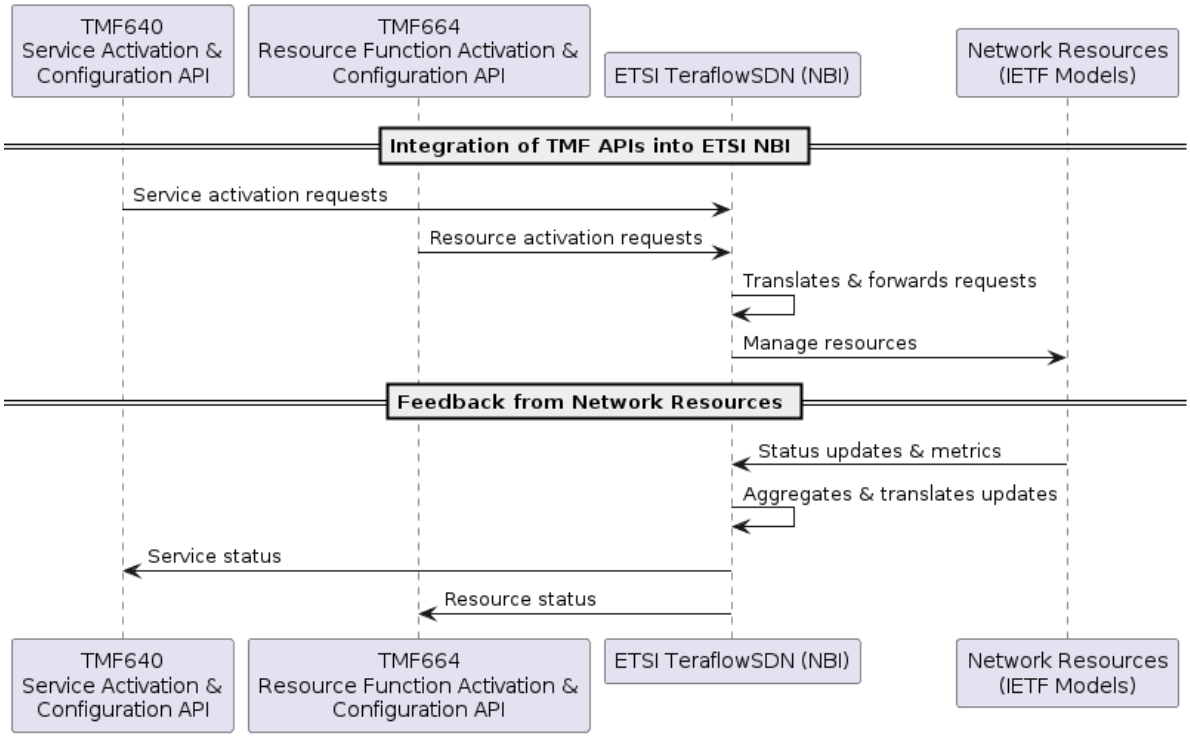


Figure 3-15: TMF API integration within ETSI TFS NBI interface.

3.1.3 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this Enabler 1 are the following:

- **Scalability**. As it considers a cloud-native architecture, the proposed Enabler 1 is intrinsically highly scalable, which is considered quite relevant regarding the provisioning of intense volume of connectivity services. The distributed orchestration network elements would make possible to handle an increased amount of network services and workloads of different shapes and sizes, and without requiring complex centralized systems that could become bottlenecks or single points of failure, and that would need to be upgraded to be able to manage more services and resources.
- **Latency**: The proposed approach relies on configuring network elements on the whole network continuum. The optimal allocation of services, through network programmability in the cloud continuum can lead to a large reduction in latency.
- **Flexibility**: Scalable and cloud-native network control and management introduces support multiple network hierarchies and technologies.
- **Services Creation Time** is a KPI that would be of course affected by the Enabler, which shall focus on minimizing it.
- **Reliability**. The enabler has been designed to specifically target the necessary redundance by design, so multiple instances can be easily managed.
- **Programmability**. The proposed system is highly programmable by itself, relying on the cloud-native principles as a whole (e.g., all the network elements, such as routers, communicate using exposed interfaces, which enable programmability, and could be provided relying on highly automated DevOps practices).

3.2 Enabler 2: Monitoring and telemetry framework

Programmable network monitoring and telemetry are revolutionizing the way network data is collected and analysed, by harnessing the power of Software-Defined Networking (SDN) and a suite of automation technologies. While monitoring ensures ongoing oversight of network performance metrics and status indicators, telemetry facilitates the automated collection and transmission of real-time data from diverse network sources. Network Data Analytics Function (NWDAF) is a functionality already available in 5G core. The purpose of this enabler is to extend the current SoA and consider monitoring a kind of pervasive functionality, in the sense that for the upcoming 6G networks the collection of monitoring data from multiple and heterogeneous sources is considered necessary. Enabler 2 approach also goes beyond data collection, as it also addresses the immediate reception and the forwarding of network related events, as well as its fusion and processing. Also, a specific scalable architecture is proposed to provide a cloud-scale solution.

This methodical approach allows for the real-time gathering of intricate data from both virtual and physical network components as well as applications. The essence of this framework is its multi-layered data collection capability, which spans across various services and applications, ensuring a comprehensive view of the network's operational dynamics. By implementing a multi-vendor strategy, it achieves higher scalability and flexibility. This real-time data provides a robust foundation for decision-making, allowing network administrators and automated frameworks to dissect network traffic flows and performance metrics meticulously. Through this analysis, insights into network utilization patterns emerge, pinpointing areas of resource underutilization. With this knowledge, network configurations can be refined and optimized, fostering enhanced performance and cost-efficiency. Moreover, this sophisticated monitoring extends to the measurement of energy consumption across network elements, including computational resources. The collation of these data is pivotal for the creation of algorithms aimed at deploying networks that balance operational demands with energy efficiency.

The breadth of technologies and protocols that the proposed enabler will leverage is crucial for its efficacy. And they play a vital role in enabling real-time data harvesting and analysis. These protocols not only facilitate the automation of network management tasks but also aid in the orchestration of network functions and policies. Moreover, the system is designed to incorporate robust authentication and privacy measures, addressing the ever-important concerns of security, privacy, and systemic resilience. To manage cloud-scale operations and circumvent any potential monitoring bottlenecks, the system architecture contemplates a

sequential processing approach for metrics and alerts. This includes a series of independent yet interconnected steps—acquisition, normalization, visualization, evaluation, and publication of metrics and alerts—ensuring that each aspect of network monitoring is handled with precision and efficiency, thus maintaining the smooth operation and scalability of the system.

D6.2 [HEX223-D62] presented the SoTA and expected beyond State of The Art of this enabler. To this end, work on the multiple components was proposed, as well as possible external interfaces. The following subsections present the internal architecture, as well as preliminary implementation details, and early validations results.

3.2.1 Enabler design

Monitoring and telemetry are crucial components that enable system reliability, performance, and security. The framework depicted in Figure 3-16 provides a clear pathway from data collection to actionable insights, utilizing a set of tools and processes. The first stage involves the Collector, which is responsible for gathering raw data. Tools like gnmic [GNMIC24], OpenTelemetry [OTL24], and Prometheus [PRO24] are often employed at this stage. Once data is collected, it is passed to the Processors. The processors, which may use OpenTelemetry and tools like Apache Airflow, are responsible for parsing, aggregating, and transforming the data into a coherent format that can be analysed. After processing, the data is sent to the Exporters, which may also utilize OpenTelemetry for exporting the processed data to various destinations. This step is essential for delivering the information to the appropriate tools used for analysis and visualization. Finally, the data arrives at tools like Prometheus for storage and Grafana for visualization. Prometheus can store the processed time-series data, making it queryable, while Grafana specializes in turning this data into actionable insights through its powerful dashboards. These insights enable operations teams to detect and respond to issues in real-time, making informed decisions based on comprehensive data. This architecture has the challenge to be adapted and extended to support all specific requirements for upcoming 6G networks.

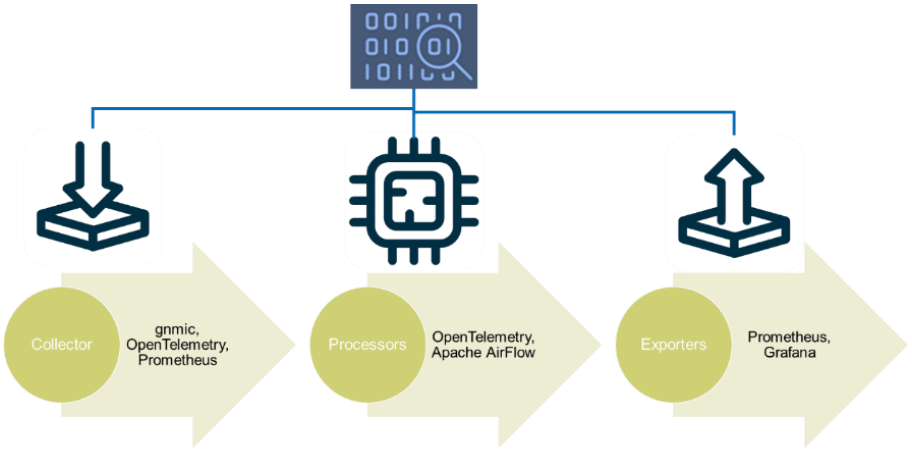


Figure 3-16: High-level architecture of monitoring and telemetry framework.

3.2.1.1 System components

Figure 3-17 depicts a schematic of a data processing system, arranged horizontally to illustrate the flow and interaction of data between various components. It consists of a cloud-native design with microservices connected through a common bus. At the forefront of the system is the "Subscription Manager," which likely oversees the configurations for data collection across the system. Next in line are "Exporter #1" and "Exporter #2," denoting two distinct pathways through which data is extracted and possibly disseminated to different destinations or for various uses. An "In-band telemetry exporter" is also featured, implying a mechanism tailored to handle telemetry data within the operational bandwidth, ensuring efficient monitoring.

As we move further right, there's an "Event Processing" unit, indicative of a subsystem dedicated to handling discrete events that occur within the system—these could be irregular, significant, or require special processing separate from the main data flow. Adjacent to this is a "Time Series #1" database, which suggests storage and retrieval functionality for data points collected sequentially over time, a common requirement for monitoring trends and patterns.

The "Non-SQL DB #1" component indicates the use of a non-relational database, which is designed to store and manage large volumes of unstructured or semi-structured data, signifying flexibility and scalability in data handling. Finally, the "Visualization #1" module posits an endpoint for the system's data, where it is likely transformed into graphical representations to aid users in interpreting and analysing the data.

Supporting these main components are several "Collector" blocks, labelled "Collector #1" through "Collector #N." These are depicted beneath the main data pathway, suggesting a hierarchical relationship where these collectors feed into the upper-level components. The term "Collector #N" alludes to a scalable array of data collection points, with "N" representing an indefinite number that can be expanded as needed. Among these, an "Active Collector (in-band telemetry)" is specified, highlighting its active role in gathering in-band telemetry data, which is crucial for real-time monitoring and analysis. Below the description of each of the components in the Figure 3-17.

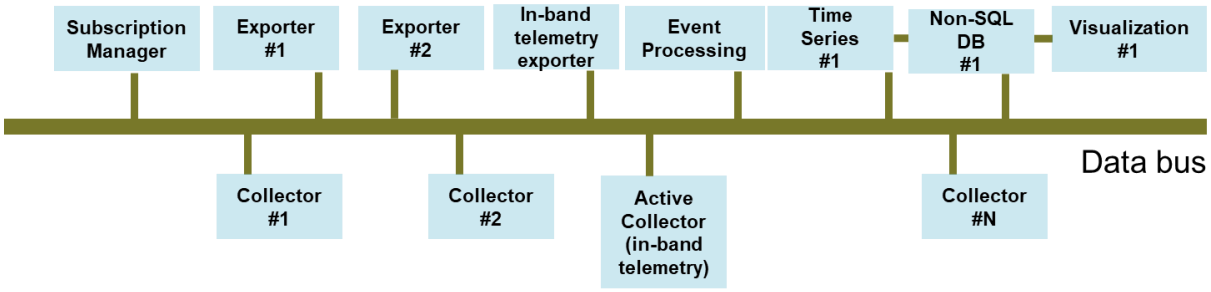


Figure 3-17: Monitoring and telemetry framework internal architecture.

- Distributed event streaming platform (depicted as Data Bus).
- Communication Protocols: Choose protocols for data transfer between devices and the central system. Common ones would include MQTT, HTTP, or CoAP for IoT.
- Real-time Processing: Implements real-time processing for immediate analysis and response. Tools like Apache Kafka [KAF24] or Apache Flink [FLI24] can help with streaming data.
- Collector: Collects data from various sources. This could include IoT devices, or other data points. Sensors and probes are also included in this category.
- Exporter: Set up alerts for abnormal conditions or thresholds. This ensures the delivery of notifications when something requires attention.
- Subscription Manager: is the responsible for handling the specific topics in the data bus.
- Time series: Plan for long-term storage and analysis of historical data. This is crucial for trend analysis and making informed decisions. Services like Prometheus [PRO24] or Nagios [NAG24] could be used.
- Database: robust database system to store the collected data efficiently. Options include SQL databases or NoSQL databases, depending on the data structure.
- Visualization: Tools like Grafana [GRA24] can help in visualizing data and creating dashboards.
- Security Measures: Implement robust security practices to protect the platform from unauthorized access. This includes encryption, authentication, and access controls.
- Scalability: Design the platform to scale easily as the number of monitored devices or data points increases. The solution can be deployed in the entire network continuum.

3.2.1.2 Workflows

Figure 3-18 shows the sequence diagram of the flow of events through a system that collects, processes, and visualizes data. In the diagram, there are several entities or components involved: Collector1, Data Bus, Exporter1, Timeseries Component, External App, Database, and Visualization.

A data-driven process begins with Collector1, which is responsible for generating an event. This event is then passed to the Data Bus. The role of the Data Bus is to distribute the event to various components within the system. In this case, it distributes the event to both Exporter1 and Timeseries Component.

Exporter1 seems to have a role in further distributing the event, perhaps to different systems or for different uses. Meanwhile, the Timeseries Component relates with Database and Visualization.

Next, the flow of the event leads to the Database component, which stores the event data. This suggests that the system has a persistence mechanism to maintain a record of events over time, which is typical in systems where event data needs to be analysed retrospectively.

Finally, the Visualization component receives the event data. This suggests that there is a user-facing aspect of the system where the data collected and processed by the system is displayed. This could be in the form of dashboards, charts, or reports that allow users to understand and interpret the data.

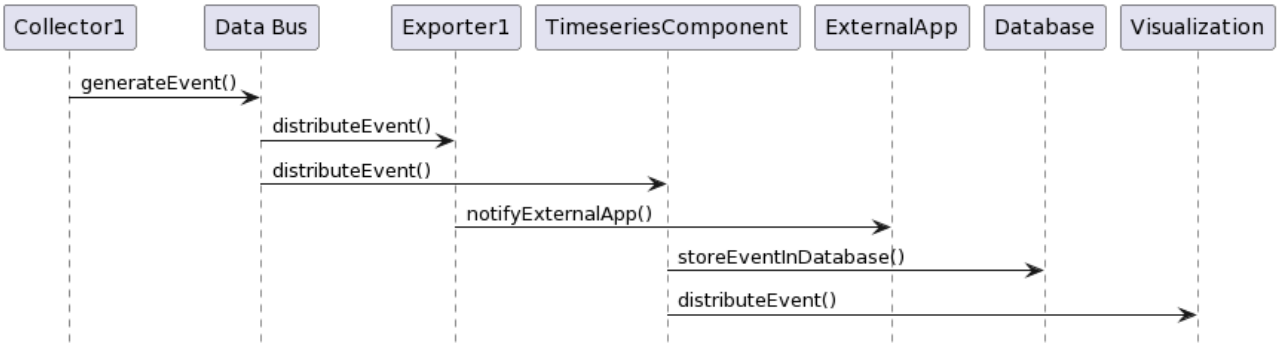


Figure 3-18: Monitoring and Telemetry Framework sequence diagram.

The interfaces of the Monitoring and Telemetry Framework are described in Appendix 8.2.2.

3.2.2 Preliminary implementation and early validation results

This section presents several implementations of the proposed enabler that will be released either as open-source, or be present in several proof-of-concepts (PoC).

3.2.2.1 TeraFlowSDN event-driven monitoring

TeraFlowSDN has started to modify its implementation following the event and data-driven architectural principles previously detailed. To this end, the new event-monitoring system will be included in release 4 of ETSI TeraFlowSDN and it will be part of the System PoC B.1. Figure 3-19 outlines a sequence of steps in a data processing workflow where large volumes of data are collected, processed, and analysed to gain insights or trigger automated actions. Each step represents a critical stage in the data lifecycle, from initial extraction to final visualization.

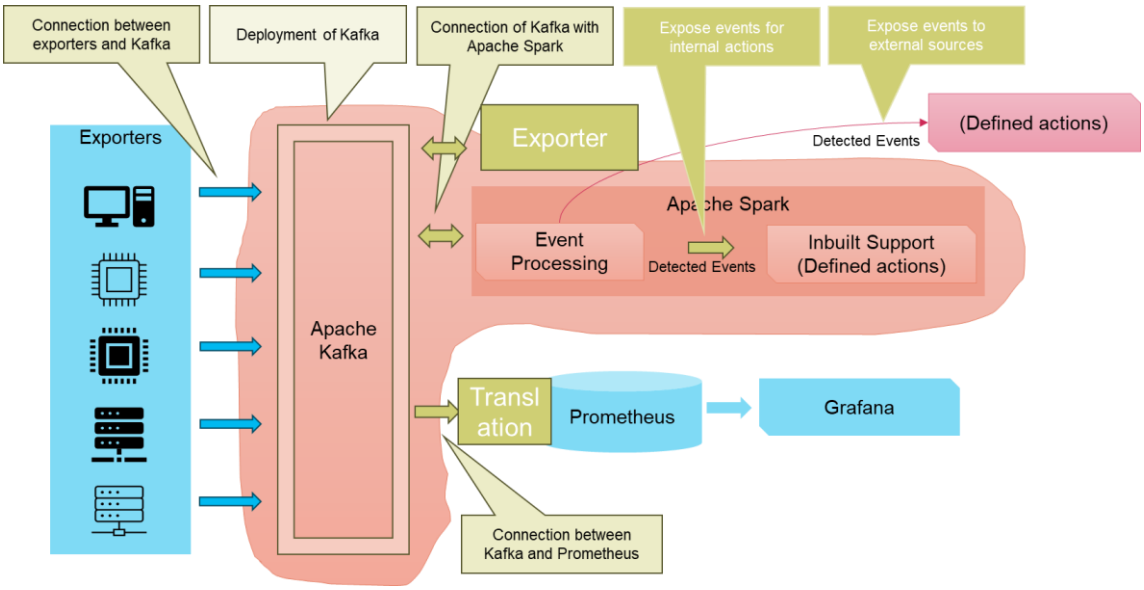


Figure 3-19: TeraFlowSDN event-driven monitoring.

Data extraction serves as the crucial first step in the monitoring and telemetry framework, focusing on the retrieval of raw data from the system under observation. This stage utilizes Application Programming Interfaces (APIs), agents, or other collection methods designed to interact with the managed device. These

tools effectively capture the necessary metrics, logs, events, or traces that reflect the operational state of the device. The precision and comprehensiveness of data extraction are fundamental to ensuring the quality and reliability of the entire monitoring process. This foundational step sets the stage for all subsequent data handling and analysis, making it imperative to use reliable and efficient methods for data gathering.

Following extraction, the raw data is streamed into an event streaming platform, such as Apache Kafka [KAF24]. These platforms are specially built to manage high-throughput data pipelines, facilitating a seamless and continuous data flow. Their robustness allows for the accommodation of vast amounts of data being transmitted in real-time, ensuring that the data pipeline is both resilient and scalable. By providing a distributed system for data queues, these platforms enable the rest of the framework to process the continuous stream of information without bottlenecks, making them an integral part of the telemetry and monitoring ecosystem.

As the data traverses through the event streaming platform, it reaches a stream processing platform where it is analysed in real-time. Tools such as Apache Flink [FLI24] or Spark [SPA24] Streaming are employed at this juncture to perform immediate processing tasks. These tasks include filtering out irrelevant data, aggregating relevant metrics, and detecting significant events or patterns within the data stream. This real-time processing is critical as it enables the system to promptly identify and react to operational anomalies, potential failures, or optimization opportunities, thereby enhancing the system's responsiveness and reliability.

Once processed, the data is funnelled into a data storage system for long-term retention and deeper analysis. Data storage solutions like Prometheus are adept at handling large volumes of processed time-series data. The storage system must be capable not only of accommodating the data scale but also of facilitating fast retrieval for analysis and reporting purposes. This step is pivotal for historical data analysis, trend identification, and forensic investigation, underpinning the future decision-making processes and strategic improvements.

With the processed data at hand, the TFS event-driven monitoring framework now defines actions based on the derived insights to be considered by TeraFlowSDN controller. These actions can either be automated responses within the processing platform or can trigger external workflows or processes. The definition of actions is based on pre-set thresholds, anomalies, or patterns identified in the data. This automated response mechanism enhances the system's efficiency by enabling quick resolution of identified issues or by flagging them for human intervention.

Data visualization is the presentation of the data in an interpretable format. Tools such as Grafana [GRA24] are utilized to create intuitive dashboards that showcase the data patterns, trends, and alerts.

The integration of specialized tools and platforms for each stage of the data pipeline ensures that the system can handle data streaming and provide actionable insights efficiently. The synergy between these components forms a resilient and intelligent monitoring and telemetry ecosystem, vital for the proactive management of software systems.

As mentioned, the *TFS event-driven monitoring* module will obtain large amounts of data from external devices that can be located in multiple environments (including cloud and edge). Elements in the Hexa-X-II blueprint could use this data to make the proper decisions to optimise the conditions of the infrastructure, taking as input the values that these devices report. These decisions should also be used to react against anomalous behaviours, for instance alerts from anomalous behaviours that might occur when a device sends values outside a certain threshold.

One example of the data that would be monitored through this module is energy consumption. The *TFS Event-driven monitoring* module must be able to monitor the energy of each external device in order to optimise network deployments or react against sudden energy cutoffs and consumption spikes. Naturally, all the monitored data must be available to the *TFS event-driven monitoring* module as soon as possible in order to dynamically implement the necessary policies for the best functioning of the monitoring platform.

This first validation explores the utilisation of a Kafka bus for data consumption and distribution for a preliminary implementation of the *TFS event-driven monitoring* module. Apache Kafka [KAF24] is an open-source distributed event streaming platform designed for high-throughput, fault-tolerant, and scalable data streaming. Figure 3-20 depict the implementation of the bus to monitor a data topic from a set of simulated external devices. The first step of the validation involved the deployment of a Kafka server with the creation of a data topic, where different devices (i.e., the data publishers) could push their data (data ingestion).

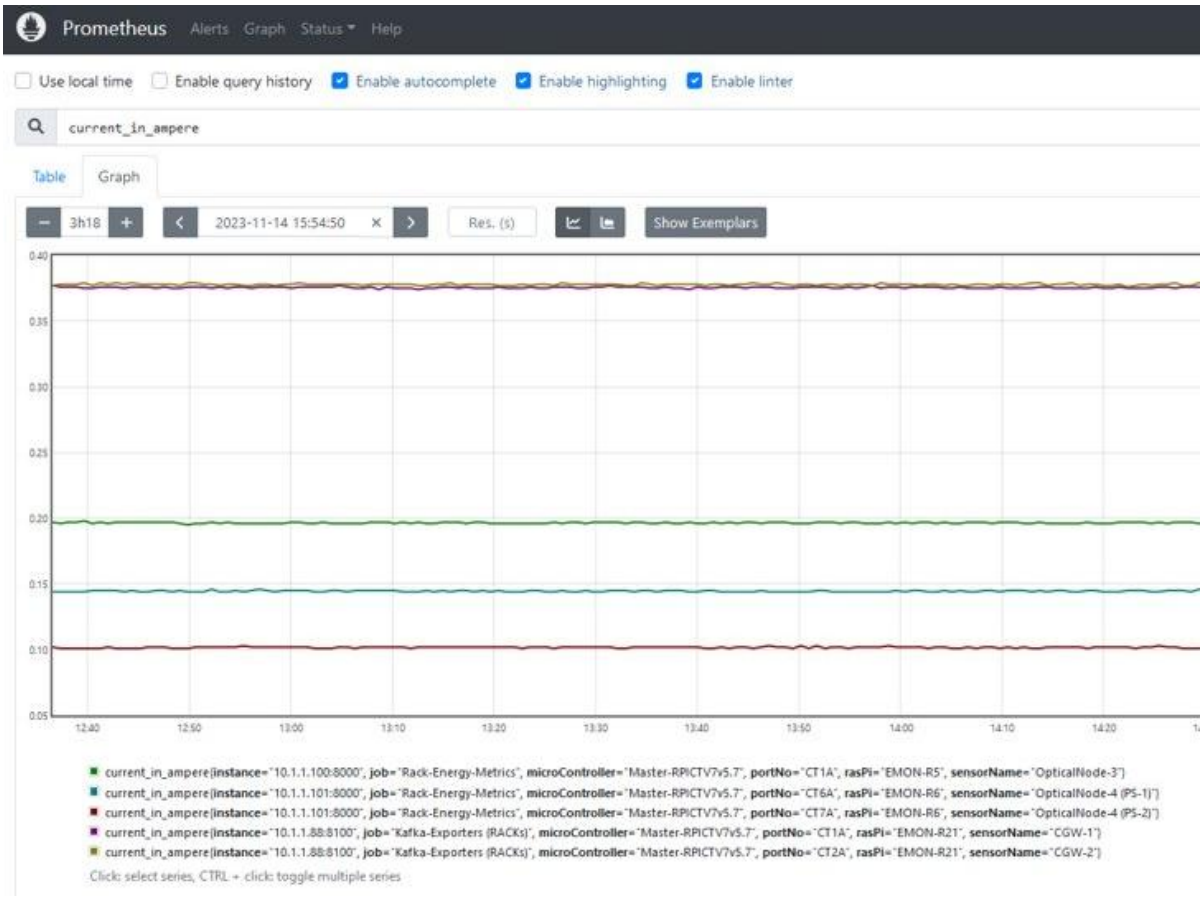


Figure 3-20: Prometheus visualisation of data topic.

The generated data can be read later by data consumers, when a member registers its intention to receive updates from a specific topic, it will continuously retrieve the metrics form the topic data stream. The TFS event-driven monitoring module could also use visualization platforms such as Grafana for graphically represent the data in real-time (as seen in Figure 3-20), as well as assisting the module into triggering alerts when its values surpass certain pre-defined thresholds.

This first validation demonstrates that both Kafka and Prometheus can be a suitable implementation for the event-driven architecture proposed for Enabler 2. Although this first validation focused on a single Kafka server, multiple instances of a server can be deployed in the infrastructure, enabling the data exchange within a topic between multiple servers to provide the solution with resilience and fault-tolerance, as well as high-availability (to support devices located in various sites).

The TFS event-driven monitoring module will also implement the stream processor platform functionality to process the data coming from the topics in the Kafka bus. This event processing platform could be implemented using tools like Apache Flink or Spark in order to filter and detect events within the data shared in the topics from the Kafka bus.

3.2.2.2 Energy monitoring

In this section is presented a solution for monitoring energy consumption and carbon footprint generated through the compute continuum. It has been deployed in ADRENALINE Testbed [MNC+17] and will be part of TeraFlowSDN release 4. This implementation is aligned with the architectural blocks described in section 3.2.1.1.

Within Figure 3-21, three distinct components are clearly separated Firstly, the compute continuum serves as the allocation ground for resources, and it is where the energy monitoring platform is tasked with gathering data pertaining to energy consumption. The general implementation involves deploying distinct components across the cloud, edge, and extreme edge, all orchestrated under a unified framework, exemplified by Kubernetes [K8S]. The proposed implementation is aligned with the enabler 2 design as shown in Figure 3-21.

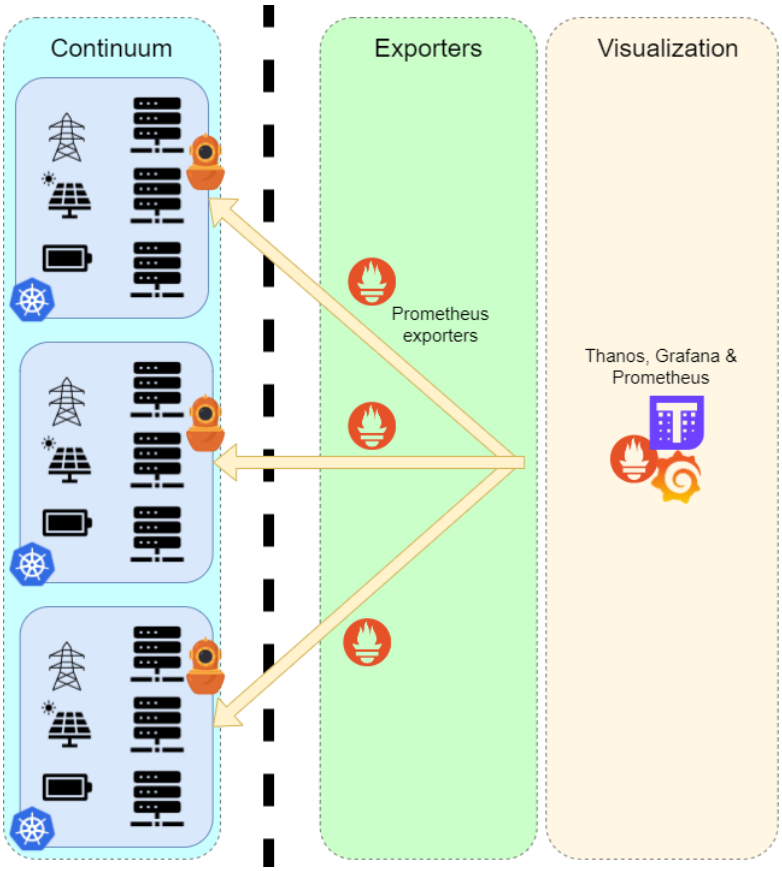


Figure 3-21: Energy Monitoring Platform implementation high-level diagram.

Secondly, the initial segment comprising the Energy Monitoring Platform (EMP) involves the use of exporters to extract energy data. The designated tool for this purpose is the Scaphandre [SCA24] agent, previously referenced in D6.2 [HEX223-D62], facilitating the retrieval of electrical power data from containers operating within a specified Kubernetes cluster. Each agent is deployed within the clusters for monitoring purposes. Additionally, it incorporates a Prometheus [PRO24] exporter to retrieve and transmit this data to a central Prometheus instance. The utilization of kube-state-metrics [KSM24] is also contemplated to acquire energy data not only at the container level but also across other hierarchical levels such as replica sets, namespaces, pods, etc., providing a comprehensive mapping of containers with their corresponding Kubernetes resources.

Figure 3-22: Dashboard for energy monitoring visualization.

Finally, the visualization phase within the EMP includes a central Prometheus instance continuously collecting data from exporters. Through predefined queries, this data is structured to meet specific requirements before being delivered to the Grafana [GRA24] instance for visualization through customized dashboards. Optionally, an instance of Thanos [THA24] is incorporated at this stage for data aggregation if the objective is to aggregate data from different clusters. An early implementation for this energy monitoring solution has been carried out by deploying the end-to-end solution in a single k8s cluster where the energy related data has been scraped, aggregated, stored and finally visualized through a Grafana dashboard that could be seen in Figure 3-22.

3.2.2.3 Monitoring platform for integration in closed loop

The preliminary implementation of the Monitoring Platform in the context of the zero-touch closed loops (see enabler #8 in section 3.8) is shown in Figure 3-23, where the Monitoring Platform is used as a closed loop (CL) monitoring function. The customization of the Monitoring Platform shown in the picture refers to the implementation integrated in PoC B (in particular the PoC with cobots deployed in one of the consortium partners -VTT- lab [PBM+24]), where the collection of monitoring data feeds two concurrent and hierarchical CLs that cooperate through a coordination model based on delegation and escalation (see section 3.8.1.1 for further details on CL coordination models). The first CL operates at the service and infrastructure layers. It automates the migration of tasks and application functions at the extreme-edge, among a set of 6G connected cobots, with a criterion that optimizes battery consumption and re-charging process while guaranteeing service continuity and reliability. The second CL operates at the network layer. It works on top of the SDN transport network that connects the edge nodes where the backend of the cobots' application is executed in the edge/cloud continuum. Its goal is to guarantee the interconnectivity at the transport level between the local User Plane Function (UPF) and the edge nodes running the backend application components. This is achieved by implementing suitable path recovery strategies with CL functions mapped in the modules of the SDN controller operating the target transport network (see section 3.8.2.2).

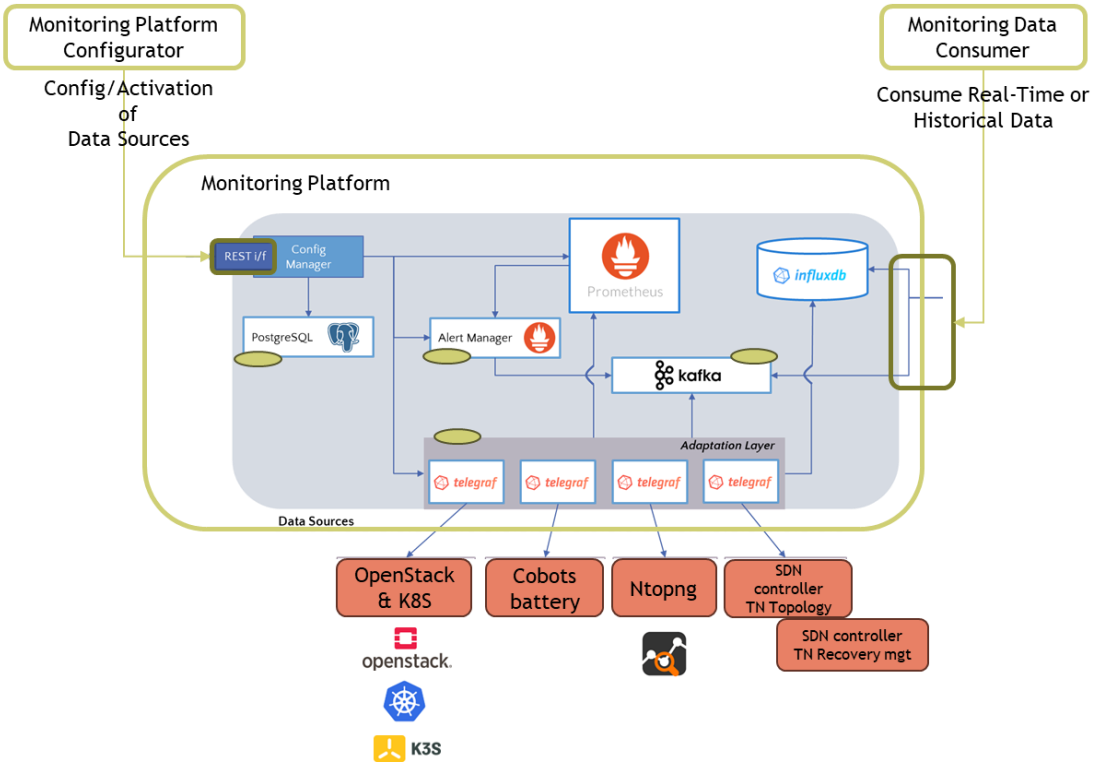


Figure 3-23: 6G Monitoring Platform in a zero-touch closed-loop instance.

It should be highlighted that the selection of the cobots application is just an example to test this preliminary implementation and to properly feed the PoC in WP2, however the implemented solution can be applied in different contexts with suitable customizations, to also deal with multi-vendor devices. This particular PoC has been selected to demonstrate the capabilities of the Monitoring Platform, since it represents a scenario with different types of monitoring data to be collected, from different layers (computing infrastructure, transport network) and different domains, including the extreme edge represented by cobots in this case. Moreover, the

kind of monitoring data consumers are entities, i.e., internal functions of the two CLs, which are deployed dynamically and need to configure and activate new data sources following an on-demand approach. In addition, these consumers may optionally implement ML techniques and, as such, they may need to access data either in real-time (e.g., for inference) or as stored historical data (e.g., for training or re-training purposes). All these characteristics allow to validate the full set of features implemented in the Monitoring Platform and well represent the variety of requirements expected for monitoring in future 6G networks.

Considering this target scenario, the Monitoring Platform is configured to retrieve data from four different sources. The first consumer processes information on cobots' battery level, collected from the cobots themselves, and usage of extreme-edge computing resources, collected from the related edge/cloud platforms. In the PoC the focus is mainly on the K3S [K3S] cluster installed in the cobots. Additional data sources at the service layer may be defined in the future to improve the ML algorithm for the prediction of battery consumption, e.g., related to tasks performed or planned for each cobot, type and characteristics of performed actions, etc. The second consumer is fed by information related to statistics on network traffic at the edge nodes, collected through network probes based on the Ntopng tool [NTO24], as well as events related to failures in network nodes and/or ports, and results of path recovery attempts gathered from the SDN controller.

In the PoC scenario, the Monitoring Platform constitutes a pre-instantiated Monitoring function, with two dynamic monitoring data consumers instantiated at the service provisioning time. During this phase the Monitoring Platform is configured to retrieve, store and expose new types of metrics. The configuration is handled by the Config Manager module, which exposes a REST API for activation and setting of new data sources. Internally, the Config Manager instantiates the new data collectors (implemented with Telegraf [TEL24]), the creation of new topics to stream the data on the Kafka bus [KAF24], as well as the configuration of rules for management of collected data, e.g., persistency time, pre-processing and alert generation rules, etc. The configuration of the various data sources is stored in a PostgreSQL database for persistency reasons. In the following step, the monitoring data consumers are configured with the parameters needed to consume the monitoring data. These parameters include the credentials to access the Monitoring Platform in read mode, and the details of the topics or the URLs to subscribe for or query the real-time or historical data available at the Monitoring Platform. During the service runtime, the consumers can thus read the data from the Kafka bus or the InfluxDB [INF24] time-series database of the Monitoring Platform. If needed, new data sources can be dynamically re-configured and/or activated during the service runtime, still interacting with the REST API exposed by the Config Manager.

Early Validation Results

As part of the closed-loop logic, the monitoring platform retrieves, through the implemented data collectors, the information related to the battery levels and status of the two cobots named *robo-pi1* and *robo-pi2*. The two cobots publish periodically around each second the information related to their batteries on an MQTT broker [MQT24] and the Monitoring platform retrieves, processes and collects such information in the time-series database InfluxDB and push into a Kafka bus, for historical and real-time data management, respectively.

Figure 3-24 shows the InfluxDB time series dashboards representing the collected information about the batteries' voltages and percentage levels, respectively. From the two dashboards, it can be noted that the blue lines represent the voltage and the percentage of a battery of *robo-pi1* cobot, around 12.3V and 80.66% around. In contrast, the purple lines represent the voltage and the percentage of the battery of *robo-pi2* cobot, around 12.04V and 72.65% around. The battery level values refer to a particular 45-minute time window and could have been changed over a longer period.

The results demonstrate the applicability of this solution for collecting data from different sources, including extreme edge nodes (from cobots in this case), to make them available for closed-loops operating at the service layer, thus contributing to the automation of service management, one of the functionalities that should be supported by the E2E system. Moreover, the implementation directly feeds one of the PoCs in WP2.

Figure 3-24: Monitoring platform dashboard showing battery data collected from cobots.

3.2.3 Conceptual solutions

3.2.3.1 Passive/in-band and active telemetry for TSN/DetNet networks

Given the critical nature of TSN and DetNet, monitoring is a very important component to verify that the network can meet its requirements regarding end-to-end latency and packet loss. There exist different monitoring strategies. With passive network measurements, data is gathered by passively listening to network traffic, i.e., without interfering with data traffic. Passive monitoring allows to collect simple statistics at switches and routers, such as bytes sent, lost packets, and other similar statistics. Conversely, active probing is an active monitoring strategy where artificial data packets (probes) are sent into the network, using the same links as the data traffic and collecting statistics along the way. Active probing can be used to e.g., measure the end-to-end delay along a specified path. Since active measurements generate additional network traffic, they interfere with the normal traffic flow, and as such they must be carefully planned. In-band network telemetry offers low-overhead monitoring possibilities, enabling an end-to-end performance view of the network (including end devices), while not injecting any new packets in the network. The idea is to embed monitoring information in the actual data packets, and as such information can be collected per-hop, per-packet and per-flow.

A monitoring algorithm for a TSN/DetNet network segment computes a suitable monitoring strategy based on the monitoring configuration and the current traffic flows in the network, as shown in Figure 3-25. This algorithm should carefully analyse the trade-offs between different monitoring strategies, and make sure that the monitoring overhead is kept between certain limits such that it does not interfere with the data traffic, which is one of the main challenges/requirements outlined in the framework of Operations, Administration, and Maintenance (OAM) for DetNet (RFC9551). In the case of active monitoring via probes, the optimal placement and deployment frequency of these probes are crucial challenges that need to be addressed. The resulting monitoring strategy might be a combination of passive, active, and in-band telemetry. After collecting the monitoring data, this data will be stored in a local database and can be consumed by external parties, or by a closed control loop (cf. Section 3.8.3.1).

The coordination of monitoring strategies in a multi-segment TSN/DetNet setup is another challenge, as a single traffic flow might cross multiple segments. This allows for the virtualization of probes across segments, and more importantly end-to-end statistics for traffic flows. The EW protocol, as described in Section 3.1.2.3, can be extended with end-to-end monitoring coordination capabilities for a multi-segment TSN/DetNet network.

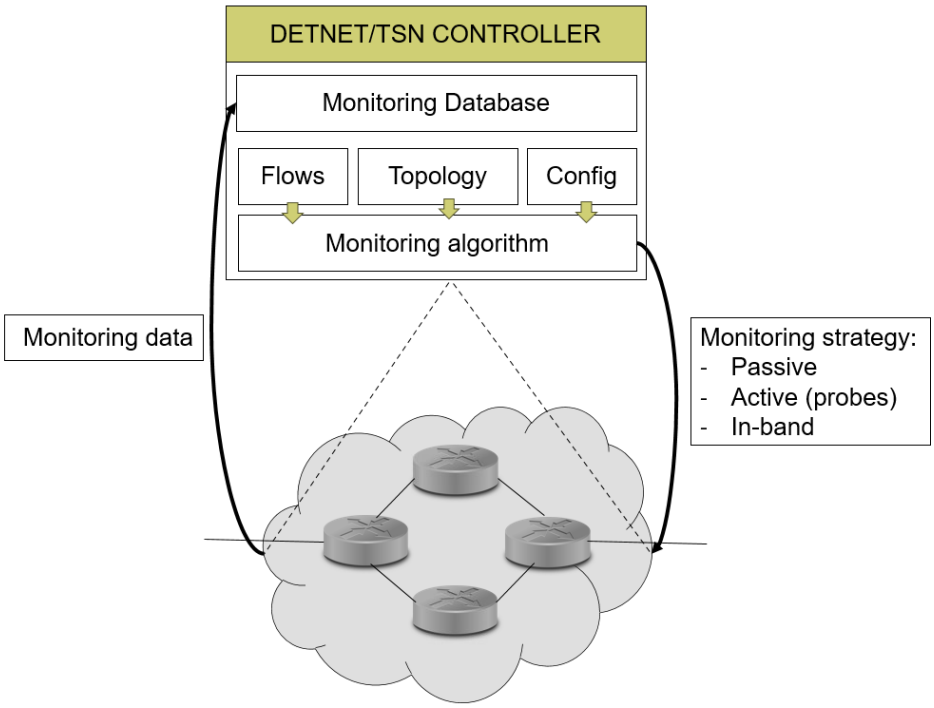


Figure 3-25: Monitoring for TSN/DetNet network segment.

3.2.3.2 Data fusion for signals correlation and remediation actions

The idea behind signal data fusion [TAZ+22] is concentrated on active and passive telemetry of different entities and sources of the system, aiming at facilitating agents to reactively mitigate system and network failures, as well as proactively preventing them altogether by analysing real-time information. A clear methodology of collecting the necessary data from the different entities contributing to a heterogeneous and distributed system is crucial to its efficient maintenance and orchestration. OpenTelemetry [OTL24] provides such a methodology for accessing data signals at real-time in distributed architectures. The OTLP (Open Telemetry Protocol) Collector provided by the OpenTelemetry framework comes in two different flavours supporting a variety of distributed data collection designs. The OTLP Gateway can be used for centralized deployments where data collection sources are directly accessible, while the OTLP Agent is available for exporters in distributed locations which forward the collected signals to the OTLP Gateways residing in centralized locations.

Data signals refer to performance information such as metrics, traces and logs collected by both infrastructure and network nodes. Application and system-related signals are collected by using OpenTelemetry instrumentation libraries, while TeraFlow SDN Controller's NBI provides accessibility to QoS metrics related to the performance of the network. As shown in Figure 3-26, this solution leverages the implementations of the OTLP Collector to deploy across the heterogeneous infrastructure of the computing continuum to collect local observability signals using the OTLP agent version of the collector, preprocess the data close to the source to identify early identification of events and finally aggregate this information to be fused with other signals at more 'central' locations using the OTLP Gateway version of the collector. Through the collection data bus, the collected information is processed for identifying anomalies or specific events at runtime and is then exported to the data fusion observability tool for further analysis and aggregation of the information, to assess higher-level orchestration components.

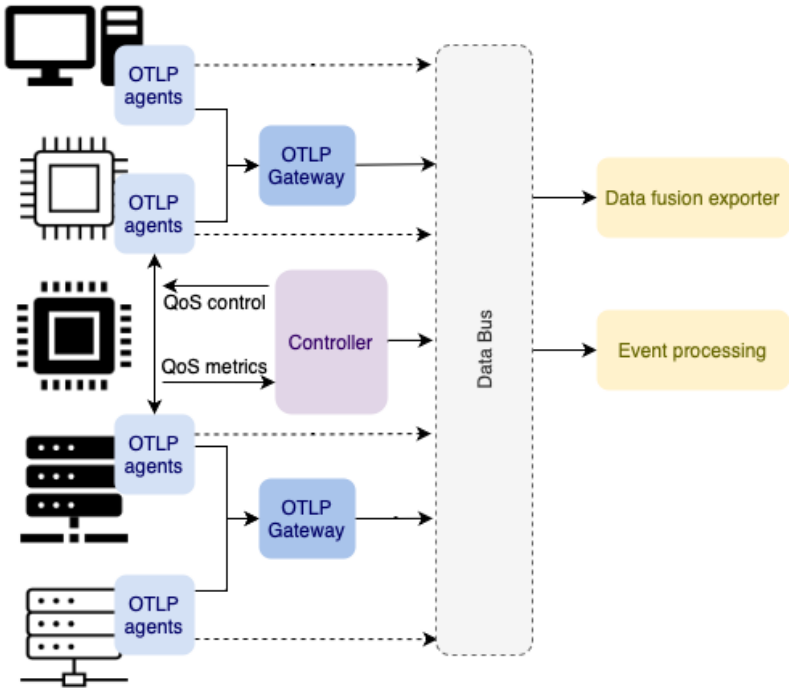


Figure 3-26: Observability signals data collection and fusion.

3.2.4 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this Enabler 2 are the following:

- **Scalability**: this enabler ensures that network monitoring and telemetry can scale to meet increasing demands without degradation in performance. The system's architecture, designed to handle cloud-scale operations, effectively manages large volumes of data and network activities, preventing bottlenecks that can impede scalability.
- **Latency**: The real-time data gathering capabilities of this system minimize latency in network monitoring and decision-making processes. By leveraging protocols like NETCONF, gRPC, and gNMI, the system can quickly collect, process, and react to data from various network components, ensuring timely responses to dynamic network conditions.
- **Flexibility**: The suite of automation technologies and the use of diverse protocols (NETCONF, REST, YANG, gRPC, gNMI, SNMP) enable this enabler to be highly adaptable to different network configurations and requirements. This flexibility allows network administrators to tailor the monitoring and management solutions to specific needs, enhancing the overall efficiency of network operations.
- **Reliability**: The enablers data acquisition mechanism underpins dependable network performance monitoring and fault detection, thereby increasing the overall reliability of network operations.
- **Automation**: Automation is a core component of this enabler, with protocols designed to facilitate the automation of network management tasks and the orchestration of network functions and policies. This allows for more efficient use of resources and reduces the need for manual intervention, thereby enhancing operational efficiencies and reducing the potential for human error.

Regarding the KVIs, it is considered that the most evident to be affected by this sub-enabler is **Sustainability** in what regards the following aspects:

- One of the distinctive features of this enabler is its focus on monitoring energy consumption across network elements, including computational resources. This capability is pivotal for developing algorithms aimed at optimizing energy use, thus supporting sustainable network operations. By enabling more energy-efficient network configurations and operations, this enabler helps reduce the overall environmental impact of network infrastructures.

3.3 Enabler 3: Management capabilities exposure framework

Hexa-X-II is intended to provide highly configurable solutions which may be tuned to the requirements of different 6G stakeholders, as introduced in section 2.3. Semantics (referring to network, cloud, zero-touch…), operational domains, and infrastructure domains are used to categorize these capabilities. Programmable compositional patterns [IS21] are considered required for the integration of these capabilities, with an emphasis on plug-and-play methods and cloud-native strategies. In order to accomplish this, a management capabilities exposure framework — which offers a service bus for smooth Hexa-X-II capability interoperability — is considered. This fabric enables integration between administrative domains and capacities by implementing features such as connection, dependability, security, and observability. By enabling these features, this component accomplishes two primary objectives: it first enables the modularization and statelessness of management services, allowing them to be deployed as scalable, containerized microservices. Second, it facilitates the system's transition to APIfication, in which traditional interfaces, i.e. older methods of system interaction and communication, like tightly coupled integrations, and legacy protocols like RPC (Remote Procedure Call), SOAP (Simple Object Access Protocol) etc., are replaced with HTTP-based RESTful APIs, which handle producer-consumer interactions. Deliverable D6.2 [HEX223-D62] presented the SoTA and expected beyond SoTA of this enabler. A draft architecture definition was also pointed out, along with the relevant internal and external interfaces. This deliverable presents the work carried out to realise the architecture planned in deliverable D6.2, the improvements to the first design, and the first deployment results.

3.3.1 Enabler design

Figure 3-27 shows the high-level architectural view of the suggested solution. Thanks to the use of a message broker, the design is event driven. The central component of the architecture, the broker, enables communication between the external services serving as a shared message bus and the framework internal components. The establishment of communication channels among the different components facilitates the connection and global orchestration of the operations of the various enablers.

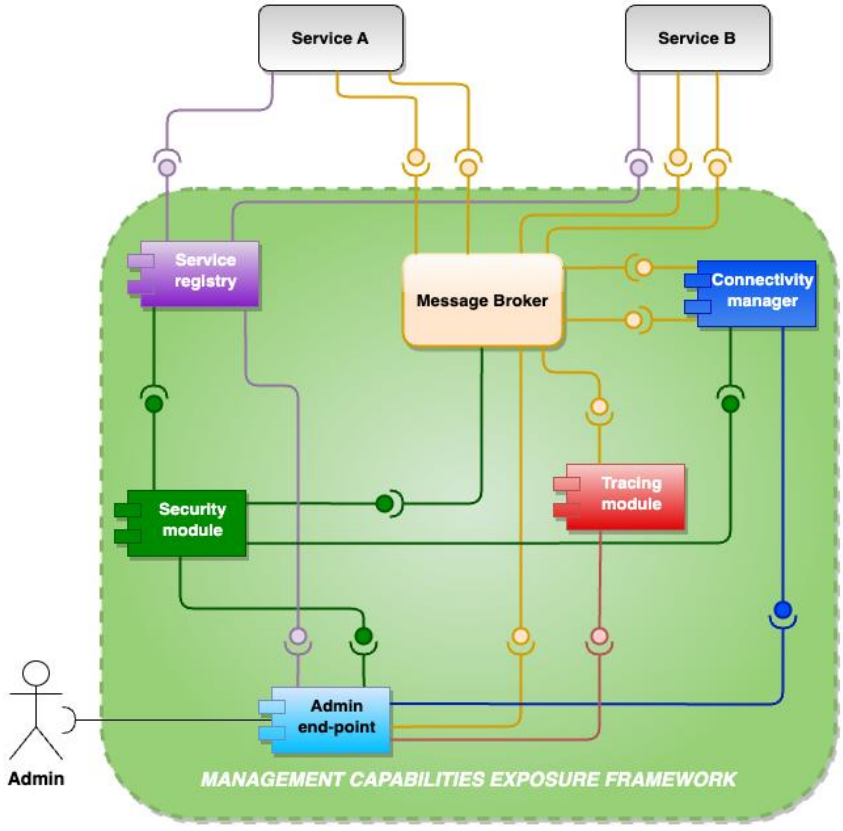


Figure 3-27: High level architecture of Management Capabilities Exposure Framework.

The implementation draws inspiration from the ETSI ZSM Integration Fabric [ZSM-002]. This component is a key element within the wider ETSI Zero-touch Network and Service Management (ZSM) framework and is

responsible for facilitating seamless integration and interoperability between a variety of management and orchestration entities in a network environment. It acts as a centralised middleware layer which enables communication and data exchange between different ZSM components, such as network controllers, orchestrators, and management systems. The ETSI ZSM Integration Fabric plays a pivotal role in the realisation of the vision of zero-touch automation. It provides a set of standardised interfaces, protocols, and mechanisms which facilitate the orchestration of network resources, the automation of service delivery, and the optimisation of network operations. As the Management Capabilities Exposure Framework would be set to serve as the central component to facilitate interoperability in the E2E system, as stated in section 2, the design principle of the ZSM Integration Fabric would align perfectly with the design view.

Further details of the internal architectural definition, in addition with early implementation details and validation findings will be presented in the following sections.

3.3.1.1 System components

The components of the Management Capabilities Exposure Framework are described in Table 3-1.

Table 3-1: Management Capabilities Exposure Framework components.

Component	Description
Services	Consist of network functions, management systems, and external applications, which need to communicate with each other. Each of them is isolated from each other, unaware of the topology. They can communicate only by interfacing with the integration fabric. To have a standardized way to connect and deploy them, each of these services would be implemented in the form of containers. Each of them alongside the container scope is equipped with a client that makes it possible to communicate with the message broker.
Service registry	It would have two main functions: it is the creation/deletion services subscriptions and manage inter-service relations. It is the insertion and removal endpoint for new services. This single-entry point makes it possible to better regulate the inter-service relations updates and address the horizontal scalability also in a multi-tenancy scenario. Inserting a new service means subscribing it to the base functionalities' topic (alarm, coordination…). After this registration step the service will not be aware of the current structure around it, but it will be aware of the relevant events happening in the architecture. One service that requires to take advantage of integration fabric services must start a one-time onboarding procedure, that may be built as a volatile process, exploiting virtualization technologies like serverless computing. The second task shipped out by this module embraces all the operations to keep the list of services managed by the integration fabric updated. It shows the amount and the type of services managed, and it keeps the current policies used, relations within the system in terms of topic and subscriptions in the scope of the integration fabric up to date. This enables service discovery features, crucial in scope of the integration fabric.
Security module	Secures access to the resources managed by the integration fabric, as well as the communication within the modules. It ensures AuthN/Z mechanisms to guarantee that the resources are accessed only by accredited stakeholders. It enables to manage token/key to unlock different QoS or certain services. Its last feature is to manage the encryption within modules (Transport Layer Security - TLS) to avoid leaking information in the inter-service communication.
Connectivity manager	It offers advanced management in terms of subscription, retention policies and all the aspects related to the management of topic and related queue. The connectivity manager routes and distributes traffic between services based on defined policies and subscriptions. This module is connected, directly, to the message broker to modify the subscription. A change in the subscription implies a change in the involvement of a particular service in the system. In fact, the interaction of the involved services depends on the consumed and published messages, and for this the service must subscribe to a certain communication channel, i.e. a topic. The modification of a subscription therefore leads to a modification of the communication patterns within the systems.

Tracing module	The tracing module oversees recording and following the requests flows through various services. Analysis, monitoring, and troubleshooting of the system is made possible by tracing
Admin endpoint	Observability is an important point of the Management Capabilities Exposure Framework, enabling the continuous check of the health and key metrics of the system. This is possible because this endpoint embeds a tracing and metric generator module (that refer to the enabler performances) that are connected to the topic of interest. An admin endpoint is also important to have a clear view and control at a high level on the general services relations, and routing rules between services. That block would be directly connected to the management's blocks of the enabler architecture, i.e., the connectivity manager and service registry, and the admin endpoint is secured through the security module, to ensure only authorized subjects can access this integration fabric control panel.
Message broker	It is the central component of the architecture. It represents the central node in which all packets are routed according to the subscription, the available topics, and the retention policies. Each service will be asynchronously connected to this module. From the broker each of them would receive all the information of the other components and will react to certain event or request by directly publishing on the broker itself (event-driven architecture).

3.3.1.2 Internal Architecture of the system components

The enabler at this stage is composed by two main macro components:

- The Kafka [KAF24] cluster and all the related services, more details are presented in Section 3.3.2.1.
- The services that manage onboarding/offboarding, listing and security aspects of the framework.

Figure 3-28 shows the general structure and how the internal components of the framework communicate each other. The core of the framework is the Apache Kafka cluster, which implements the event-driven architecture at the heart of this framework. It makes use of the Apache Kafka protocol [KAFP24], a high-performance binary messaging protocol that enables the efficient and reliable exchange of streamed data over TCP between producers and consumers in a distributed, fault-tolerant and scalable manner.

The management of the data stream is topic-oriented, using the concept of queues. A Kafka queue is a logical stream of records belonging to a topic. Queues are not a single stream, but a more granular structure divided into partitions. Partitions are ordered, immutable sequences of records within the queue, identified by unique offsets. This allows data to be replicated over multiple brokers, and allows multiple consumers to read data simultaneously, providing parallelism, horizontal scaling, and fault tolerance. This structure makes Apache Kafka a highly scalable, distributed, and fault-tolerant streaming platform that enables real-time processing of continuous data streams. As a result, with Apache Kafka:

- It is possible to handle massive amounts of data with high throughput and low latency.
- The distributed architecture provides data redundancy and fault tolerance, allowing applications to continue to operate in the event of machine failures.
- Provides strong data durability and flexible retention policies, allowing a high degree of freedom in managing message persistence. Users can either store and access data for an extended period of time or persist for a limited period of time.
- This feature fits perfectly with the design vision of the enabler, which needs to be flexible, consistent, responsive and easily integrated with a discrete number of actors.

Another key benefit of using Apache Kafka is its Access Control Lists (ACLs) feature. ACLs provide fine-grained control over user access to Kafka resources such as topics, consumer groups and brokers. This feature is critical for organisations that require strict data security and regulatory compliance, as it allows them to effectively manage and monitor user access to prevent unauthorised access and data breaches. The added value is that it is fully compliant with an mTLS approach (chosen as method to secure communications within the system), using the certificate as an identity to encrypt communications between the actor and the Kafka cluster.

Finally, it is important to note that Apache Kafka is a mature open-source project with a large community. The solid background of the project makes it able to offer a lot of integration with a plethora of frameworks and

programming languages, which is crucial in a multi-domain scenario, as the one have to deal with this framework.

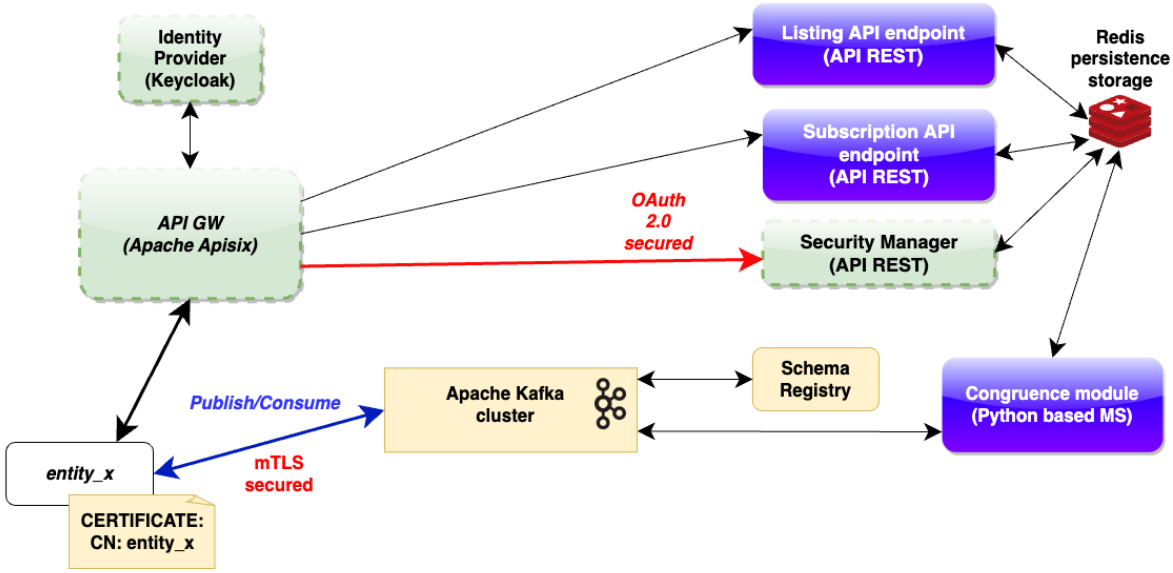


Figure 3-28: Management capabilities exposure framework main components and interactions overview.

Figure 3-28 highlights that an enabler willing to exploit the framework can interact in two ways: using REST APIs with HTTP protocol or using Apache Kafka protocol. In the former case, the new entities do not interact directly with the services REST API interfaces, but all the requests are received by a centralized API gateway in charge to forward them the requests. This makes possible to have an additional control on the received requests, in term for instance of traffic control, and, in addition, with the support of an Identity provider to secure a route. For instance, the route connecting the enabler with the Security Manager is OAuth2.0 secured, to avoid leaking security information.

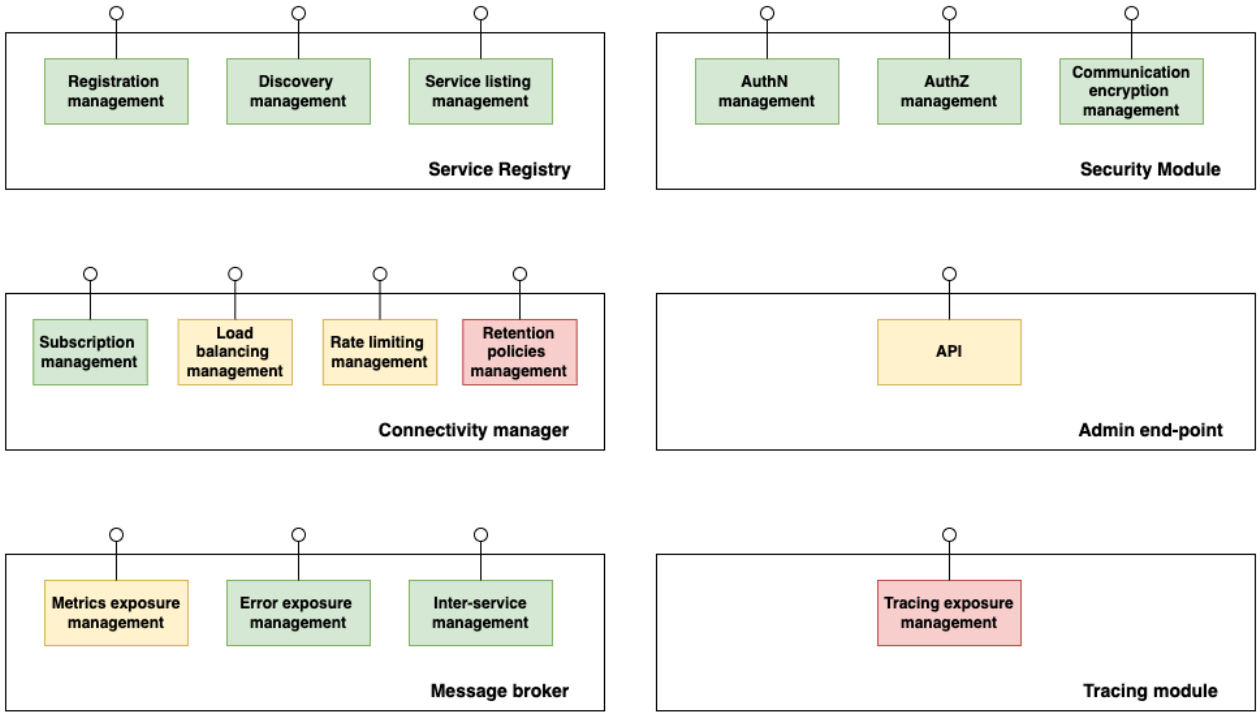


Figure 3-29: Management capabilities exposure framework interfaces.

The combination of the API Gateway and the IdP basically represent the AuthZ/AuthN interfaces, shown in Figure 3-29, of Security modules (displayed both in Figure 3-27 and Figure 3-29). The latter case involves the actual communication with the system, i.e. using Apache Kafka protocol. An enabler that is already onboarded

to the functionalities of the Management capabilities exposure framework would be able to publish to and/or consume from a set of topics. This kind of communication is secured through the usage of mTLS (mutual TLS), that makes possible to jointly encrypt the communication and give a method to identify the requesting party. Each entity interacting with this enabler would be identified by the certificate (distributed by the usage Security Manager REST endpoint shown in Figure 3-28, which API is described in Appendix 8.2.3) that is used for the TLS connection with Apache Kafka cluster. As consequence, the authorization would be proportioned by mean of ACL (Access Control List) offered by Apache Kafka, using the CN (Common Name) declared in the certificate. A more detailed description of this interaction will be described in the next Section 3.3.1.3.

Figure 3-29 reports the entry points of the management capabilities exposure framework, for its main components. Each interface would cover a different aspect of the communication, such as on boarding of new entities, service listing of the entities that are plugged to the management capabilities exposure framework, security and monitoring. The interfaces are marked with different colours to distinguish the different stage of development of the system. The ones highlighted in green are related to core functionalities, fundamental to deliver a basilar service. Given that, these functionalities are prioritized. The ones in yellow will be delivered at a later stage, providing not core functionalities but added values to complete the solution. The red ones will be developed in the final stage. Table 3-2 describes the interfaces for the core functionalities of the management capabilities exposure framework and identifying protocols and technologies that could be used as baseline for their implementation.

Table 3-2: Management capabilities exposure framework interface mapping.

Interface	Description
Registration management	It is the system interface that guarantees the onboarding of new enablers in the integration fabric scope. It basically creates, after a REST query bringing all the key information, a new topic in the message broker related to the registered service. In addition will create a consumer group and a consumer instance, to the message broker, to link the new added service. Finally, this interface will make it possible to change the settings of the registration, with ad hoc REST queries.
Discovery management	This interface querying the message broker forward the information to communicate with another enablers. In detail, after a REST call, it forwards all the useful information of the topic related to the requested enabler.
Service listing management	It has the role of forwarding the information about all the enablers registered to the integration fabric. Querying this interface with a REST call it forwards a list of all the topic registered in the message broker.
Subscription management	This interface will permit to manage the subscription of all the enablers by the admin, and to manage internal creation/revocation of the subscription of a certain topic.
Error exposure management	This interface will expose the information published to the topic that collects the errors of all enablers. This can be queried with a REST API in an asynchronous way.
Inter-service management	It is essentially represented by the message broker. Each Kafka topic can represent (i) a communication channel that a producer, i.e. an enabler, can use to deliver information to an authorized consumer, i.e. another enabler, (ii) a broadcast topic that deliver global info that are of interest to all the enablers, for instance critical errors.
Authentication management	Authentication will be offered using a 3rd party IdP (Identity Provider), to manage the identities in a more secure way.
Authorization management	Authorization features are offered as a combination offered by the ACL of the message broker and the OAuth of the IdP.
Communication encryption management	This feature is guaranteed by the functionality natively offered by the message broker, that leverages TLS.

3.3.1.3 Workflows

Figure 3-30 shows the sequence of operations for a single requesting actor for subscribing to the Management capabilities exposure framework services. The entity to onboard must interface with the subscription manager via a RESTful API call. This API is built using OpenAPI 3.0 specification. That makes it easier to integrate, speeds up development, and improves collaboration and use among developers, consumers, and other stakeholders. API description using OpenAPI 3.0 provides a standard, machine-readable format for defining and documenting RESTful APIs. This promotes consistency, interoperability and reusability across platforms and programming languages. In the message body the requesting actor should include all the requested information for the onboarding as well as the necessary info to authenticate that it is a trusted party. After the subscription manager has proven the identity of the requesting actor, it will directly communicate to the message broker. Using the info passed by the requesting actor, this will create a custom topic for the requesting actor, a consumer group, and a consumer instance. After all this roll-out is completed, the subscription manager will send all this information to the requesting actor in order to make it able to communicate with the message broker correctly.

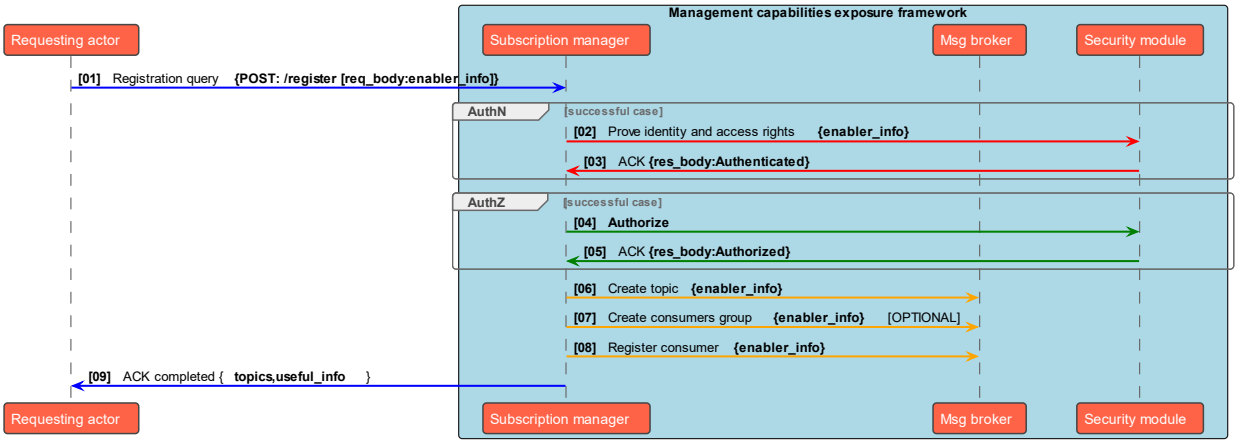


Figure 3-30: Registration to Management capabilities exposure framework.

The flows presented in Figure 3-31 and Figure 3-32 represent the functioning of the listing calls. The first is the global topic listing. When a requesting actor asks for the list of all topics, after being authenticated, the listing service will forward a list of all topics of interest to it. This means, that in addition to the global topic used for errors and general information broadcasting, the response body contains the list of all topics to which the requesting actor has access. Each of them specifies the name and the related access rights (consume/produce, consume only). When a requesting actor asks for information of a single topic, in addition to the above-mentioned information, it receives additional information on this topic. These details include the number of replicas, partitions, how the replicas and partitions are distributed among the brokers, last offsets, and finally the list of entities that have access to the information and the related access rights.

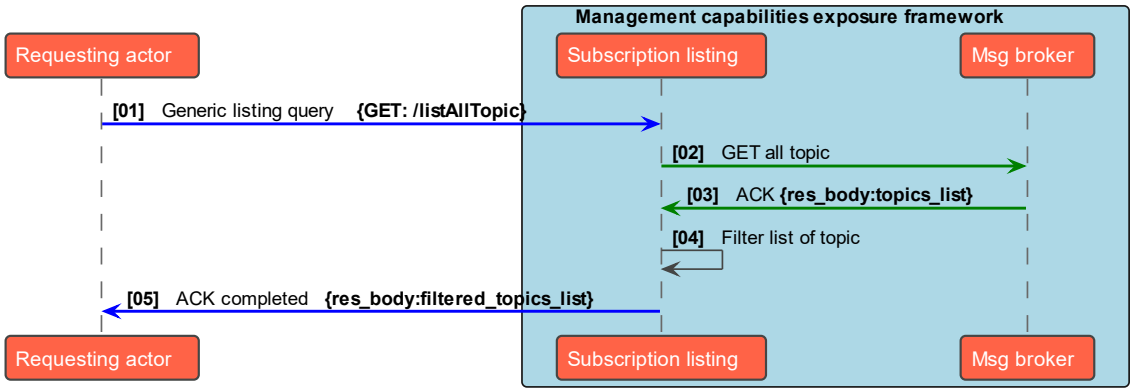


Figure 3-31: Management capabilities exposure framework topic listing.

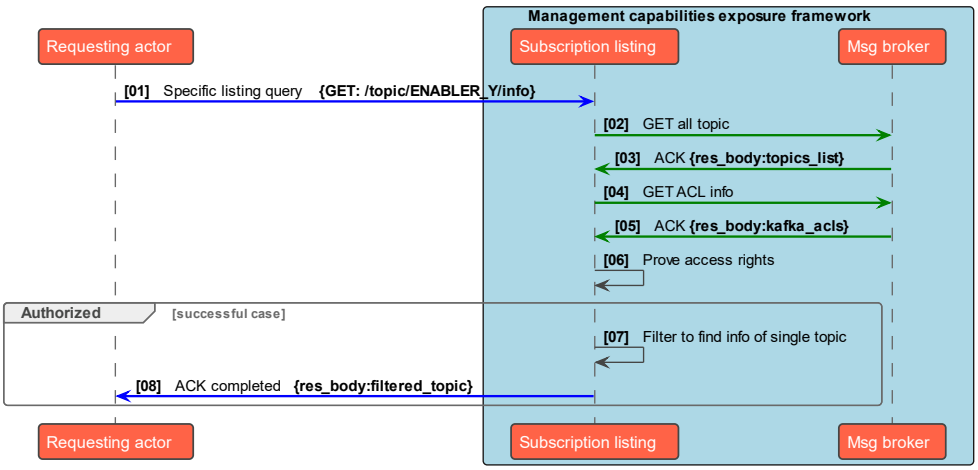


Figure 3-32: Management capabilities exposure framework single topic information retrieval.

The last workflow represented in Figure 3-33 shows the operations performed by the requesting actor when it has to produce or consume. First, out of the loop, the message broker is constantly updating the permissions that each requesting actor has, by using the native Apache Kafka features offered by ACL. All the information about identity is stored by the security module that oversees triggering the updates to the ACL of the message broker when an identity is added or changed. Based on these access rights, the client, i.e. the requesting actor, will be able to produce or consume to/from a specific topic.

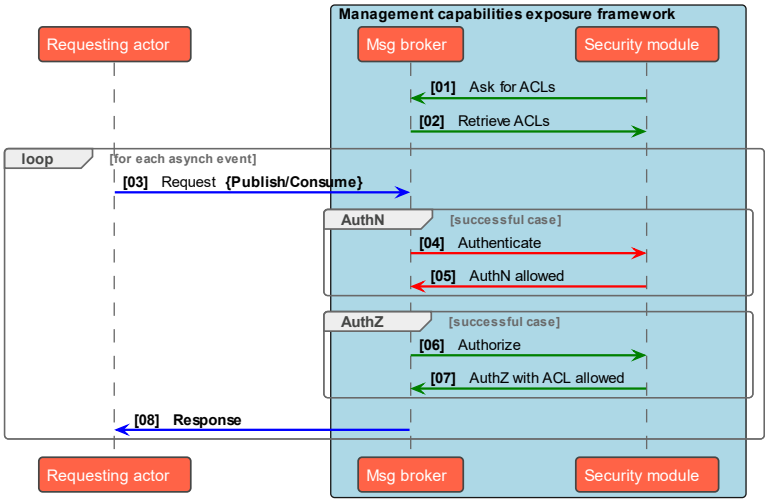


Figure 3-33: Enabler communication with Management capabilities exposure framework.

The interfaces of the management capabilities exposure framework are reported in Appendix 8.2.3.

3.3.2 Preliminary implementation and early validation results

An initial prototype of this enabler is being developed and tested at one of the partners' local testbed (Telefonica), which will be made available to other partners in the consortium. The integration with the other partners is planned in the PoC B.1. The framework developed will serve as the central communication bus and exposition of interface in the E2E architecture. In addition, once the aforementioned integration has been completed and the results achieved, the implementation will be proposed to the ETSI ZSM community as a reference implementation of the ETSI ZSM Integration Fabric [ZSM-002]. The following section presents additional information regarding the implementation works performed.

3.3.2.1 Apache Kafka cluster structure

The Apache Kafka cluster is the core component for providing the communication medium offered by Management capabilities exposure framework. The Figure 3-34 shows the internal structure of the cluster. At first glance it is possible to see that the cluster is deployed using KRaft methods. This mean that no ZooKeeper

is needed, and the cluster will be self-managed by the nodes exploiting the Raft Consensus. The structure in details will be made by:

- Two Apache Kafka controller nodes. These nodes are the one that made the Control Plane. This means that they are in charge to coordinate the communication, like partition leader election, the replicas distribution and topic partition assignation to the different broker nodes.
- Three Apache Kafka broker nodes. These nodes are the actual message brokers, in which the message is exchanged, and the topic subscription are managed. This can be referenced as Data Plane.
- A Schema Registry. This module oversees maintaining updated and storing the schemas used during the communication with the topic. The usage of Schema registry to regulate communication is not strictly mandatory but is desirable to have a trace of the exchanged message structure and make easier integration between parties.
- A REST Proxy. This endpoint makes possible to publish to/consume from the Kafka cluster using REST operations. This make possible to integrate communication with parties that is unable to deploy a Kafka client. Unfortunately, it is impossible to subscribe to a topic and emulate the behaviour of Apache Kafka protocol. The interactions are synchronous and punctual, so to retrieve more message you have to poll the REST Proxy several times.

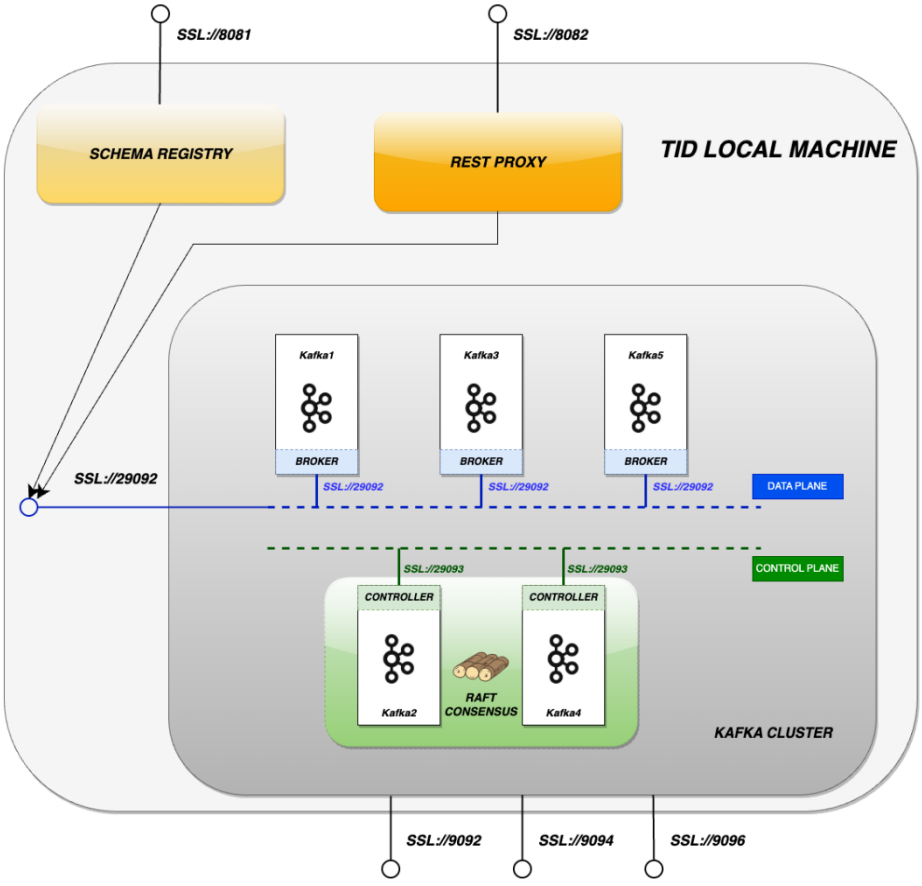


Figure 3-34: Overview on Apache Kafka cluster structure.

3.3.3 Impacted KPIs and KVIs

The following are the primary KPIs that are thought to be impacted by the use of enabler 3:

- **Scalability**. This enabler is made to communicate with any third party that embeds a Kafka or REST client. This increases scalability by enabling the quick creation of unique, fixed or volatile communication channels inside or between domains.
- **Latency**. The purpose of this enabler is to make it easier to link and communicate with enablers in a multi-domain scenario. It is well known that latency can be affected by relocating resources between

domains. Communication latency can be reduced by offering a better communication channel that improves the exposure of capabilities inside the network continuum.

- **Availability**. This enabler's event-driven architecture allows for increased network programmability and automation, hence improving availability. Recovering from runtime failures in a short amount of time is made feasible by the ability to set up communication channels using basic REST API calls. By doing this, the overall downtime caused by communication problems amongst the network continuum's enablers can be decreased.
- **Reliability**. One of the goals of this enabler is to provide dedicated global channels of communication for failures and status updates pertaining to currently onboarded entities. Having a real-time health view of communications between resource onboarded can help prevent service degradation and reduce the overall number of failures.
- **Elasticity**. This enabler provides an elastic communication medium by automatically providing and de-provisioning resources in response to changes in workload. In practical terms, this involves adjusting the communication channel according to the workload need.
- **Maintainability**. The system is intended to be easily maintained, particularly in terms of the on-boarding/off-boarding of resources that use this enabler. This can be done in a fully automated manner and facilitated by an easy-to-use REST API.

Among the KVIs, **Trustworthiness** is thought to be the most obvious to be impacted by this sub-enabler, followed by **Inclusiveness** and **Sustainability**.

3.4 Enabler 4: Security and trustworthiness framework

As 6G promises to revolutionize connectivity and bring about significant societal and technological advancements, prioritizing security and trustworthiness will be key to realizing its full potential. This enabler would result into the specification of a self-contained framework that integrates components from the different sub-enablers, as illustrated in Figure 3-35.

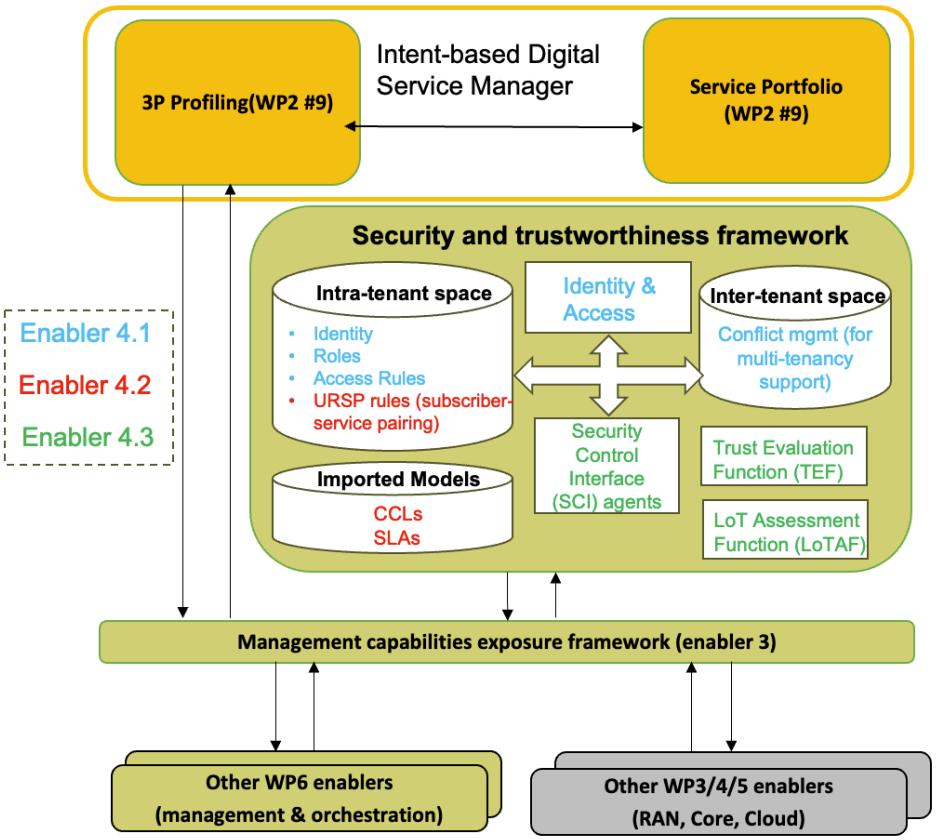


Figure 3-35: Security and trustworthiness framework.

The framework provides a set of functionalities oriented toward security and trustworthiness of 6G networks, encompassing various critical aspects such as data protection, privacy, network integrity and resilience. With regard to security, the sub-enabler 4.1 and 4.2 are of particular interest. Conversely, the 4.3 sub-enabler is responsible of trustworthiness. The design of sub-enabler 4.3 includes three sub-components. Currently, only the Trust Evaluation function has been addressed. The remaining two sub-components, namely the Security Control Interface (SCI) and the Level of Trust Assessment Function (LoTAF), will be presented in the next deliverable. The associated sub-enablers will be described in the following sections.

3.4.1 Sub-enabler 4.1: 3rd party resource control separation enabler

3.4.1.1 Sub-enabler design

Third party resource control separation in 6G networks is crucial for maintaining security, privacy and trustworthiness. Following the actors' definition from section 2.3, this sub-enabler focuses on the conceptual design and development of mechanisms to provide tenants with segregated yet customized management spaces. The management space defines what the tenant is authorized to do with regards to i) the operation of their allocated services, and ii) the control of their applications, including their interaction with the offered resources, allowing for a frictionless network-application integration. To make sure this works in resource sharing environments, these management spaces must be provisioned with permissions that do not conflict with each other. In this regard, the mechanisms will be built upon the access control framework represented in Figure 3-36. This framework is composed of the following five entities:

- **Resource Owner**: entity that can grant access to a protected resource (compute, memory, network, or even a service). This corresponds to Capability Operator (COP).
- **User:** entity that requests access to a protected resource. This corresponds to a tenant.
- **Resource Server**: entity hosting the protected resource. This would be a management function, which offers protected resources through APIs. These interfaces would be registered and discovered using the integration fabric (enabler 3).
- **Authentication Function**: used to verify the identity of a tenant.
- **Authorization Function**: determines what a tenant can and cannot access.

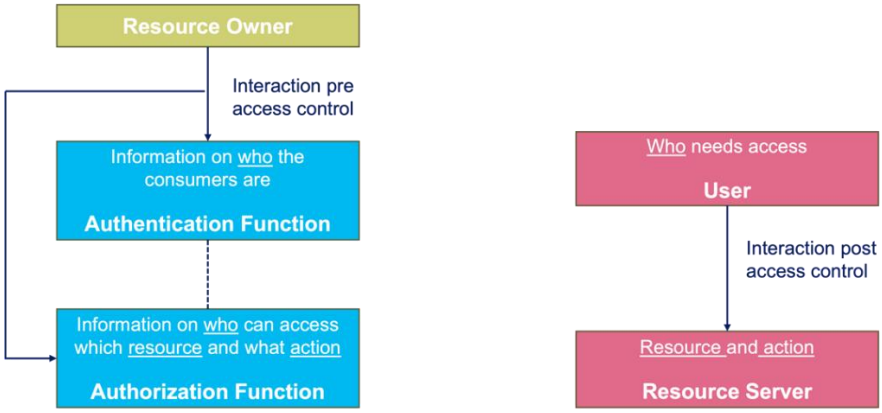


Figure 3-36: Access control governance framework for enabler 4.1.

The mechanisms developed in this framework would adhere to the conceptual model represented in Figure 3-37. This model consists of the following entities:

- **Access rule**: it represents a granular set of *permissions* to act upon a *protected resource*. Each permission would be a tuple <action, policy> that specifies whether a certain action (CRUD operation) over the protected resource is allowed or denied. The access rules are pre-defined by the COP at design time.
- **Role**: would represent a set of access rules. It enables the storage of information as to what permissions over which resources a consumer can work upon. The access rules would be pre-defined by the COP at design time.

- **Identity**: would represent the properties of a tenant in the M&O system. It is used for authentication and authorization, upon tenant's onboarding. By associating the identity with one or more roles, the management space for that tenant can be defined.

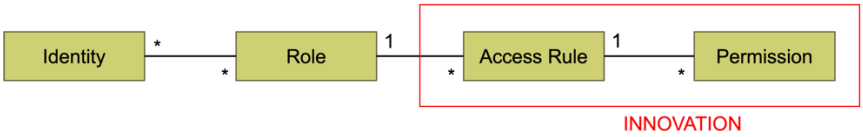


Figure 3-37: Conceptual model for the mechanisms in enabler 4.1.

The reader might notice the resemblance of this model with Role-Based Access Control (RBAC). Indeed, these conceptual models use RBAC as baseline, and extends it with new entities: access rules and permissions entities. This extension, which is required to handle the surgical control that tenants may require in resource sharing environments, has not been defined so far in the industry, at least following model-driven approaches (a must to achieve full automation). This is the specific innovation that Hexa-X-II brings for this feature. The contents presented in this, and the following sections pertain to a conceptual design, with no provision for subsequent implementation.

3.4.1.1.1 System components

This section focuses on those components that are inherent to enabler 4.1, highlighted in blue. These components are the following:

- **Intra-tenant space**: it would specify the management space provisioned to each registered tenant. This management space would be the result of mapping information from 3P profiling component (within the Intent-based Digital Service Manager, see [HEX223-D22]) into properties that the M&O layer can interpret and act upon. For the purposes of resource controllability separation, the intra-tenant space would implement the conceptual model presented in Figure 3-37. For further details on the solution specification for this model, see section 3.4.1.1.2.
- **Inter-tenant space**: it would specify the collection of policies providing multi-tenancy support in resource sharing environments. These policies are aimed to help the COP manage (i.e., detect and resolve) conflicts existing among two or more management spaces. For example, a conflict may occur when two management spaces allow write attributes over the same resource; in this case, two tenants will be able to update the same attribute with different values results, resulting into a race condition on resources that may compromise the system stability. The inter-tenant would be composed of two groups of policies:
  i. priority-related policies, based on tagging management spaces with different priorities. This would work as follows: if two conflicting management spaces have different priorities, then the management space with higher priority becomes eligible for application.
  ii. pre-emption-related policies, based on enabling pre-emption capability for selected management spaces. These policies would address the gap when two or more conflicting management space have the same priority; in such a case, it is impossible for the COP to determine which one needs to be applied.
- **Identity & Access**: it would provide the authentication and authorization functionality with regards to access control. As anticipated in Figure 3-36, *authentication* is used to verify the identity of users or entities attempting to access, whereas *authorization* determines what users can and cannot access.

3.4.1.1.2 Internal Architecture of the system components

This section focuses on the internals on the intra-tenant space component. As said earlier, the resource controllability features of this component are implemented with the model represented in Figure 3-38. The first solution specification of this model was reported in D6.2 [HEX223-D62]. The solution has evolved since then, with the sight set on optimization and UML compliance, resulting into a much more streamlined solution, as shown in Figure 3-38. In the final version of this solution, "identity", "role" and "access rule" entities are modelled as classes (also referred to as Information Object Classes [28.622]), whereas a "permission" entity is modelled with a <<datatype>> (i.e. standalone set of attributes [28.622]). The detailed information model is reported in Appendix 8.1.1.

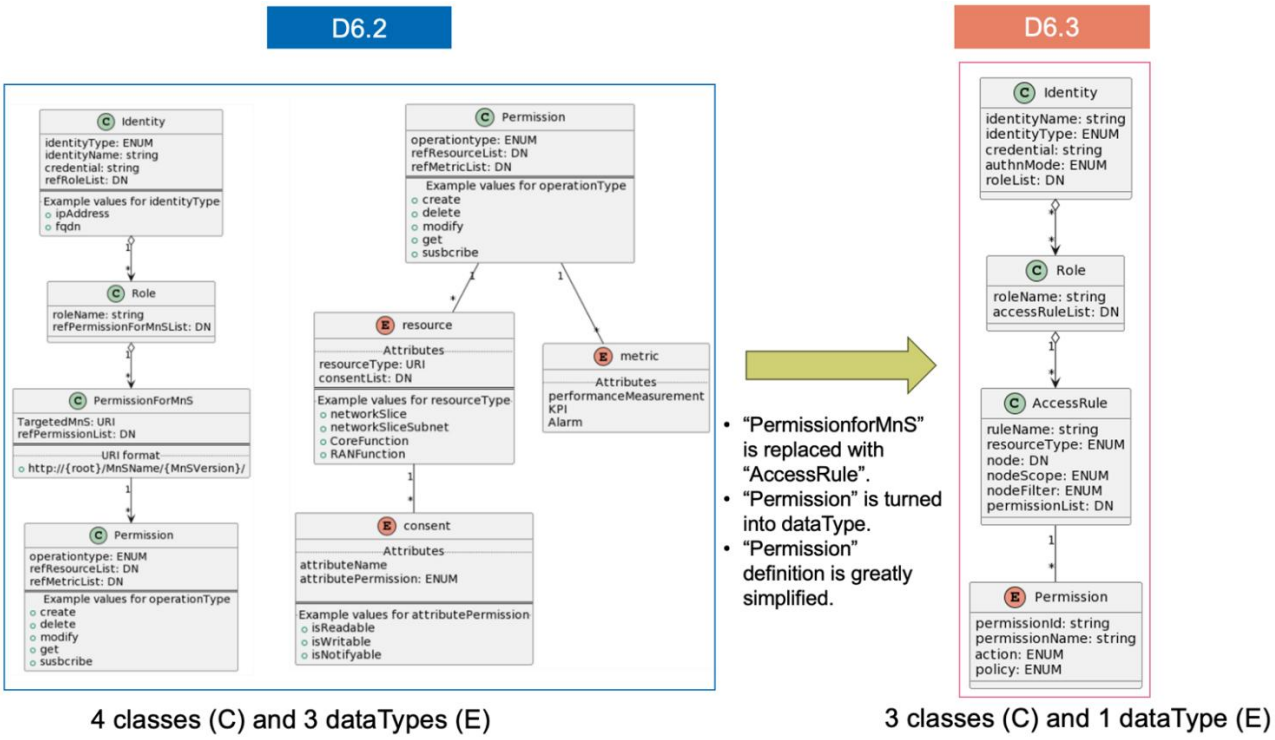


Figure 3-38: Trustworthy 3P information model.

3.4.1.1.3 Workflows

The execution logic of this enabler would span three different stages:

- Design time: in this stage, the system administrator (the COP) specifies granular access rules and associates them with pre-defined roles. These roles represent archetypal consumer types. At this stage, the tenant (the 3P) does not exist in the DSP domain (see section 2.3 for DSP definition).
- 3P onboarding: this stage is triggered once the tenant gets registered into the DSP. The DSP stores tenant information in the "3P profiling" component. Next, the COP should be able to define a tailored management space for that tenant. To that end, the DSP sends relevant information from "3P profiling" component to the Security and trustworthiness framework (enabler 4), through the management capabilities exposure framework (enabler 3). The received information is then mapped to an instance of the "identity" class, which is in turn associated to one or more pre-defined roles. The "identity" class instance is stored in the intra-tenant space. From this moment on, the tenant is successfully onboarded into the COP system.
- Operation time**:** once onboarded, the tenant can gain access to M&O enablers handled by the COP (bundled into management functions) and start consuming their capabilities via granular access control.

Figure 3-39 shows the workflow for the design time. As seen, the COP first creates instances of access rules (steps 1-3), and then roles (steps 4-6). The instances are stored in the Management Information Base (MIB) [28.622]), also referred to as inventories.

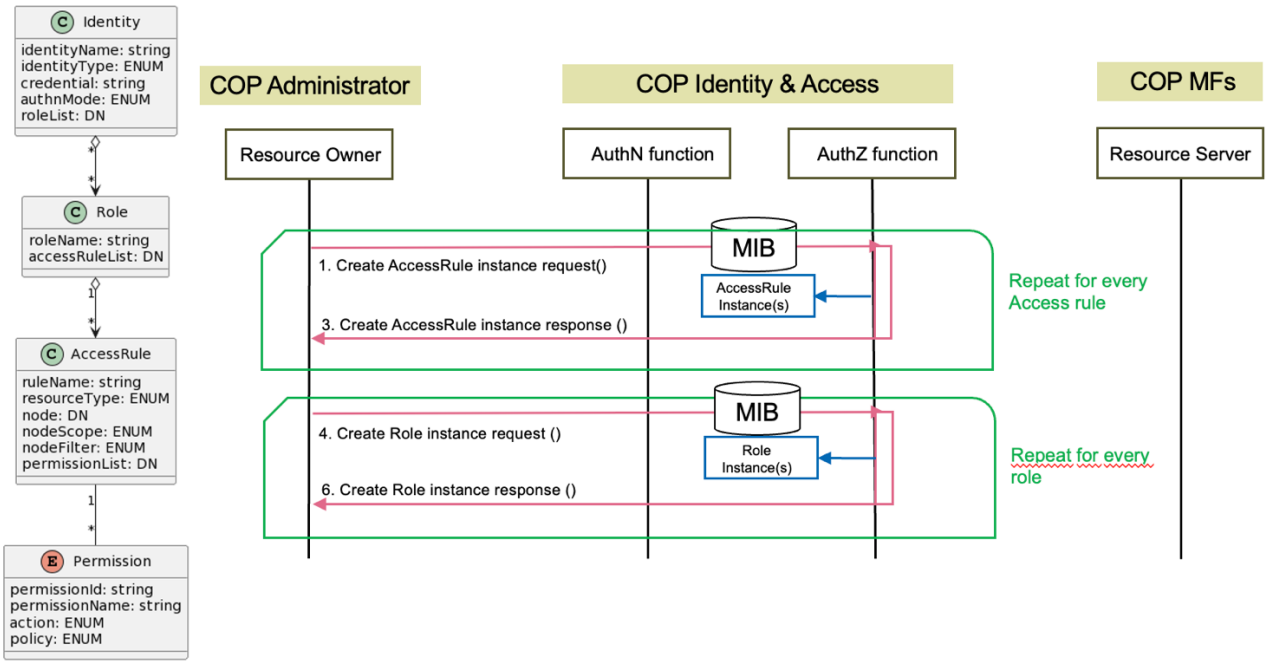


Figure 3-39: Design time – workflow.

In the onboarding stage, the DSP sends relevant tenant information (see Figure 3-40), so that the COP can create the "identity" class instance accordingly.

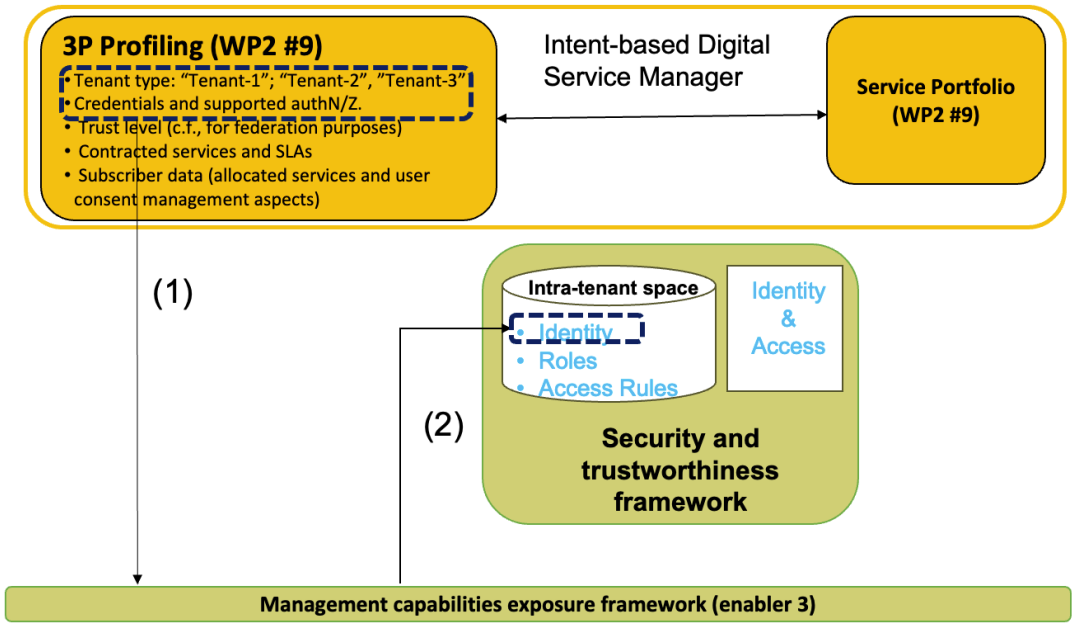


Figure 3-40: 3P onboarding – system view.

As detailed in Figure 3-41, this creation operation involves the following workflow:

- Request (step 1), whereby the COP takes the following information: i) tenant type; received from 3P profiling, it is mapped to the "identityType" attribute; ii) credentials and supported authN/Z; received from 3P profiling, it is mapped to the "credential" attribute; and iii) the collection of role instances which were created at the design time; fetched from the MIB, it is mapped to the "roleList" attribute.
- Record (step 2), whereby the COP creates the identity instance and stores it in the MIB. The Identity and Access component produces a unique identifier for that tenant (i.e. tenant-uuid value) and allocates it to the "identityName" attribute.
- Response (step 3), where the identity instance is notified to the requestor. The "identityName" attribute will be used by the tenant to identify itself from that moment.

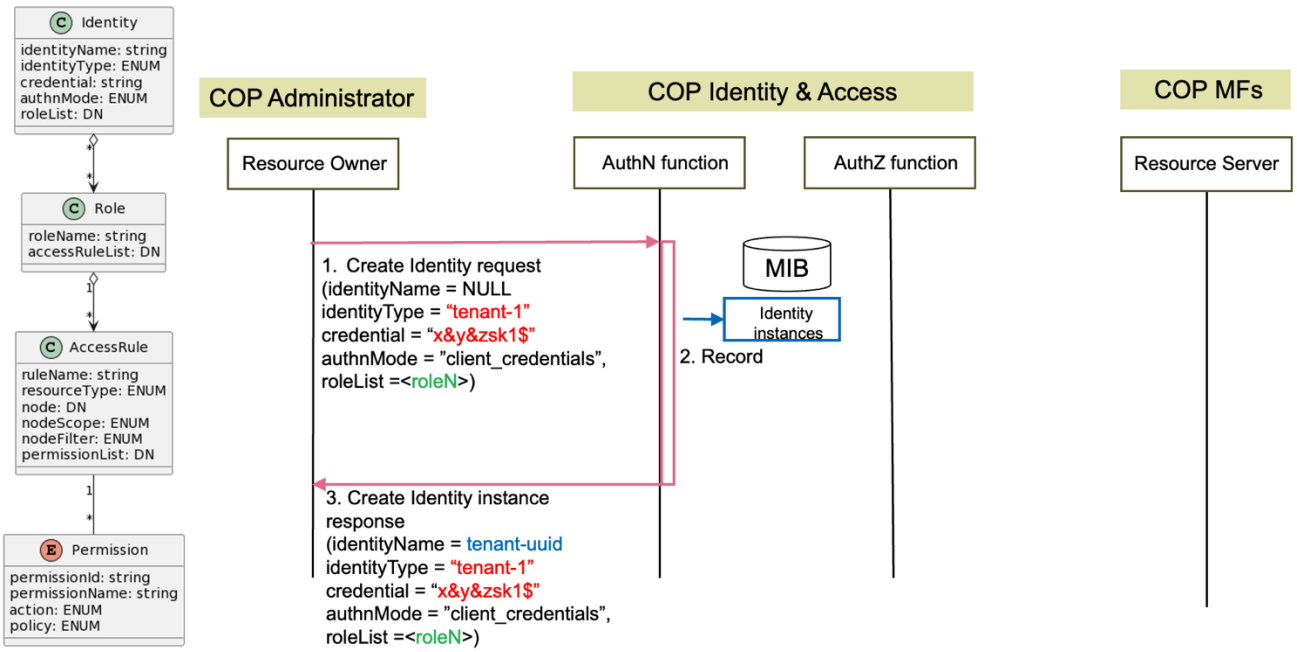


Figure 3-41: 3P onboarding – workflow.

From that moment on, the operation time starts, whereby the tenant can access capabilities offered by COP MFs via granular access control. The actual solution for this access control is highly specific to the protocol used for data model specification, either REST/Open API or Netconf/YANG. Table 3-3 summarizes the main differences between both options.

Table 3-3: AuthN/AuthZ – protocols comparison.

Aspect	Solutions	
	REST/OpenAPI	**Netconf/YANG**
AuthN and AuthZ functionalities	Co-located: both functions are integrated into the same server (AuthN/Z server)	Separated: Authentication takes place on the transport layer, and the authorization on the Netconf server
AuthN protocol	Client credentials assertion based authentication (with optional OIDC)	Mutual TLS (mTLS)
AuthZ protocol	Token based authorization framework with various grant modes, as specified in RFC 6749 [RFC6749] OAuth2.0.	Static authorization, as specified in RFC 8341 [RFC8341] Network Configuration Access Control Model (NACM).

Figure 3-42 illustrates the workflow for the operation time using REST/OpenAPI protocol.

In this option, the tenant first asks for authentication (step 1), issuing the "tenant-uuid" attribute. The COP AuthN/Z server identifies which "identity" instance is populated with this "tenant-uuid" attribute (step 2). With this information, the server generates an identity token (step 3), which is sent back to the tenant (step 4). The tenant then asks for authorization (step 5), issuing the received token. The server proceeds with token decryption (steps 6-7), in order to identify which roles and permissions have been allocated for that tenant (steps 7-8). Upon this identification, the server is informed about the tenant's management space, and proceeds with the generation of an access token (step 9), which is issued to the tenant (step 10). From this moment on, the tenant can gain access to the MF and invoke necessary capabilities (steps 13-14).

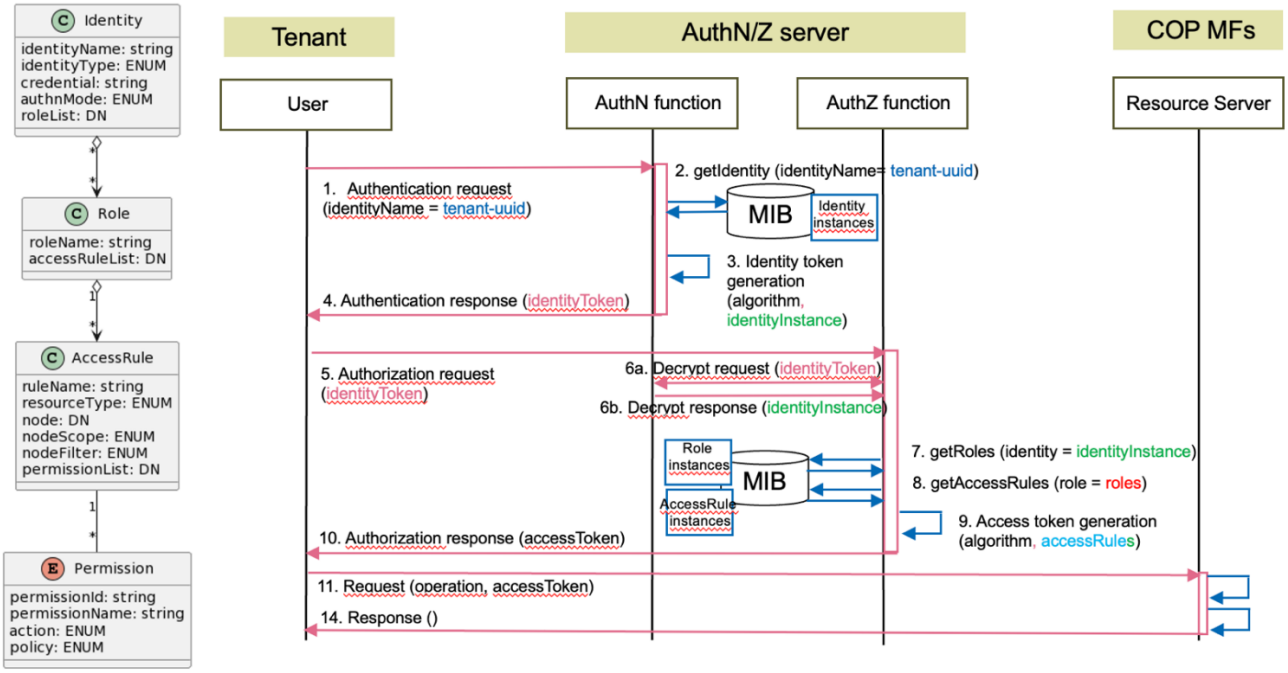


Figure 3-42: Operation time – REST/OpenAPI workflow.

Figure 3-43 illustrates the workflow for the operation time using Netconf/YANG protocol. The process is rather similar to the REST/OpenAPI solution, with three main differences: i) authentication occurs at the transport layer, using mTLS; ii) once authenticated, the tenant has direct visibility on COP MFs; and iii) the COP MFs delegates authorization in NACM server.

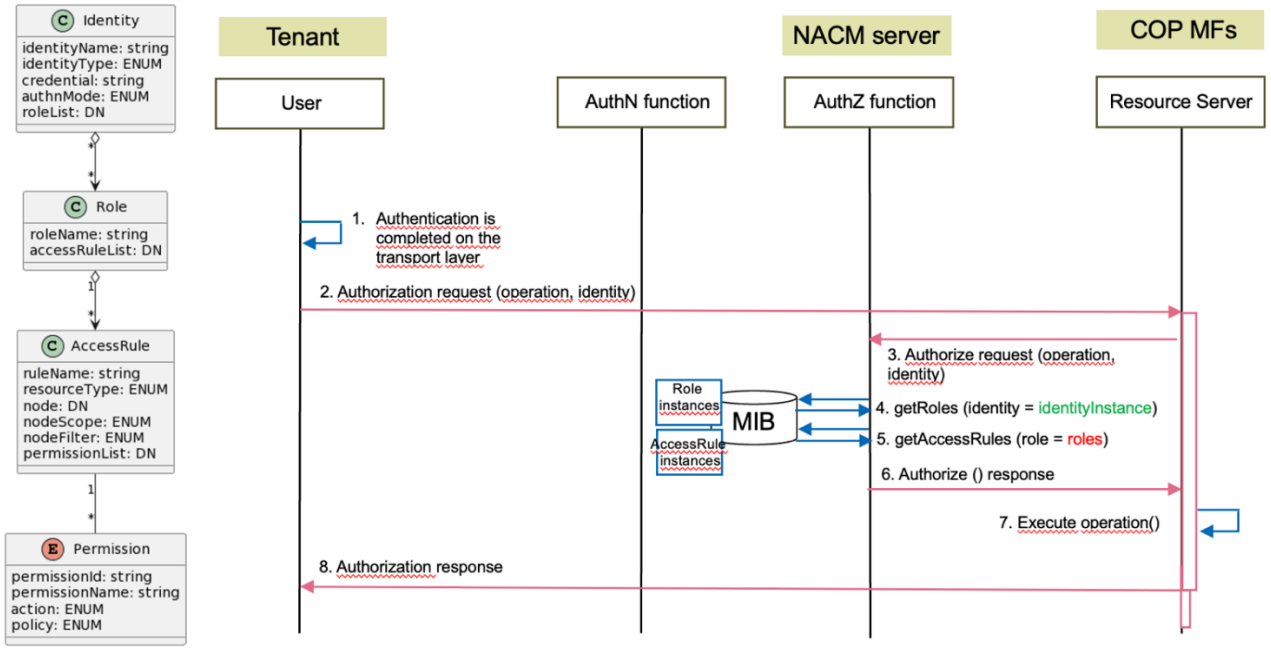


Figure 3-43: Operation time – Netconf/YANG workflow.

The interfaces used for this solution are APIs conveying YAML or YANG models, as illustrated in the workflows.

To ensure clarity regarding this enabler, it is essential to understand its inherent relation with enabler 3 and the specific role it plays in defining the information model that identifies a third party within a system. The identity onboarding process described here is complementary to the definition provided in the enabler 3 definition. The identity, roles, and access rules determine whether and how a third party can access the M&O resources that are exposed by the enabler 3. The security described in enabler 3 is solely related to internal security. It is

certain that the identity of the third party defined in this sub-enabler 4.1 must be translated into an entity that can interact with enabler 3 (thus enabling the establishment of mTLS with the Apache Kafka cluster). In conclusion, this enabler can be regarded as a layer that separates the enabler 3 and the resources it exposes from the external environment.

3.4.1.2 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this sub-enabler 4.1 are the following:

- **Availability**: by separating resource control for third-party entities, the system can prevent one tenant from monopolizing resources or causing disruptions that affect the availability of services for other tenants. This ensures that each tenant has fair access to resources and reduces the likelihood of downtime due to resource contention or malicious activities.
- **Reliability**: resource control separation helps maintain the reliability of the system by isolating the impact of failures or errors within individual tenant environments. If a particular tenant's application or activities lead to a failure, it can be contained within their allocated resources without affecting the reliability of other tenants' services. This isolation minimizes the propagation of failures and enhances the overall reliability of the system.
- **Security**: separating resource control for third-party entities enhances security by enforcing strict access controls and permissions. It allows administrators to define and enforce policies governing the interactions between tenants and system resources, mitigating the risk of unauthorized access, data breaches, or malicious activities. Additionally, by isolating each tenant's environment, the impact of security breaches or vulnerabilities can be contained, limiting their scope and protecting the integrity and confidentiality of other tenants' data and services.
- **Maintainability**: resource control separation simplifies system maintenance and management by providing clear boundaries between tenants' environments. This allows to apply updates, patches, and configurations more efficiently, without affecting the operations of other tenants. It also facilitates troubleshooting and debugging processes by isolating issues to specific tenant environments, making it easier to identify and resolve problems without disrupting overall system maintainability.

Regarding the KVIs, the most affected by this sub-enabler is **Trustworthiness**. Third-party resource control separation significantly impacts the trustworthiness of 6G networks by reinforcing confidentiality, integrity, availability, data privacy, operation resilience and security. It serves as a foundational element in building trust among stakeholders providing a resilient, secure, and trustworthy digital ecosystem.

3.4.2 Sub-enabler 4.2: User-centric service provisioning system

3.4.2.1 Sub-enabler design

In the context of 6G networks, the relevance of the user-centric service provisioning lies in its ability to enhance user experience, optimize network resources and support the diverse and dynamic requirements of future applications and services. This sub-enabler aims at provisioning tenant subscribers (e.g., enterprise users, end-users) with optimal and personalized Quality of Experience (QoE), according to their preferences, SLAs of subscribed services, and network context (e.g., context scenarios). This sub-enabler would provide the following capabilities:

- Definition of customized service policies for individual tenant users. These policies would be based on the UE Route Selection Policy (URSP) concept defined in 3GPP [23.501] and introduced in D6.2 [HEX223-D62]. These rules are interpreted and executed by mobile modems of user devices. The range of eligible devices not only includes OS-type devices (e.g., smartphones), but also other novel devices that Hexa-X-II is considering [HEX223-D52]. This capability is relevant for service fulfilment.
- Assisting users to gain access to subscribed services, while keeping their data safely stored, preventing any unauthorized entity to read/modify them. This would be done by injecting URSP rules into user devices. This capability is relevant for service activation.
- Ensuring the behaviour of tenant services comply with the KPIs (section 3.4.2.2) defined in the SLA, throughout the entire service lifetime. In case of violation, corrective actions will be taken to set

service back to the desired stated, leveraging closed control loop automation features. This capability is relevant for service assurance.

This enabler is strongly related to the concepts of intents, SLAs and Closed Loops (CL), which are correlated through a hierarchical relationship. This relationship can be split into two parts: intents to SLA and SLA to CL. With regards to the former, intents are translated into SLAs: the high-level goals defined by intents are broken down into specific, measurable requirements in SLAs. This translation involves understanding what performance metrics are necessary to achieve the high-level intent. Conversely, SLAs provide the benchmarks that CLs strive to maintain. CLs continuously monitor performance metrics and make adjustments in real time to ensure that the service meets the SLA criteria. For each UE, the relevant SLAs and associated CLs are stored and utilised to keep the KPIs within the defined parameters. It can be argued that the aforementioned flow from intents to SLAs to CLs allows for the systematic management of resources and service quality. This enables the rapid identification and rectification of any deviations from the defined performance thresholds, thus maintaining operational stability and performance consistency.

The contents presented in this and the following sections pertain to a conceptual design, with no provision for subsequent implementation.

3.4.2.1.1 System components

Figure 3-35 illustrated the framework eligible for enabler 4. In this section, we will be focusing on those components that are inherent to enabler 4.2, highlighted in red. These components are the following:

- **Intra-tenant space**: it specifies the management space provisioned to each registered tenant. This management space is the result of mapping information from the 3P profiling component (within the Intent-based Digital Service Manager, see [HEX223-D22]) into properties that the M&O layer can interpret and act upon. For user-centric service provisioning, the intra-tenant space captures the URSP rules eligible to individual tenant users. These URSP rules are injected into mobile user devices (service fulfilment), so that devices can grant access to user subscribed services (service activation).
- **Imported models**: this is a repository that stores the models for SLA (defined in Intent-based Digital Service Manager) and Closed Control Loop (defined in enabler 8). The mission of these models is to help provide zero-touch solutions for service assurance, using closed loop automation to supervise the compliance of tenant services to the signed SLAs. To that end, SLA attributes must be allowed as input for the goal of Closed Control Loop, so both models can work together.

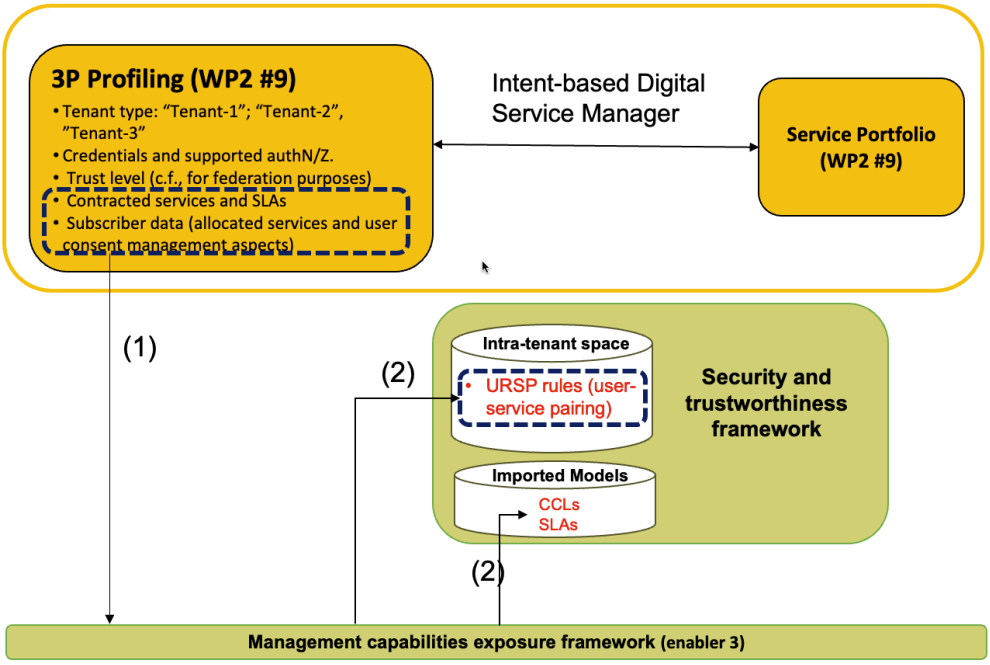


Figure 3-44: User-centric service provisioning – internal architecture.

Figure 3-44 shows the internal architecture of impacted components. As seen, the DSP sends the relevant information from "3P profiling" to the COP, which allocates it within the trustworthy 3P service provisioning framework. The relevant information for this sub-enabler consists of: i) contracted services, ii) SLAs, and iii) subscriber data. All contribute to the definition of URSP rules, which are stored within the "intra-tenant space". The SLA attributes are also captured according to the SLA model definition registered in the "imported model".

The key point in this solution for this sub-enabler is the use of URSP. URSP rules can be pre-configured on the device or be provisioned by the network to the device. In addition, a device may store a local configuration about the association of an application to a e.g. a PDU session (e.g., an operator provided S-NSSAI and DNN or application-specific parameters to set up a PDU Session). The format and contents of Local Configuration is left for specific device implementation. For further details on URSP structure and working, please refer to 3GPP TS 23.503 [23.503] and 3GPP TS 24.526 [24.526]. For a comparative analysis of different URSP rule options, see [HEX223-D62].

3.4.2.1.2 Workflows

As introduced in Section 3.4.2.1, this enabler provides three capabilities, each corresponding to a different stage: service fulfilment, service activation and service assurance. This section reports on the workflows for these stages.

Figure 3-45 shows the workflow for the service fulfilment. Firstly, the DSP sends the relevant information from "3P profiling component" (see Figure 3-44) to the COP, including the following data:

- Contracted services, among the service offerings available in the service portfolio. In case the contracted service includes application servers that are not managed by the DSP (i.e., 3rd party application servers deployed in Internet or in a local data network), the tenant shall specify the IP address ranges where these application servers are reachable. The tenant should also ensure these IP addresses are resolvable via existing DNS mechanisms.
- Subscriber data, needed for tenant users to become subscribers of contracted services. This information, received from the tenant, includes i) the identifiers of tenant users, e.g. MSISDN or IPv6; ii) which contracted service(s) are eligible for each device to be subscribed on; and iii) other related information as to privacy and consent management, as required by EU regulation.

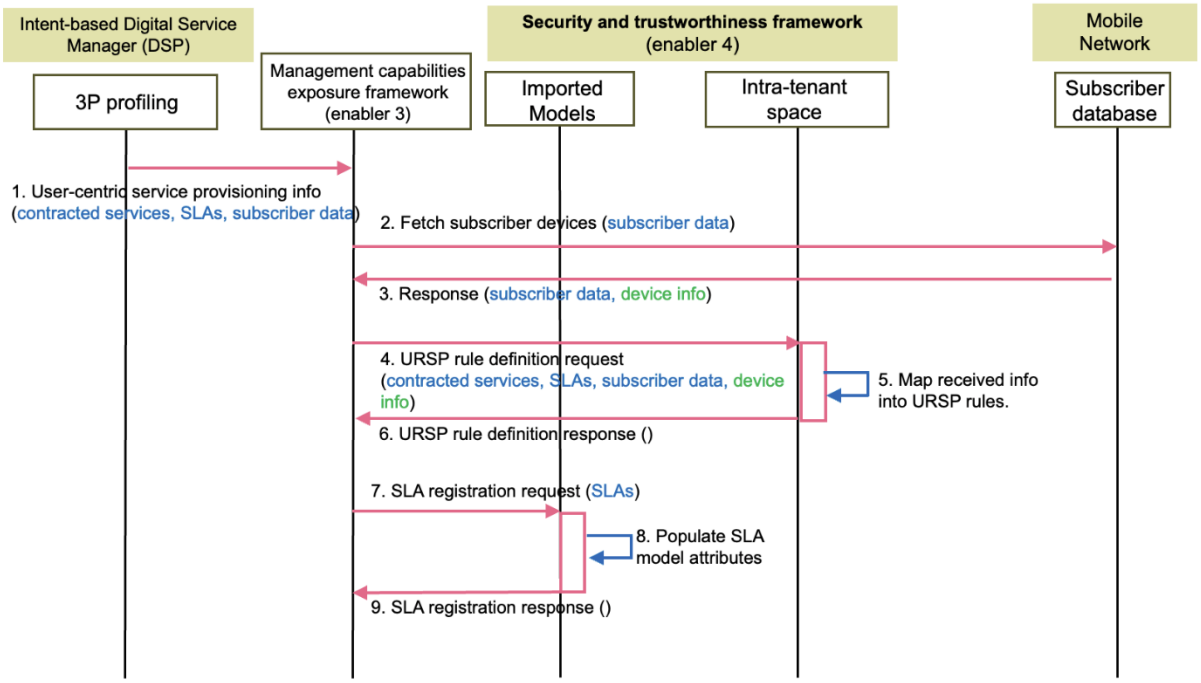


Figure 3-45: Service fulfilment workflow.

The Management Capabilities Exposure Framework receives data from the DSP and queries the mobile network to identify devices for tenant users. Based on the attributes received, it defines URSP rules, filling out

the rule attributes as defined in 3GPP TS 24.526 [24.526], and sends the information to the intra-tenant space for rule creation. Once this process is complete, it notifies the user and formats SLA values according to the SLA model. These values are then registered for later use, involving message exchanges between "imported models" and the Management Capabilities Exposure Framework.

Figure 3-46 shows the **service activation** stage, which requires the provisioning of URSP rules to the devices of tenant users, so that the client applications can issue service requests to the application server(s), so the connectivity between service endpoints can get established according to the registered SLAs.

First, the mobile network must know which URSP rules to inject on which devices. To that end, the integration fabric gets this information from the intra-tenant space (steps 1-2), and forwards it to the subscriber database (step 3), which in turn makes it available to the rest of control plane (CP) network functions from the core network and RAN (step 4). These network functions are responsible to inject applicable URSP rules into the individual devices, using the signalling interfaces defined in the mobile network for that purpose (step 5). From then on, the application client is enabled to issue service requests (step 6).

Upon receiving the application request, the modem (or other device component, e.g. the Operating System) evaluates the URSP rules to use in order of priority, and proceeds to select a rule as follows (step 7):

- If an URSP rule other than the default URSP rule matches the request, then the device selects this URSP rule as matching rule.
- If no matching URSP rule is found, then the device selects the default URSP rule as matching rule.

Finally, a matching URSP rule instructs the modem to start a PDU session establishment procedure (step 8). The modem uses the Route Selection Descriptor in the matching URSP rule to determine PDU Session connectivity parameters such as S-NSSAI (slice identifier), DNN (data network through which the application server is reachable), etc. These parameters enable the device to determine if the user traffic data can be routed through an already established PDU Session or if there is a need to trigger the establishment of a new PDU Session. Upon completion of step 8, the service gets activated, and traffic between service endpoints (application client and server) starts flowing.

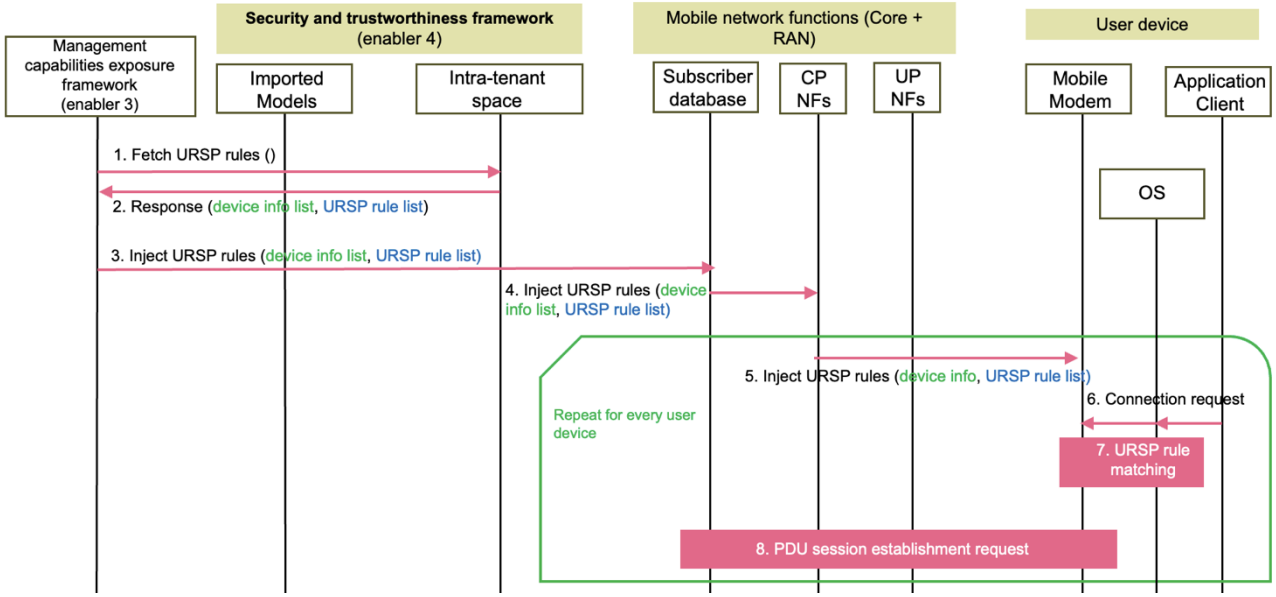


Figure 3-46: Service activation – workflow.

In the service assurance stage (Figure 3-47), the mission is to supervise that the state of the service instance is conformant to the SLA, and take corrective actions otherwise. To that end, closed loop control (CL) instances will be used. In this regard, it is needed to insert SLA attributes into the CL model, such that the SLA becomes the goal that the CL instance must fulfil and assure. The “imported models” component oversees this (step 2), and informs the management capabilities exposure framework accordingly, specifying the CL class attributes associated to each service (step 3). With this information, the integration fabric reaches out to the CL Governance (step 4), requesting the creation of an CL class instance for every contracted service. The CL Governance completes the provisioning (step 5-6).

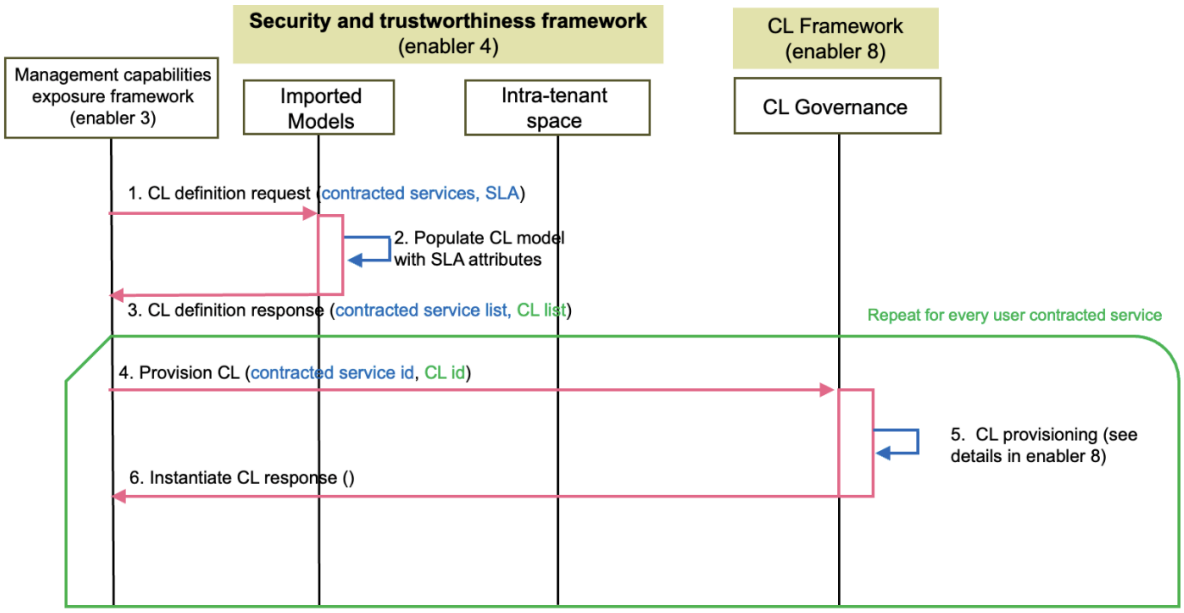


Figure 3-47: Service assurance – workflow.

3.4.2.2 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this sub-enabler 4.2 are the following:

- **Availability**: it measures the proportion of time a service is fully operational and accessible to users. High availability ensures that the network services are always accessible, minimizing downtime and ensuring that subscribers can rely on continuous service performance.
- **Reliability**: it assesses the ability of the network to perform its required functions under stated conditions for a specified period of time. Reliability includes the network's capacity to handle errors, recover from failures, and maintain service continuity in the face of challenges such as hardware malfunctions or high traffic.
- **Maintainability**: this involves the ease with which a system can be maintained in order to correct defects or their causes, improve performance or other attributes, or adapt to a changed environment. In network services, maintainability can refer to the ease of updating the system, fixing issues, and scaling services to meet changing demands.

Additionally, user-centric service provisioning may include other KPIs similar to those found in existing service templates such as the Generic Network Slice Template (GST) [NG116]. These could include:

- **Guaranteed device throughput**: this KPI measures the minimum data transfer rate that must be maintained for each device connected to the network, ensuring efficient and consistent performance.
- **Maximum number of PDU sessions**: it represents the upper limit of simultaneous packet data unit sessions that can be handled by the network, ensuring that the network can support a high volume of data traffic without degradation in service quality.
- **Maximum number of registered subscribers**: this is the maximum number of subscribers that can be registered and supported by the network at any given time, indicating the network's capacity.
- **Maximum latency**: this KPI specifies the maximum allowable delay in data transmission, ensuring timely communication which is critical for applications requiring real-time interaction, such as VoIP or gaming.
- **Error rates**: measures the frequency of errors during data transmission, which can affect the quality and reliability of network services.

These KPIs are essential for monitoring the performance and health of user-centric services, allowing providers to guarantee compliance with the terms specified in SLAs and ensuring a high-quality user experience.

Regarding the KVIs, the most affected by this sub-enabler is Trustworthiness. User-centric service provisioning significantly enhances the trustworthiness of future 6G networks by focusing on the specific needs and expectations of users regarding security, privacy, and reliability.

3.4.3 Sub-enabler 4.3: Trust Management System

3.4.3.1 Sub-enabler design

6G networks are expected to support massive connectivity, ultra-reliability, and high mobility for IoT devices, which are typically situated within the extreme-edge domain. Within this domain, a myriad of devices with different characteristics continuously communicate and exchange information. However, they often do so over networks that are unreliable and unstable. To address this challenge, there is a pressing need for a flexible, lightweight, and adaptive access control mechanism. Such a mechanism is crucial for ensuring secure communications among trusted devices, particularly in the context of 6G's architectural framework.

Trust management system sub-enabler proposes Trust Evaluation Functions (TEFs) for assessing the trustworthiness level of entities like infrastructure components, compute nodes as well as services, applications, and 3rd party consumers. Also, this sub-enabler is planned to establish trustworthy communication paths holding sensitive data workloads as they transit a network, for data transferred between subnets, or connecting different elements that belong to different trust domains.

TEFs estimate the trustworthiness level of the system of interest depending on the use case. In the case of workload placement/orchestration, a TEF focusing on the infrastructure layer is used for assessing the trustworthiness level to the infrastructure and compute nodes. In the case of flexible topologies, a TEF of flexible networks and flexible nodes is utilised. In the case of service-centric orchestration, a TEF focusing on the service layer is utilised, and so on. The TEF of the infrastructure layer is used by cloud orchestration engines to allocate workloads to compute nodes with the goal of achieving maximum trustworthiness. The TEF takes input from the monitoring telemetry data (enabler 2) of the various components of the system and triggers one of the TEFs depending on the intent request. The TEF for infrastructure layer analyses the data coming from the compute nodes and the computational workloads placed on them and outputs the trust indexes of the compute nodes of the infrastructure of interest based on the novel mathematical formula described in the following subsection. The output is fed to the functionality allocation component which along with other metrics and data suggests a close-to optimal placement of the computational workloads to the available compute nodes. This suggestion is analysed by the orchestrator which would be the responsible for enforcing the needed decision. A schematical representation of the high-level architecture of TEFs is show in Figure 3-48.

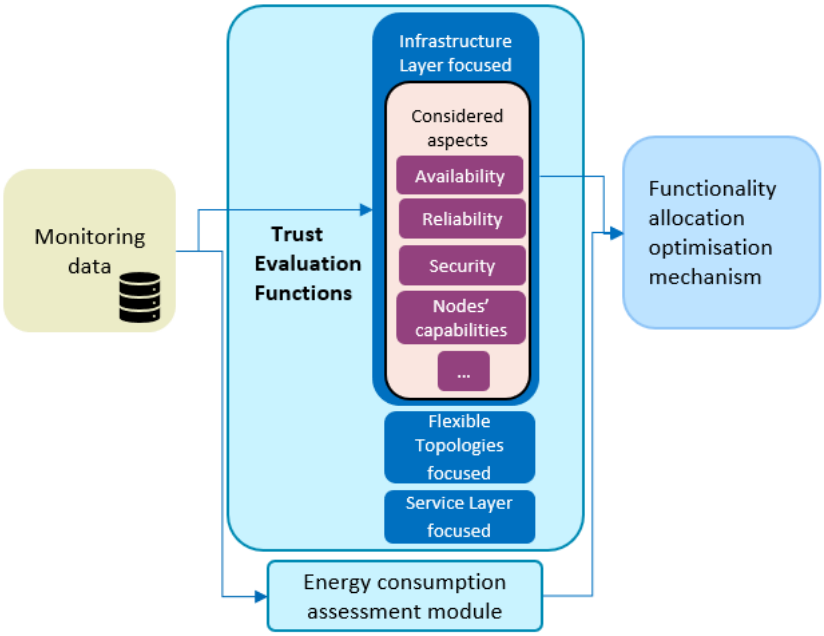


Figure 3-48: High-level architecture of trust evaluation functions.

3.4.3.2 Preliminary implementation and early validation results

The early stage implementation outlined in this section includes a detailed description of how the trust indexes of the compute and infrastructure nodes were calculated. These indexes are used by cloud orchestration engines to allocate workloads to compute nodes targeting maximum trustworthiness. In particular, the engine used in this implementation is the functionality allocation mechanism described in section 3.6.1.2.3. The functionality allocation mechanism is a metaheuristic optimisation algorithm based on the genetic algorithm paradigm. It optimally places the computational_workloads to the available compute nodes, towards maximum energy efficiency and trustworthiness by minimising the objective function consisting of an energy consumption term, an E2E latency term and a trustworthiness related term. The trust indexes are estimated by the TEF python algorithm which is described below. The described TEF is integrated to the PoC#A and PoC#B of the project, but these results are described in the respective deliverables of E2E system PoCs' evaluations [HEX223-D21, HEX223-D22]. Here, some preliminary simulation results are presented considering fifty compute nodes with various capabilities and increasing number of workloads with various requirements.

As described in Hexa-X [HEX23-D14], trustworthiness has a wide scope and is related to security, privacy [HEX23-D13] as well as availability and reliability which fall under the dependability framework [HEX21-D71]. Based on this, the trust evaluation function envisioned for infrastructure layer, for now, considers the following aspects:

- Availability, A_j, which is initially approached as the percentage of a time window during which the compute node was not fully loaded or unavailable.
- Reliability, R_j, which is approached as the times that the compute node succeeded to execute a task/workload within a time threshold.
- Security, S_j, which is related to secure communication and trusted computing but for this preliminary implementation it is envisioned as a binary variable indicating if Transport Layer Security (TLS) is available or not.
- Multi-connectivity capabilities, M_j, which spans between 0 to 1 depending on which capabilities (e.g., 4G/5G, Wi-Fi, NB-IoT, BT, interfaces - device/UE-based adaptive RAT selection) are supported.
- Battery level, B_j, of the battery powered devices/nodes (e.g., robotic units).

Hence, the trust index $trustN_j$ of the compute node j is calculated based on the following formula:

$$trustN_j = w_1 A_j + w_2 R_j + w_3 S_j + w_4 M_j + w_5 B_j$$

The weights, w_1, w_2, w_3, w_4, w_5 vary depending on the use case and the intent request.

Figure 3-49 shows the preliminary results obtained related to trustworthiness which were calculated with the trustworthiness formula described before. In particular, the graph on the figure shows the percentage increase of trustworthiness obtained by using the metaheuristic functionality allocation mechanism (see section 3.6.1.2.3) compared to the feasible round-robin placement algorithm (baseline). The functionality allocation optimisation mechanism estimates the close to optimal placement of the computational workloads to the available compute nodes by minimising the weighted (a_1, a_2, a_3) objective function, $\min_{x,z}(a_1 E + a_2 L - a_3 T)$, which consists of the energy consumption term E, the E2E latency term L, and the trustworthiness term T. The trustworthiness term is the sum of the trust indexes of the compute nodes utilised for the workload placement. These measurements were taken for different trust weight levels a_3, low, medium and high. These levels correspond to an almost zero contribution, a moderate contribution and full contribution, respectively, compared to the other terms of the objective function. Fifty fixed virtualised compute nodes were used, with various capabilities and an increasing number of compute workloads with varied requirements. In particular, the compute nodes used had {1000, 1200, 1500, 2000, 2500, 2600, 8800} MIPs levels of available CPU, {2048, 4096, 8192} MB levels of available memory, {160, 360, 460} W levels of power consumption when fully loaded and {70, 100, 170} W when idle. The links between nodes have capacity $3.3 - 20$ Mbps. The workloads have {250, 300, 500} MIPS levels of required CPU, {256, 512} MB levels of required memory, {10, 20} MB levels of data transferred. The weights of the trust index formula were all set to 0.2 $(w_1, w_2, w_3, w_4, w_5)$. In these measurements/experiments the initial population of genetic algorithm's chromosomes was 100, the crossover rate was 0.8 and the mutation rate was 0.008-0.07.

As shown in the graph, the higher the weight a_3 of the trust term of the functionality allocation mechanism used, the higher the percentage of increase of the trustworthiness was obtained which was as expected. Also, the more workloads we need to place, the percentage increase of trustworthiness decreases. This is because when many workloads need to be placed, the number of feasible placement solutions decreases and the potential gains in the percentage increase of trustworthiness become limited. In general, in this scenario, the metaheuristic functionality allocation algorithm, compared to the baseline (round-robin placement), can gain up to 43% increase of trustworthiness. However, beyond these measurements, a significant benefit of this enabler is its applicability to integrate the extreme-edge domain, where devices are highly volatile. Therefore, employing a TEF to assess the trustworthiness of devices is crucial for deploying network service components within this domain. Of course, one important challenge that is under investigation is the actual utilisation of this enabler to the possible huge extreme-edge domain (cloud native) where thousands or millions of devices are to be assessed.

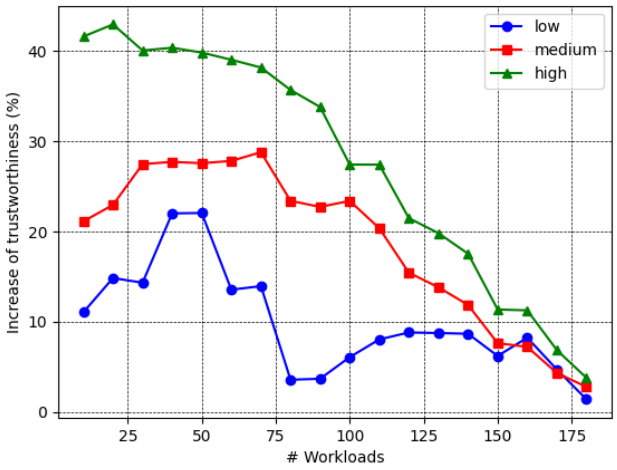


Figure 3-49: Trustworthiness measurements of the metaheuristic functionality allocation mechanism.

3.4.3.3 Impacted KPIs and KVIs

The main KPIs influenced by the adoption of this sub-enabler 4.3 are the following:

- **Reliability:** This KPI is directly impacted by sub-enabler 4.3 as it plays a critical role in the trust evaluation functions, which are essential for accessing trust indexes.
- **Availability:** The implementation of sub-enabler 4.3 has a direct effect on availability, again due to its involvement in the trust evaluation functions.
- **Scalability:** Sub-enabler 4.3 impacts scalability through its close communication with enablers 5 and 6, which collectively contribute to the optimisation of resource management and functionality allocation.

Regarding the KVIs, **trustworthiness** is the most significantly impacted due to the influence of sub-enabler 4.3.

3.5 Enabler 5: Synergetic orchestration mechanisms for the computing continuum

The enabler 5 focuses on the development of synergetic orchestration mechanisms for the management of network services over programmable resources in the computing continuum. With the term computing continuum, we refer to resources that span from the IoT to the edge to the cloud part of the infrastructure. The development of distributed services that are deployed over resources in the continuum, while having strict QoS requirements in the edge (or extreme edge) part is arising in the 6G ecosystem, taking advantage of emerging technologies such as integrated communication and sensing. The enabler aims to develop different orchestration techniques that can be applicable for distributed, de-centralized and/or federated management of the available resources. Depending on the orchestration needs and the involved stakeholders, the selection of the most appropriate technique can take place. Peculiarities based on the need to manage resources that span

across the computing continuum (IoT devices, extreme edge devices, edge/cloud infrastructure) are considered. This enabler is composed of 3 sub-enablers, referring to the development of multi-agent synergetic orchestration mechanisms that mostly follow a hierarchical approach, decentralized orchestration mechanisms that build upon decentralized intelligence, and federated orchestration mechanisms that consider the interaction between multiple orchestrators (e.g., by considering operation under different administrative domains). In the following, details for each sub-enabler are provided.

3.5.1 Sub-enabler 5.1: Multi-agent systems for multi-cluster orchestration

3.5.1.1 Sub-enablers design

Sub-enabler 5.1 contributes towards the development of orchestration mechanisms for the lifecycle management of network services and applications across resources in the computing continuum. Two main aspects are considered; the need to address management of heterogeneous resources that are made available across distributed clusters in the continuum, and the need to increase the intelligence and automation characteristics of the provided orchestration mechanisms considering both the deployment and operational/runtime phases of network services and applications. With the term cluster we refer to a set of computational and network resources that are managed through a unique interface. Such resources may belong to any part of the computing continuum. When the resources are spanning across multiple clusters, we refer to multi-cluster management of resources.

For the first aspect (multi-cluster management of resources), focus is given on the proper modelling of the resources and the development of distributed resource management mechanisms. The supported functions include -among others- the registration of resources made available from multiple clusters, the abstraction of such resources and their management from centralized interfaces (e.g., a unique interface that is able to manage resources in the different clusters), the management of the connectivity, security and the QoS assurance for the interconnection among the clusters. For the second aspect (intelligence and automation in orchestration mechanisms), the term synergy is introduced for the development of synergetic (or collaborative) orchestration mechanisms.

Various levels of synergy are considered. Synergies may be applied through the adoption of multi-agent systems that collaborate to achieve a joint objective. Multi-agent systems are going to be used for the development of orchestration mechanisms. Such mechanisms can support various actions, including the autoscaling of functions/microservices of a specific application/service graph, deployment and compute offloading techniques with specific optimization objectives, and migration mechanisms. Synergies may be also applied between different stakeholders in a 6G ecosystem. For instance, interaction between network providers and over the top (OTT) players such as edge/cloud application providers could be applied. In this case, the specification and development of open APIs is envisaged, mainly northbound interfaces offered by network providers to OTT players.

3.5.1.1.1 System components

In Figure 3-50, a high-level view of the main components considered in this sub-enabler is depicted. Sub-enabler 5.1 includes a main orchestration entity (Synergetic Orchestrator) that supports a set of mechanisms for managing the deployment of distributed service/application graphs over resources in the computing continuum. Synergies among multiple agents are applied, where each agent is responsible for decision making in a local or global administrative area. The provided functionalities include the initial deployment of network services/applications across multi-cluster infrastructure, as well as a set of optimal resource management mechanisms during runtime (e.g., scaling, compute offloading, migration). To design such mechanisms, proper resources abstractions have to be provided into a set of information models, while continuous data consumption from the available monitoring probes and observability stacks (as provided by Enabler 2) is taking place. A top-down execution plan is considered where higher level entities manage lower-level entities. The Synergetic Orchestrator interacts with the Multi-cluster Manager and the Network Manager. The Multi-cluster Manager is responsible for the management of compute resources across the continuum (e.g., extreme edge, edge, cloud clusters). The Network Manager is responsible for the management of network resources across the computing continuum and could take advantage of the mechanisms provided through Enabler 1. Among others, network service mesh mechanisms can be applied for managing the connectivity across the network service/application components deployed in the different clusters.

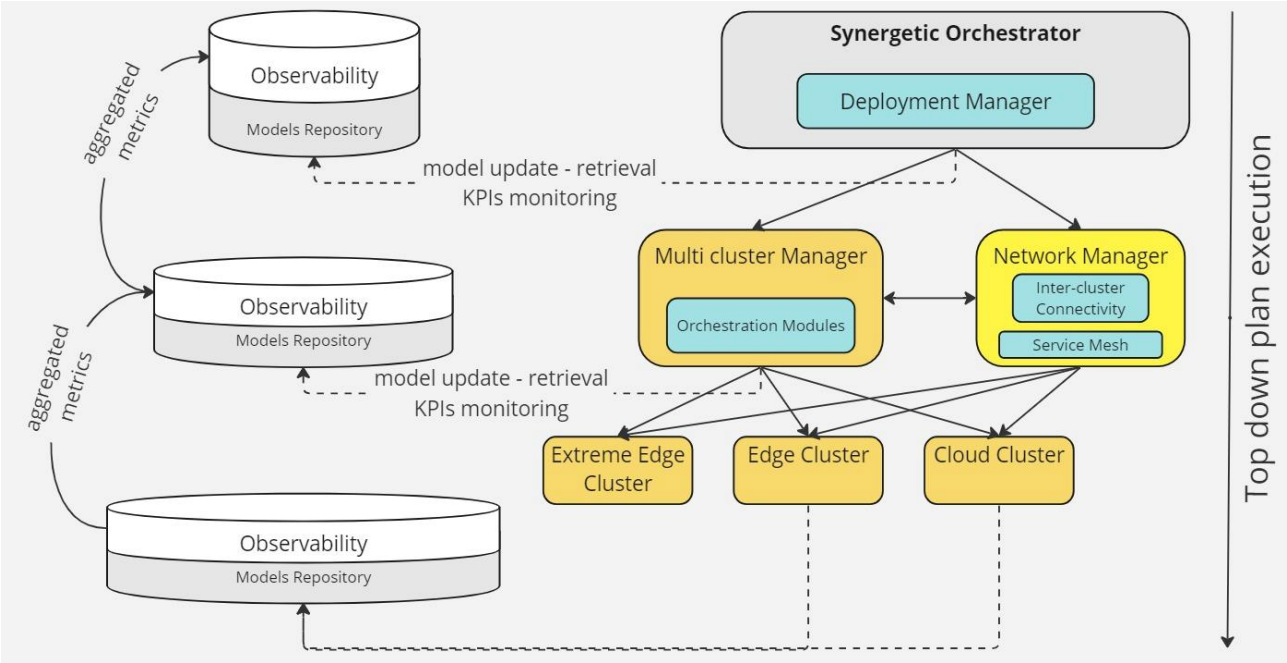


Figure 3-50: High-level view of multi-cluster orchestration components for the computing continuum.

3.5.1.1.2 Description of the system components

3.5.1.1.2.1 Synergetic Orchestrator

Orchestration in the computing continuum will become very complex with the introduction of multiple domains, multiple objectives and multiple agents managing different resources. For mitigating this complexity, a multi-agent management mechanism is proposed, decoupling the agents' orchestration functionality from the agents' deployment. In this way, the management of heterogeneous agents, representing different entities or stakeholders, with different inputs, outputs and algorithms can be deployed and coordinate to achieve their corresponding Service Level Objectives (SLOs) and the users' high-level intents.

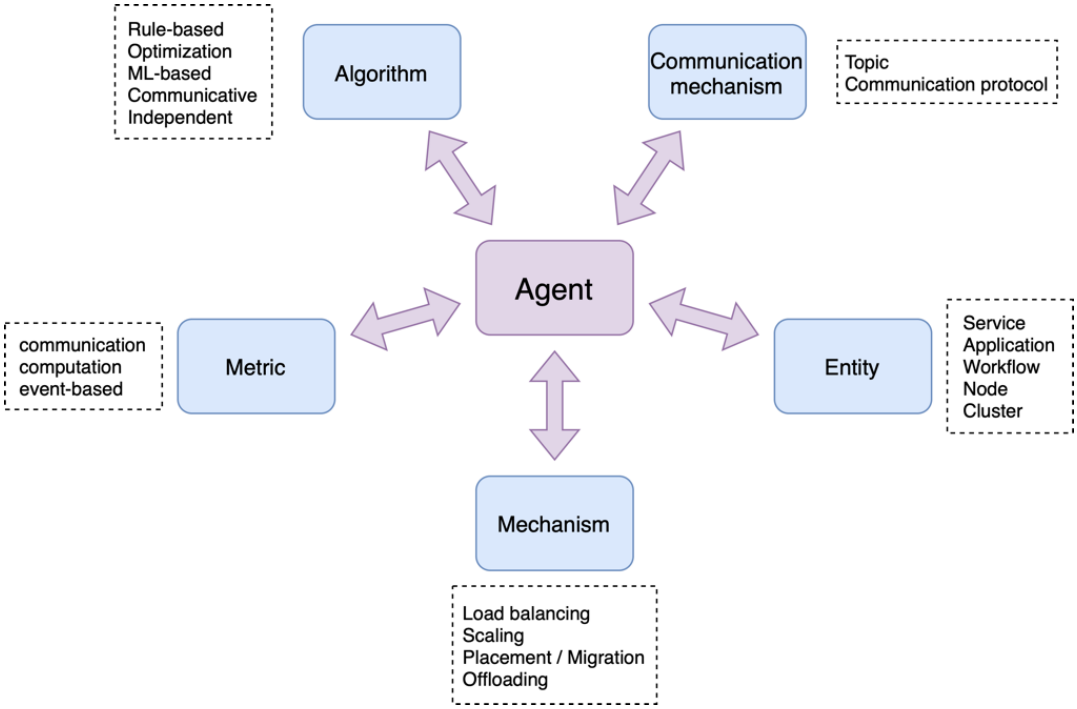


Figure 3-51: Multi-agent interactions.

As shown in Figure 3-51, the deployed agents can be associated with specific entities such as services, resources or workflows and thus, their optimization criteria differ and are based on different metrics. The underlying algorithms can be rule-based, optimization or ML-based, while the need for inter-agent communication may be necessary. All these characteristics, are considered and managed by the Synergetic Orchestrator to allow modular functionality for different scenarios and different stakeholders.

Deployment Manager

The lifecycle management of network services and applications is provided by the Deployment Manager (see Figure 3-50). The main functionalities of the Deployment Manager (functionality allocation, service autoscaling, and computation offloading and migration) are described in the following, while initial implementation of part of them is detailed in Section 3.5.1.2.

Functionality allocation, responsible for the optimal placement of the computational workloads to the available compute nodes of the infrastructure, including edge/cloud servers, extreme edge nodes, towards energy efficiency and maximum trustworthiness. A more detailed view of this mechanism with the algorithms' description focusing on energy efficiency is presented in Section 3.6.1.2.3 (Enabler 6) and the trustworthiness aspects are analysed in Section 3.4.3 (Sub-enabler 4.3). This section describes the functionality allocation algorithm in the orchestration process, focusing on the communications of the algorithm with the necessary system components to ensure successful deployment of network services and applications in the continuum. The algorithm takes as input the computational workloads with their computation and physical requirements, the data size and the location of data generation corresponding to each workload. It also considers the available compute nodes with their capabilities, and the network topology graph. Output of the algorithm is the near optimal placement of computational workloads to the available compute nodes. The initial development and the description of the communications among the various components utilised for this implementation are described in Section 3.5.1.2.3.

Autoscaling functionalities. With the term autoscaling, we refer to the automated horizontal scaling of resources, where the resources refer to specific components of a distributed service or application (e.g., scaling of pods in a Kubernetes system). Two different approaches are studied for autoscaling services in the continuum, a centralized Reinforcement Learning (RL)-driven autoscaling scheme and a decentralized Multi-Agent RL (MARL) one for enabling the distribution of the agents in the continuum. Both those flavours are based on the agents learning how scaling service replicas influence the objectives and identify the optimal combinations according to the implementation (for details regarding the mechanism's initial implementation see section 3.5.1.2.2). The autoscaling mechanisms work in coordination with a scheduling optimizer that supports the placement of services across the multi-cluster environment (see Figure 3-52).

The autoscaling mechanism is dependent on the multi-cluster setup where the application is deployed. Thus, a crucial aspect for the end-to-end implementation of the underlying techniques, depends on the existence of the RL environment, an interface responsible for accessing application metrics, but also for applying orchestration actions instructed by the mechanism's intelligence. The RL agent interacts with the environment on two occasions, to obtain the state information from the multi-cluster monitoring tools (e.g. for metrics such as application latency, CPU, memory, request rate, throughput etc.) and to scale up a service (agent actions). The environment is responsible to calculate the reward and the next state of the agent, based on the metrics obtained. The core of the mechanism leverages the learning power of Deep Reinforcement Learning (DRL) for navigating in an unknown environment by trial-and-error. During training the naïve agent explores the environment by applying different scaling actions and by observing the metrics and calculating the reward and the next state the agent's network is trained to produce actions with optimal state-action values.

Service scaling depends on the incoming workload, which can be considered either reactively or proactively. The former approach is based exclusively on the system state and fails to proactively facilitate decision making for repeating workload patterns. The latter relies on the introduction of the Workload Estimator, a forecasting component which can be implemented using both analytical/mathematical techniques and modern machine learning techniques. Analytical techniques refer mostly to regression or autoregression models such as Auto-Regressive Integrated Moving Average (ARIMA), while ML models have been proposed for predicting future time-series data like Support Vector Regression (SVR), Time-Delay Neural Networks and more recently Recurrent Neural Networks (RNNs), (Long Short-Term Memory (LSTM) networks or Transformers [STS18] [ZHA20]. Apart from the different algorithms that can be used for the forecasting task, a complete

methodology is necessary on the whole workflow of data collection, analysis and future values prediction stages and their integration in the orchestration workflow loops. The resulting component is utilized by the autoscaling mechanism to provide the predicted workload values as input and enable the mechanism's proactive capabilities.

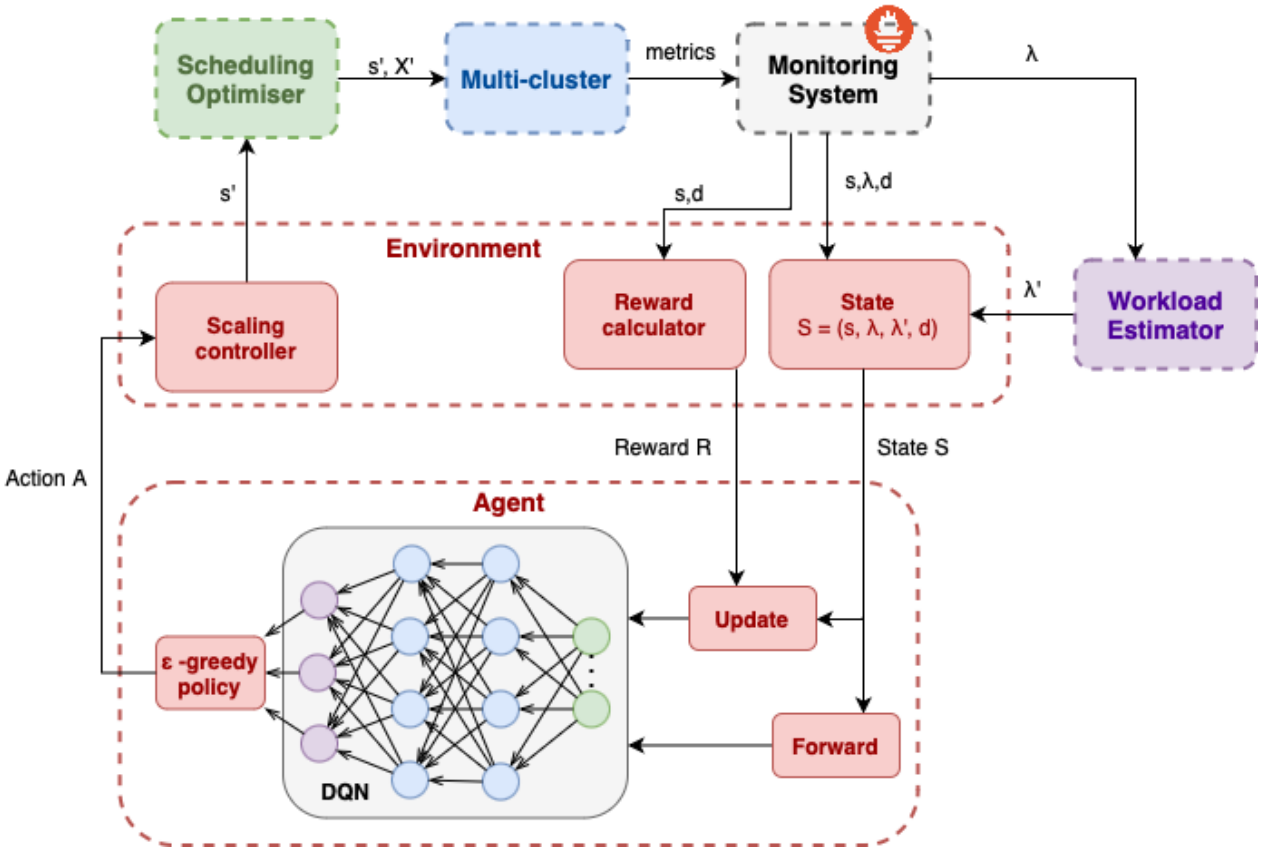


Figure 3-52: RL-driven autoscaling mechanisms for multi-cluster deployments.

In order to tackle higher-scale application graphs where massive applications need to be scaled up, a decentralized approach is suitable to efficiently handle large state and action spaces. This approach considers an agent for each service deployed in the continuum and allows decentralized decision making. While generally, decentralized processing misses the benefits of centralized information gathering, the mechanism considers global feedback during training and allows decentralized actions during execution. In this way, the advantages of global information will guide the agents towards globally optimal decisions, while allowing for responsibility distribution across multiple agents. This approach relies on value function factorization techniques [RSS+18], while for cases where communication is constrained other decentralized techniques [FAF+16] are to be trialled. More details on this approach are given in section 3.6.1.2.6.

Computation offloading and migration. The scheduling optimizer referenced in Figure 3-52 mainly tackles the problem of identifying the suitable place for a service to be deployed. In specific, for multi-cluster deployment two layers are considered, an Edge layer close to the users and a Cloud layer holding more computing power further away geographically. For large application graphs including multiple overlapping service chains this is a computationally intensive problem and a solution can be provided either independently from the autoscaling decision or jointly. The mechanism takes into consideration variables such as the application structure and the network topology to provide an initial deployment plan. At runtime, this information combined with variables such as the main objective to be optimised (e.g. end-to-end latency, power consumption etc.), system state variables such as CPU, RAM or bandwidth utilization and the corresponding upper limits, or even application specific metrics such as execution and queuing times, request success rate etc., can be considered for optimizing real-time performance by migrating services closer to the users or for offloading intensive computations to machines with sufficient resources. An initial implementation jointly with the autoscaling mechanism is provided in section 3.5.1.2.2.

3.5.1.1.2.2 Multi-cluster resource manager

Organizations face several challenges when managing resources across multiple clusters. To enable efficient and effective operations, the multi-cluster resource manager (see Figure 3-50) provides various functionalities that help streamline operations across diverse clusters and ensure optimal utilization of resources. These functionalities include:

- Resource Allocation and Scheduling: efficiently scheduling and allocating resources to meet the diverse requirements of different applications and workloads across clusters (appendix 8.4.1 contains a related approach for edge computing convergence).
- Networking Management: facilitating seamless communication between services and applications distributed across different clusters; addressing latency and bandwidth constraints to enable smooth interaction between geographically dispersed clusters.
- Fault Tolerance and Recovery: building robust systems capable of withstanding failures within individual clusters without compromising the overall operation of the multi-cluster environment; implementing effective recovery mechanisms to swiftly restore operations after failures occur.
- Configuration and Policy Management: ensuring consistency in configurations across clusters to maintain uniformity and reduce the likelihood of errors; enforcing organizational policies and compliance standards across multiple clusters.
- Scalability: addressing the challenges associated with scaling clusters up or down to accommodate evolving demands; ensuring that resource management systems are scalable to cope with increasing requirements.
- Massive Device Connectivity: managing resources efficiently to accommodate the proliferation of connected devices that demand computing across clusters; prioritizing low latency and high throughput to maintain optimal performance amidst massive device connectivity.

By encompassing these functionalities, the multi-cluster resource manager enables organizations to easily navigate the complexities of distributed environments, fostering operational efficiency and agility.

3.5.1.1.2.3 Network Manager

Most of the functionalities supported by the Network Manager (see Figure 3-50) are already detailed in Enabler 1. However, in this section some extra mechanisms are introduced that can be assistive to the Multi-cluster Manager and support network connectivity and management functionalities. Focus is given on Network Service Mesh (NSM) [NET24] that is a concept that aims to enhance the flexibility and programmability of networking. NSM enables dynamic and on-the-fly composition of network services to meet the requirements of specific applications or workloads. In the context of 6G, which is expected to bring about advancements in terms of data rates, latency, and network architecture, NSM can play a crucial role in achieving a more adaptable and efficient network. Considering that cloud-native functionalities are becoming dominant in the development of microservices-based network services and applications, the adoption of network service mesh techniques for the support of control plane functions are highly considered. Service Meshes can decouple aspects related to network connectivity from the management of the microservices that are combined in the form of cloud native functions and application graphs. Functions such as network connectivity, service discovery, load balancing and telemetry can be provided through service meshes. Service meshes are usually based on the specification of open APIs to provide and consume control plane services (e.g., connectivity, security, observability). Service meshes can be adopted and deployed in parallel with compute orchestration mechanisms, enabling inter-cluster connectivity and traffic management without the need to engage complex SDN management solutions.

A service mesh is a software infrastructure layer for controlling communication between services and it is generally made of two components: the data plane, which handles communications of the application and is usually deployed with the application as a set of network proxies, and the control plane, which is the brain of the mesh that coordinates the behaviour of proxies and provides APIs to push configurations, ensure service discovery and observe the entire network. One of the most prominent solutions that are considered to be used for the provision of services through service meshes regards the Network Service Mesh [NET24]. Network Service Mesh allows individual workloads, wherever they are running to connect securely to Network Service(s) that are independent of where they run.

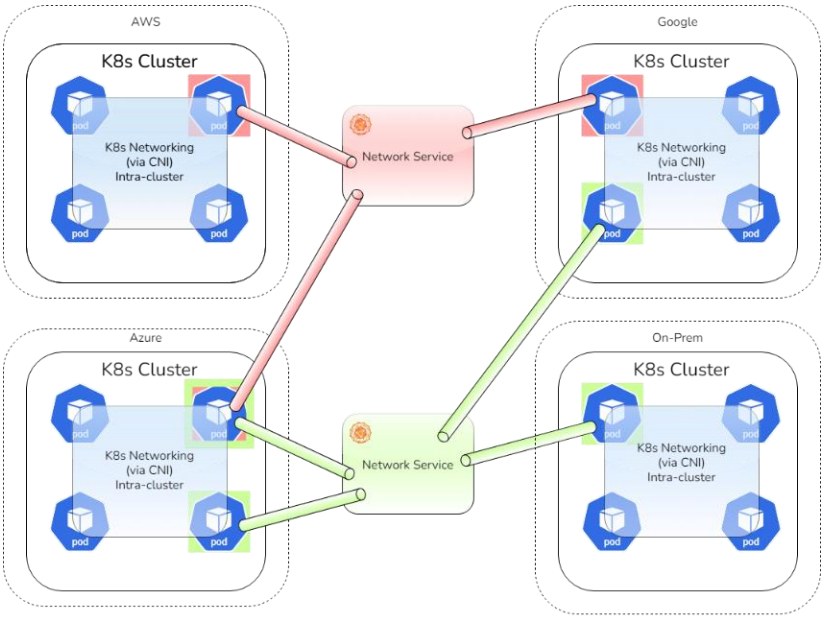


Figure 3-53: Network Service Mesh solution over Kubernetes clusters [NET24a].

3.5.1.1.3 Workflows

This section describes workflows for multi-cluster resource management, including (i) discovery and synchronization of nodes organized in multiple clusters and (ii) service deployment in multi-cluster environments. The workflows consider, as an example, a multi-cluster resource manager based on the REC-EXEC resource orchestrator, an evolution of a resource orchestrator for extreme-edge domains initially developed in Hexa-X project [HEX23-D63] (see section 3.5.1.2.1 for details on its implementation and refer to [HEX224-D33] for high-level design). This orchestrator allows dynamic discovery, continuous monitoring and inventory of extreme edge, edge and cloud continuum resources, organized in multiple clusters and exposed by platforms that could be administered by different stakeholders. The interaction with these platforms adopts a driver-based approach where platform-specific and device-specific drivers are responsible for the collection of platform-specific computing capabilities and device-specific characteristics from the onboarded platforms. This would allow to overcome the potential fragmentation of information models and interfaces in multi-vendor scenarios, as well as the diversity of devices at the extreme edge. The information collected by the REC-EXEC is then translated to common, internal information models, as detailed in Appendix 8.1.2.

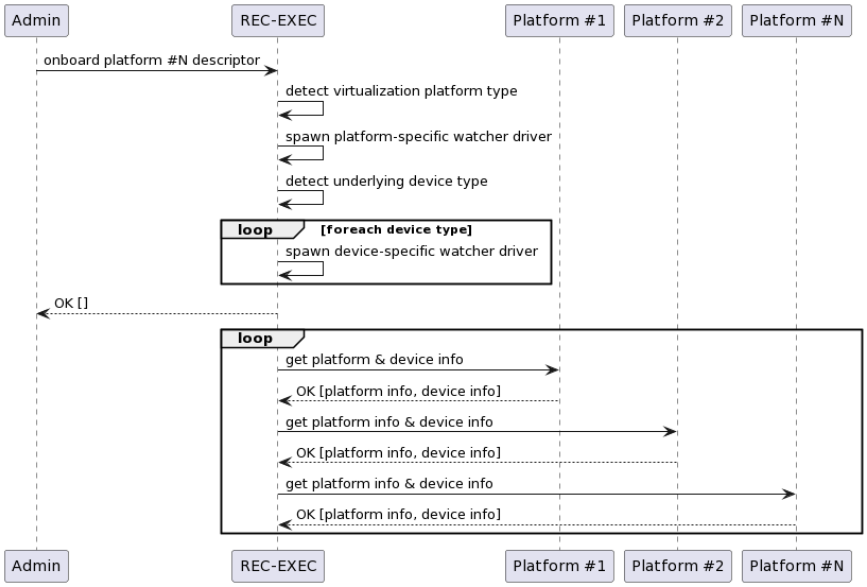


Figure 3-54: REC-EXEC workflow for platform monitoring.

Figure 3-54 depicts the high-level steps performed by the REC-EXEC to start monitoring a new platform, enabling the dynamic discovery and inventory of its underlying nodes and embedded resources, along with the ones already available. When a system administrator onboards a new platform, the REC-EXEC detects the virtualization platform type and the underlying device types that are involved, and spins-up dedicated drivers that collect the platform-specific and device-specific information. For example, in case of nodes with computing resources managed by Kubernetes-based [K8S] platforms, the REC-EXEC spawns a platform driver that leverages the Kubernetes API server to collect the computing capabilities and other relevant information of the Kubernetes nodes that compose the cluster. In the case of nodes corresponding to specific types of devices with additional capabilities (e.g., drones, cobots, IoT gateways, etc.), the REC-EXEC deploys dedicated drivers to collect additional device-specific characteristics and information -beyond computing resource capabilities- following the way those devices expose them. Devices are always considered as part of a cluster, which may be limited to a single element.

The driver-based approach is used to enable the dynamic discovery, continuous monitoring and inventory of the computing resources available in the nodes within the clusters, controlled through computing platforms like, e.g., Kubernetes or K3S [K3S]. Each platform, in turn, can expose one or more clusters. Moreover, extreme-edge nodes often consist of devices with additional capabilities, beyond computing resources, which could be jointly controlled (e.g., movement of a cobot, commands to an IoT actuator, etc.). This per-device information is collected with additional drivers, customized for each type of device. The implementation allows to plug and unplug specific drivers, even dynamically at runtime, depending on the scenario and on the types of involved devices. In parallel, it introduces a unified approach to handle computing resources in the continuum for multi-cluster management. The same driver-based approach is used for the execution of orchestration operations towards different types of virtualization platforms, e.g., for service provisioning, scaling, migration, etc. For example, the Kubernetes specific deployer (i.e., driver) leverages the Kubernetes API server to execute the deployment of new virtualized applications.

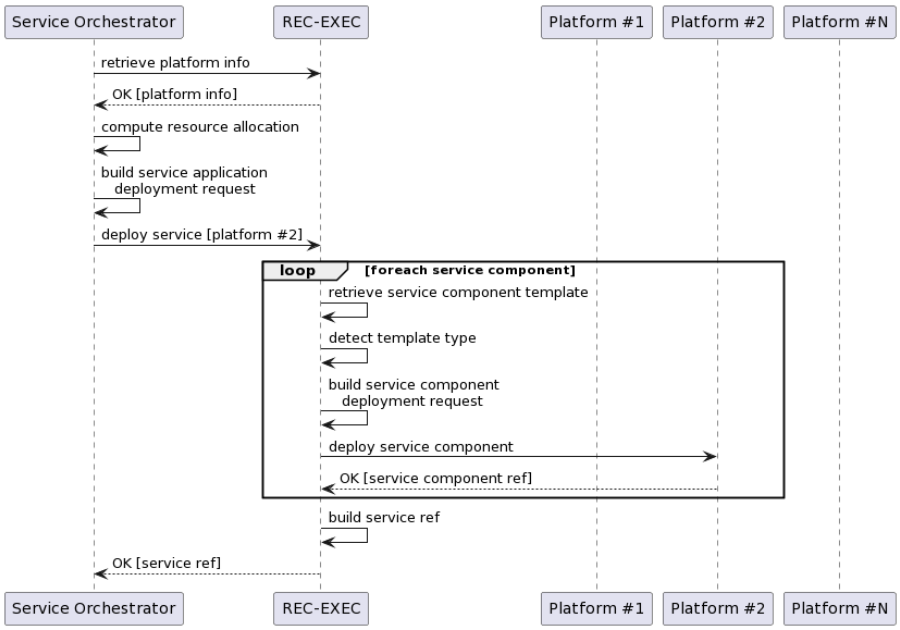


Figure 3-55: REC-EXEC workflow for service deployment.

The REC-EXEC exposes a unified and platform-agnostic set of APIs, leveraging the platform-agnostic service deployment request information model detailed in Appendix 8.1.2, to enable the deployment of applications. Thus, each deployer, when invoked for the deployment of a service component, has to translate the agnostic high-level requirements specified in the application component deployment request in the platform-specific deployment payload making also use of the orchestrator template (e.g., Helm Chart, plain Kubernetes descriptors, Heat Template, etc.) specified for that particular application component. Figure 3-55 depicts the full workflow executed by the REC-EXEC to deploy an application upon receiving a deployment request from a Service Orchestrator that has already performed the allocation of the components using the available Continuum resources retrieved from the REC-EXEC itself (i.e., the continuum resource that has been discovered from REC-EXEC drivers).

3.5.1.2 Preliminary implementation and early validation results

3.5.1.2.1 REC-EXEC

This section describes the preliminary implementation of the REC-EXEC multi-technology resource orchestration platform. It constitutes an example of multi-cluster resource manager, which has been developed as an evolution of the extreme-edge resource orchestrator initially implemented in the Hexa-X project [HEX23-D63]. The software architecture is depicted in Figure 3-56: the prototype is a microservices-based platform where each module, developed as a REST Java Spring Boot Application, contributes to implement the functionalities of the orchestrator.

The Platform Manager software component enables the dynamic discovery, continuous monitoring of Extreme-Edge, Edge and Cloud Continuum resources, organized in multiple clusters, and the virtualized applications orchestration operations, leveraging a driver-based approach where platform-specific and device-specific drivers operate over the underlying virtualization infrastructures and devices. These drivers are embedded in an agnostic skeleton and plugged or unplugged depending on the specific scenario. The Platform Manager includes a PostgreSQL database to store the information about the onboarded platforms (i.e., the platforms whose resources and characteristics, also in terms of devices, are discovered and monitored by the Platform Manager drivers).

The Resource Manager software component retrieves the platform-specific and device specific information collected by the Platform Manager drivers from an internal Kafka message broker and exposes the discovered information (i.e., the Extreme-Edge, Edge and Cloud Continuum resources) through a REST interface. The Resource Manager includes an H2 in-memory database to store the information of the platforms discovered and monitored by the Platform Manager. Both the Platform Manager and the Resource Manager implement the information models outlined in Appendix 8.1.2.

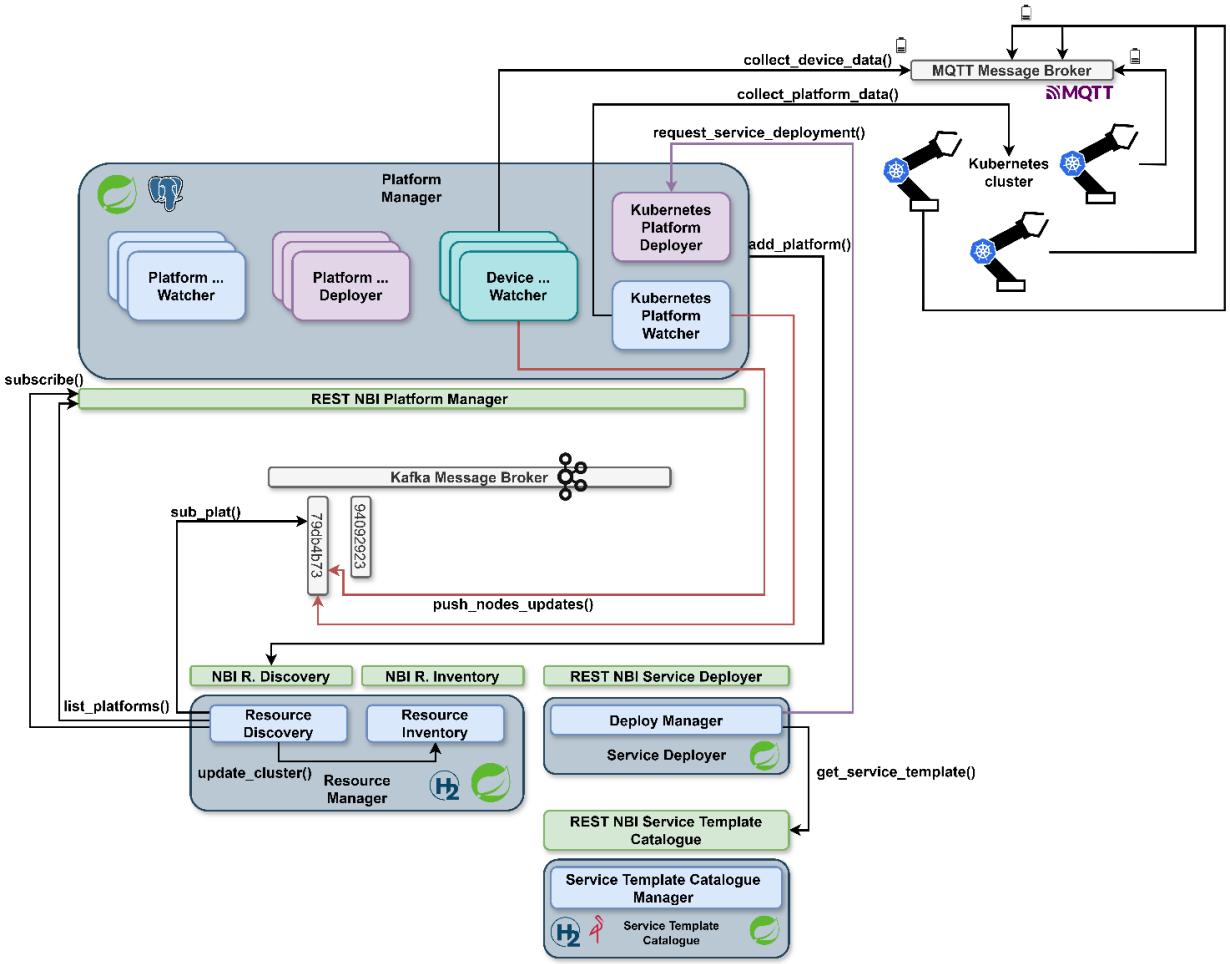


Figure 3-56: REC-EXEC implementation.

The Service Deployer module enables the platform-agnostic applications orchestration operations exposing a REST interface that leverages the deployment information models outlined in Appendix 8.1.2. The Service Deployer takes care of translating the application orchestration requests in the exact formats that can be handled by the platform-specific deployer driver in the Platform Manager that can manage the target virtualization platform types.

The Service Template Catalogue exposes a REST interface consumed by the Service Deployer to retrieve the platform-specific orchestration templates to be used to fulfil specific orchestration operations (e.g., a Helm Chart [HEL24] template to perform the deployment of a service component in a Kubernetes cluster managed by the REC-EXEC). The Service Template Catalogue leverages an H2 in-memory database to maintain the metadata of the orchestration templates stored in a MinIO Object Storage managed by the Service Template Catalogue software component itself.

In the context of the PoC-B, the REC-EXEC has been used to discover the resources and the device characteristics of a Kubernetes cluster made of cobots (provided by one of the consortium partners, VTT) and to perform the deployment and migration of a cobot application. The deployment of the application is requested by a Service Orchestrator to the REC-EXEC targeting a specific node of the cluster chosen by the Service Orchestrator upon the resources discovered and monitored (i.e., the cobot cluster resources and battery level information of the cobots themselves) by the REC-EXEC. Additionally, a migration of the application is triggered (and executed by the REC-EXEC) by the decision and execution functions of a closed loop operating at service and infrastructure layers. This automates the migration of the application in the Extreme-Edge domain when the battery level of the cobot where the application has been originally deployed decreases beyond a certain threshold (for further details on the closed loop for service migration see section 3.8.2.1).

In order to leverage the REC-EXEC in the PoC-B, the information models for resources in the continuum have been extended to represent the characteristics of the target cobots. In particular, a VTT Cobot class has been introduced as concrete extension of the abstract class Cobot (sub-class of Device Info) to maintain the peculiar information of the VTT cobot, including the battery level of the devices following the format exposed by the cobots themselves. This information has been collected by a new device driver introduced for the PoC.

Early Validation Results

In PoC-B, the Resource Orchestrator REC-EXEC has been deployed in the VTT testbed alongside the Monitoring Platform (section 3.2.2.3), the Closed-Loop Governance (section 3.8.2.1) and a Service Orchestrator.

The testbed dedicated to the PoC-B is also providing a Kubernetes cluster made out of six nodes where two of them are cobot devices working as targets for the deployment of a network monitoring application; such application will be deployed in one of the two cobots and then moved to the other to carry on the monitoring job when the battery level of the first target goes below a certain threshold.

In the described scenario, the roles of the above-mentioned services are the following:

- Resource Orchestrator REC-EXEC: discovers and monitors the computing resources of the PoC-dedicated Kubernetes Cluster and the device characteristics of the cobots that are part of the cluster itself, in particular their battery information collecting the latter directly from the Monitoring Platform. Deploys and migrates the network monitoring application among the cobot Kubernetes workers.
- Monitoring Platform: collects the battery level information of the cobots in order to make the latter available for historical and real-time management. The information is collected from an MQTT message broker available in the Kubernetes cluster where the cobot devices push the information regarding their batteries.
- Closed-Loop Governance: responsible for the management of the lifecycle of closed-loop instances associated with a service application. In case of the PoC-B, it is responsible for the closed-loop associated with the network monitoring application, having as goal the migration of the application when the battery level of the target cobot goes below a predetermined threshold. The closed-loop functions (analysis, decision and execution) that are part of the closed-loop involved in this scenario are being deployed by the Resource Orchestrator in the PoC-B dedicated Kubernetes cluster. Their objective is to continuously monitor the battery level of the cobot chosen as target for the deployment of the network monitoring application by fetching the relevant real-time data from the Monitoring Platform and then request the

migration once the threshold is reached. The communication (i.e., event-based messaging) between the closed-loop functions is implemented leveraging a Kafka message broker.

- Service Orchestrator: works as a higher-level orchestrator managing the full lifecycle of service application instances and their associated closed-loops when its instantiation is requested through its APIs. The Service Orchestrator makes use of a Resource Allocation service to determine the placements of the components of the service application and the closed-loop functions making use of the resources discovered, monitored and collected by the Resource Orchestrator. In the context of the PoC it will chose one of the two cobots for the network monitoring application and other worker nodes of the Kubernetes cluster for the three closed-loop functions. The Service Orchestrator leverages the APIs of the Resource Orchestrator for the deployment operations of the service applications and the APIs of the Closed-Loop Governance for the deployment operations of the closed-loops (i.e., that in turn makes use of the Resource Orchestrator APIs). The Service Orchestrator also exposes APIs for starting the migration of a service application instance; such APIs are used by the execution function of the closed-loop to request the migration of the network monitoring application when the threshold is reached.

The deployment of the network monitoring application alongside the dedicated closed-loop (i.e., the closed-loop analysis, decision and execution functions) in the PoC-B Kubernetes cluster has been tested and validated successfully, migrating the application on a threshold base from one cobot to another. The instantiation operation of the service application instance, comprehensive of the network monitoring application and the closed-loop functions, (i.e., from Service Orchestrator to Resource Orchestrator and Closed-Loop Governance) requires an average of ~36 seconds: ~20 seconds for the deployment of the network monitoring application and ~16 seconds for the deployment of the three closed-loop functions. The above-mentioned timings consider also the time needed by the network monitoring application and the closed-loop functions to be up and running in the PoC-B Kubernetes cluster (i.e., RUNNING pods' status). The average time needed for the decision closed-loop function to check the cobot battery levels received through the analysis function (i.e., collected by the Monitoring Platform; in this scenario the analysis function works as an intermediary) and sends an event to the execution closed-loop function if the threshold is triggered, inclusive of the time needed by the execution function to receive such an event, is less than a second. Finally, the average time needed to execute the migration operation, from the reception of the event by the execution closed-loop function to the network monitoring application being up and running in the other cobot worker, is ~5 seconds.

3.5.1.2.2 RL-driven autoscaling

An initial implementation of the RL-driven autoscaling mechanisms towards their integration with Component PoC#B.1 is illustrated in Figure 3-57. A setup of two clusters has been deployed as an initial environment for the application graph of the PoC. A preliminary loop instance of the RL autoscaling workflow is the main object of this description and of the corresponding experiments later on.

As shown in Figure 3-57, the application designed for PoC#B.1 is deployed in a two-cluster multi-cluster setup in a lab deployment emulating the edge-cloud environment. For validating the RL agent's functionality, a set of initial experiments were executed based on a low-high workload scenario. In specific, a low-level workload rate (frames/sec) is considered for normal operation of the application, while a high-level workload is applied when there is potential risk, so a faster detection is required.

For the specific experiments, the state of the agent is constituted by the replicas deployed for the service (1-3 replicas), their placement (edge=0 or cloud=1) and the current workload. The agent is responsible for deciding the replica number suitable for the deployment and whether these replicas should be deployed at the cloud or the edge. The reward considers end-to-end latency and is calculated as -100 if the SLA is not respected or else using the following formula:

$$\mathrm{R} = 100 * \frac{SLA - lat}{SLA}$$

The complexity of the specific problem comes from the fact that the edge server has by design low resources to support high numbers of replicas and increased parallelization may cause performance to deteriorate. End-to-end latency of a service includes communication as well as computation latency. When executing a service at the cloud introduces higher communication time, but the average computation time when more replicas (which the cloud server may be able to support) are available is decreased. Thus, the agent needs to identify

the optimal point to which to increase the replica number of the service without decreasing performance, while considering the difference in communication time in each case.

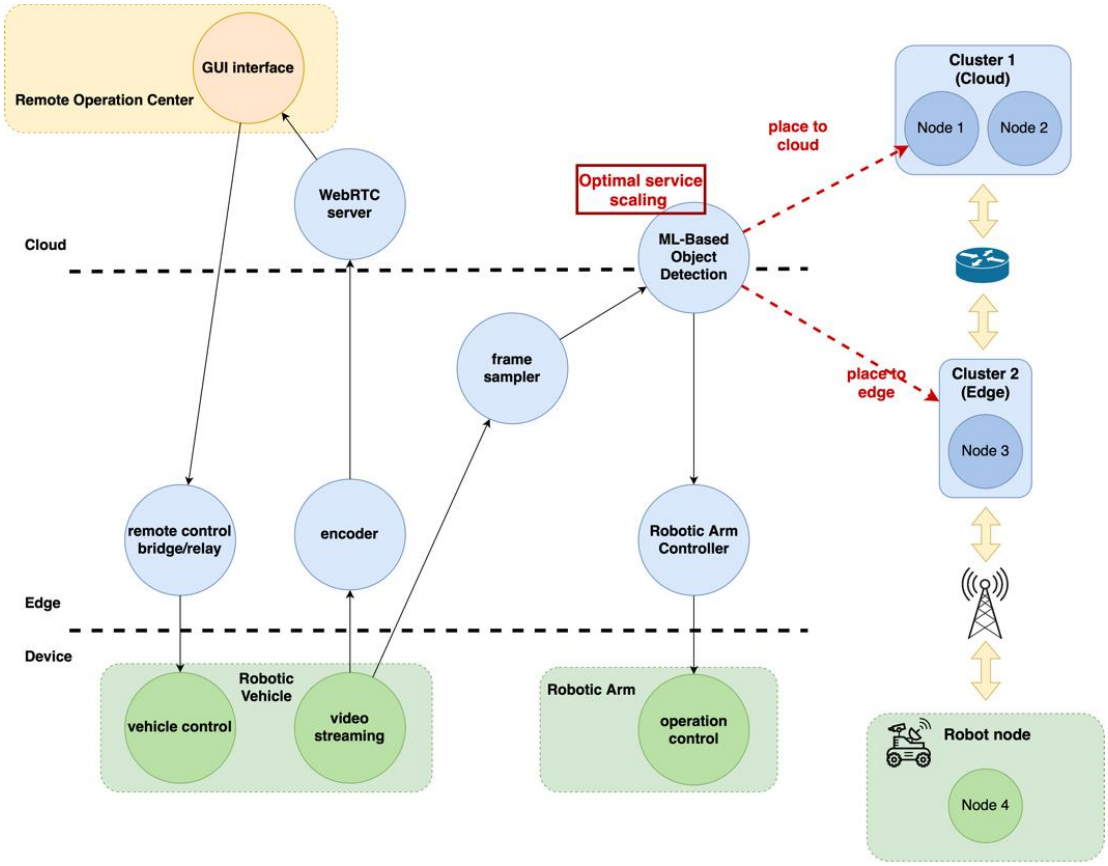


Figure 3-57: RL-driven autoscaling implementation.

Figure 3-58 shows the initial results from the deployment described above. The described workload is given as input in the form of a pulse, which is what is expected based on the specific scenario. The agent learns how to optimally combine the placement and scaling decisions and identifies that for low workloads, placement with 1 replica at the edge (placement=0) provides lower latency, while high workloads demonstrate additional stress and need to be deployed at the cloud (placement=1) with 3 replicas. The deployed version of the algorithm is reactive, meaning that it does not consider forecasted values of the workload and thus, demonstrates a delay in optimal decision making. This is obvious in Figure 3-58 where we see that latency is below the SLA for all cases instead of the workload level shift points where unanticipated workload increase causes latency increase, which is regulated in the next steps.

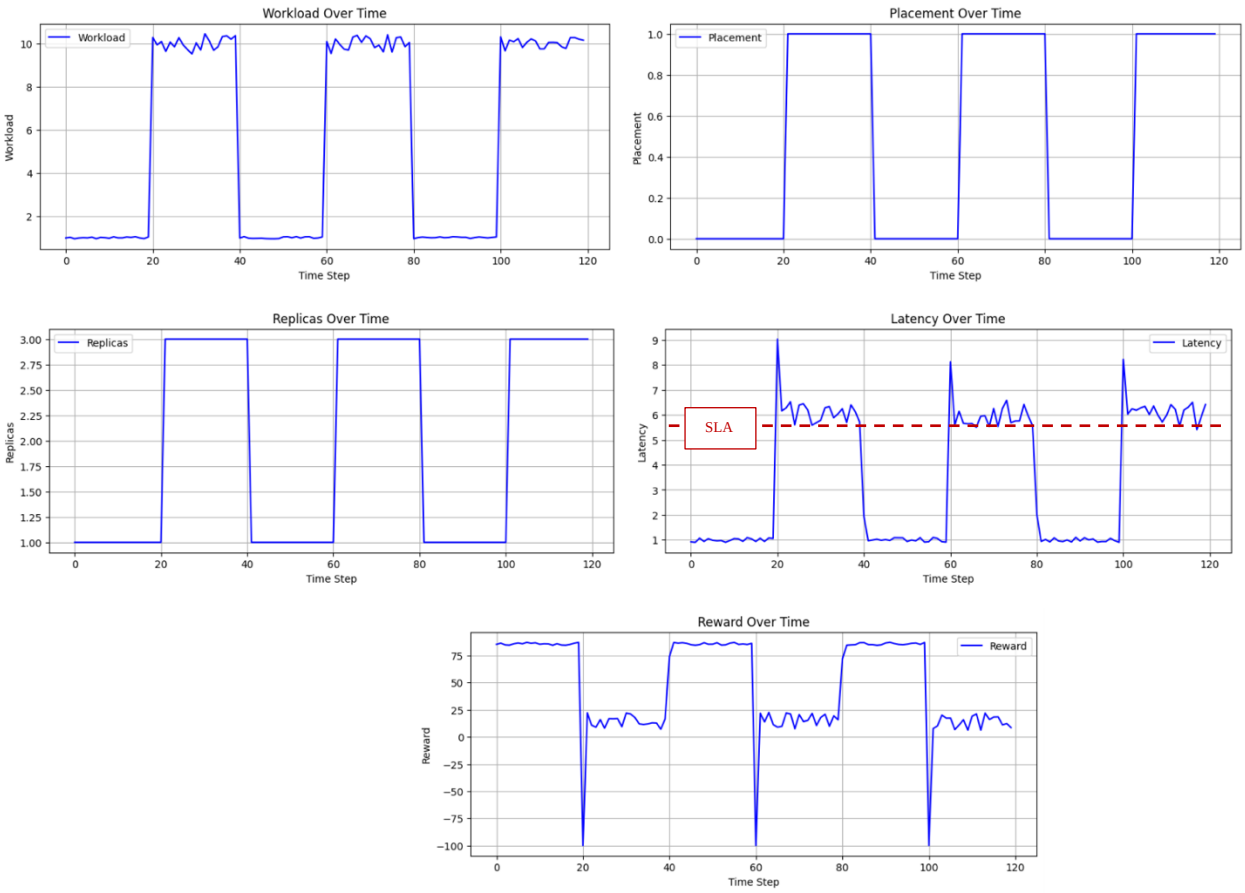


Figure 3-58: RL autoscaling early results.

3.5.1.2.3 Development and communications of functionality allocation mechanism

The computational workload placement functionality allocation algorithm developed is a metaheuristic algorithm based on the genetic algorithm paradigm [Yan14] as described in Section 3.6.1.2.3. The algorithm accepts computational workloads with their computation and physical demands, the volume of data, and the data's origination points corresponding to each workload. It also considers the available compute nodes with their capabilities, trust indexes, along with the network topology graph. The algorithm's output is a nearly optimal assignment of these computational workloads to the available compute nodes towards energy efficiency and trustworthiness. It communicates with the observability/KPI monitoring component for collecting the capabilities of the available compute nodes and the network topology graph. It also communicates with the service registry for the computation and physical requirements of the computational workloads, the data's origination points and the size of the data corresponding to each workload. The functionality allocation algorithm additionally considers the various trust indexes of the compute and infrastructure nodes coming from the trust evaluation function (sub-enabler 4.3). The output of the algorithm is sent to the multi cluster manager which then manages the system's resources. All these communications are controlled by the northbound interface (NBI) component named API Server. This component triggers the algorithm in the case of a possible intent coming from an end-user or when there is a need for reallocation (e.g., increased latency, malfunction in an extreme-edge component) and accepts requests from the functionality allocation component for data as well as it ensures the feasibility of the algorithm's output before sending it to the multi cluster manager (see Figure 3-59). The data model used for this procedure is shown in Figure 8-7. In this schema, the three types of compute nodes are shown with their capabilities, some example application components are given with their requirements and the location of the generation of the utilised data. Finally, some main metrics are quoted for physical and virtual resources, network, and application layer. The described development could be extended for multi domain orchestration where the management capabilities exposure framework which collects the various available resources (application, AI, cloud, network domain) could communicate with the observability/ KPI monitoring component and the API Server and then trigger the functionality allocation mechanism if there is a need.

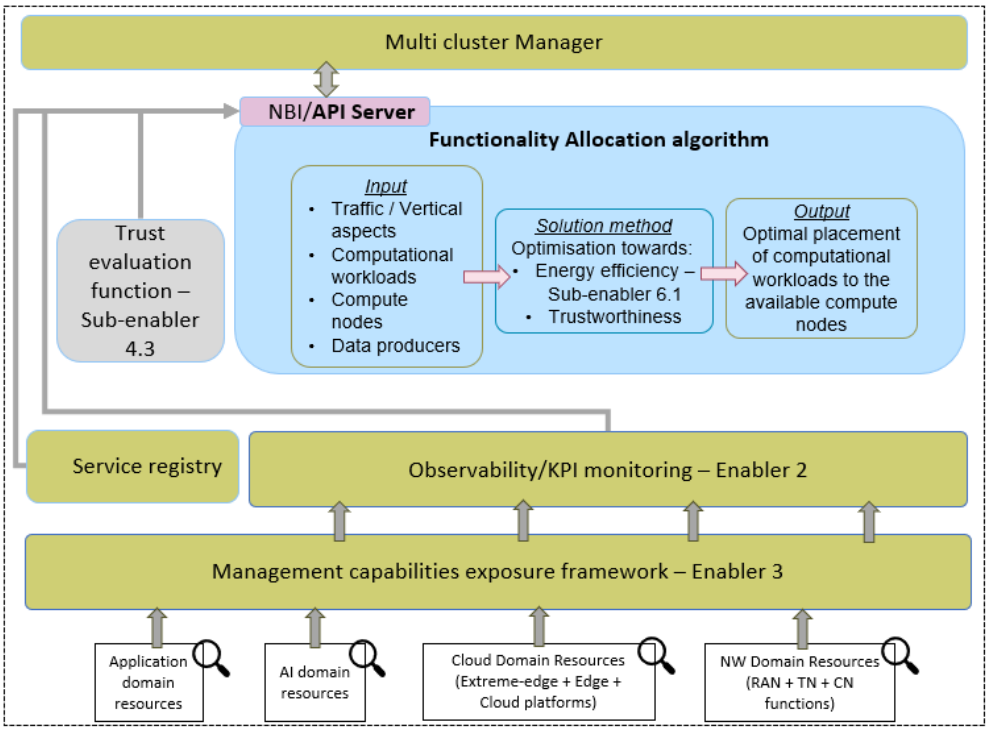


Figure 3-59: High level view of the communications for the functionality allocation mechanism.

3.5.1.3 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this sub-enabler are the following:

- **Latency**: the provided mechanisms for optimal placement and runtime management of network/application service graphs can lead to improvements in the end-to-end latency for the provision of a service/application.
- **Reliability**: the continuous monitoring/observability of the performance metrics for the various deployments can lead to proactive and reactive decision making for the management of events and failures in the service provision and the infrastructure management,
- **Scalability**: the provided mechanisms for autoscaling of the compute resources across the continuum lead to increased efficiency of the scaling decisions, considering performance, energy and cost metrics.
- **Programmability:** the provided mechanisms manage programmable compute and network resources across the computing continuum, while the provided orchestration interfaces are fully programmable and configurable.
- **Automation**: automation and decentralized intelligence characteristics are injected within the various orchestration mechanisms, leading to reduction of the administration overhead by network/system administrators and optimal services provision.

Regarding the KVIs, it is considered that the most evident to be affected by this sub-enabler is **Sustainability**. The developed orchestration mechanisms can support optimal placement and lifecycle management of distributed network services and applications from an energy efficiency perspective.

3.5.2 Sub-enabler 5.2: Decentralised orchestration system

This sub-enabler is based on the work regarding virtualisation and the cloud transformation studies in WP3 (6G Architecture design), and initially described in [HEX223-D32] and [HEX224-D33]. It proposes a decentralised M&O approach, targeting to integrate the broad heterogeneity of stakeholders envisaged for 6G in the M&O processes, as well as the consequent diversity of administrative and technical network domains, including the extreme-edge. The approach basically relies on two main ideas:

- The deployment of multiple instances of a reduced set of network elements intended for the network resources management and the network services provisioning, which would be distributed through the entire network continuum.

- The embedding of service-specific, tailor-made, and diverse M&O components as part of the network services themselves, intended for the network services assurance, which would also be distributed through the network continuum.

Aligned with the cloud-native concept, this enabler tries to address one of the main challenges regarding the M&O of the network resources and the network services in the future 6G networks: the extension of the M&O scope beyond the MNO's own domain, integrating also the extreme-edge domain, in line with the *continuum orchestration* concept [HEX22-D62] [KLM+22] [RCV+22].

In this regard, the integration of the extreme-edge is considered specially challenging, due, on the one hand, to its size and complexity. The extreme-edge domain can include high number of domains and devices, with a cloud native scale, and with devices integrated in different networks belonging to multiple stakeholders that could be deployed, e.g., in homes, vehicles, industrial environments, robots, satellites, smart cities, ground transport routes, agricultural fields, and more. On the other hand, the network resources in that domain can be highly heterogeneous, with many kinds of devices that could be asynchronously connected or disconnected (they won't be necessarily in well-controlled premises), move, or with limited and/or unexpectedly changing computing or storage capabilities.

However, putting this concept into practice involves more than simply offering connectivity to a variety of devices outside the MNO access network (which is already the case today, e.g., by integrating IoT devices to gather data from them), but also, to make it possible to deploy and orchestrate NS components on those "beyond the edge" infrastructure resources in a cloud-native way. The reason is that, unlike what happened in previous generations, the computing power and storage capacity of the extreme-edge devices can be significant, especially if they are considered as a whole, which could provide an additional valuable set of infrastructure resources that in many cases could be very close to the end-users, which may help reducing latency and distributing workloads, and reducing data communication needs.

As its name states, this sub-enabler 5.2 tackles this challenge relying on a *decentralised* M&O approach. The rationale behind it is to make the system highly flexible and scalable. It is considered that it would be impractical to deal with the large number and diversity of devices on the extreme-edge domain, as well as with the large number of microservices that could be deployed on them, in a centralized manner (e.g., just the gathering and the processing of the monitoring and diagnostics data from a huge amount and diversity of resources in a centralised way could be a relevant challenge by itself). Besides, there are other problems envisaged for a possible centralised approach, e.g., the capacity planning with non-owned resources (the extreme-edge includes resources beyond the operator own premises), the well-know "single-point-of-failure" issue associated with the centralised approaches, or the increased operational costs (managing a network with many elements and mechanisms can be very costly for a single stakeholder).

On the other hand, a decentralised approach offers better resources optimization, since network services may be very different in scale and complexity, so not always requiring the same kind of orchestration needs (a common MNO-centric orchestration solution would bring the same orchestration framework for all the network services, while the decentralised approach relies on tailor-made "adapted to each service" approach). Besides, as described in the following sections, the decentralised approach is "multi-domain by design": service chaining would be performed through multiple domains relying on the service components exposed interfaces, in a cloud-native way, relying on the microservices federation concept [GKV+19] [FSP+20], which contrasts with the centralised approach that typically relies on complex business and technological agreements among different MNOs to communicate different centralized orchestration frameworks.

3.5.2.1 Sub-enabler design

From a design perspective, this sub-enabler 5.2 proposes to include two main conceptual updates in the high-level view of the well-know ETSI NFV MANO framework [NFV13] represented in Figure 3-60, as highlighted in Figure 3-61, i.e.:

– To add the explicit declaration of the application scope of the ETSI NFV MANO framework which, as can be appreciated, extends through the entire network continuum, represented in the figure by the outer frame with the cloud on top, and not only in the domain of a single MNO, as in Figure 3-60.

– To introduce the concept of *cardinality* (represented using the crow's foot convention [Eve76]) to explicitly denote that there would be multiple MANO functional blocks (which would be distributed) associated to the VNFs and the NFVI layers.

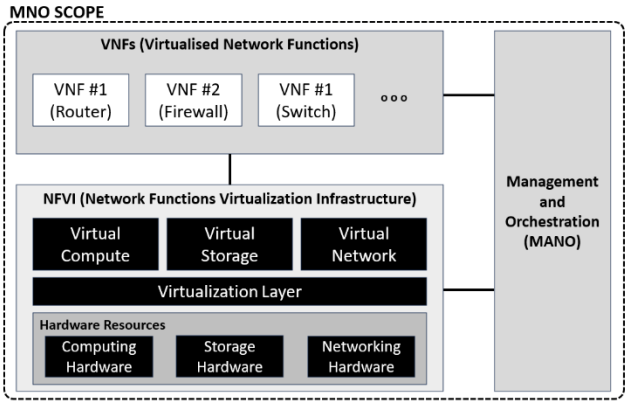


Figure 3-60: ETSI NFV MANO Framework.

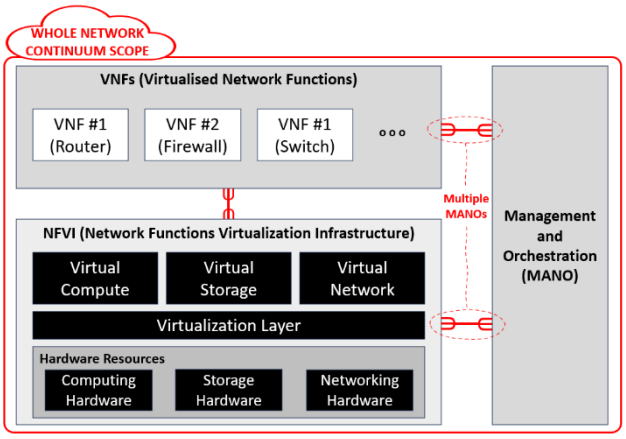


Figure 3-61: Proposed update aligned with the ETSI NFV MANO Framework.

Although at first glance it may appear to be only two minor changes, they may have relevant implications in the M&O architecture towards 6G (though still in good alignment with the abstractions of the ETSI NFV framework): on one hand the expansion of the ETSI NFV MANO framework through the whole network continuum shows that the other architectural blocks (VNFs on top and infrastructure resources at the bottom) do not necessarily have to be associated to a single stakeholder (e.g. the MNO) but distributed through different domains of the network continuum, and associated to different stakeholders. I.e., VNFs in the Virtualized Network Functions block on top would be grouped together to compose Network Services as the ETSI NFV MANO framework defines, but with the decentralised approach, those VNFs could be provided and executed by different stakeholders distributed through the whole continuum (instead of being deployed on a single MNO domain), so that both, industry and MNOs can benefit in tandem. Besides, the NFVI layer at the bottom would still represent the physical and virtualized infrastructure (also as the ETSI NFV MANO framework defines), but in this case, such infrastructure represents all the computing/storage/networking resources distributed along the entire network continuum, including the regular (5G) core and edge domains, but also, extreme-edge resources beyond the MNO own domain. On the other hand, the inclusion of a cardinality “>1” regarding the MANO block, and its implementation as part of the network continuum, indicates the possible implementation of multiple MANO blocks, which could be also distributed throughout the network continuum.

However, to make this view possible in practice, in addition to these two conceptual changes, this sub-enabler also proposes other design updates in terms of implementation, namely:

- The multiple MANO blocks resulting from considering that cardinality ">1" mentioned above would be implemented in two ways:
  - There would be one “common” decentralised (i.e., distributed through the network continuum) functionality specifically devoted to the network resources orchestration (those at the NFVI) and the network services provisioning. This functionality would be provided by the so-called Common Infrastructure Management and Services Provisioning System, which will be explained in more detail in Section 3.5.2.1.1 (System components) below.
  - A set of tailor-made, distributed, and diverse MANO resources embedded in the network services themselves, specially oriented to implement the necessary services assurance mechanisms for those services to which they are attached.
- Beyond to what was originally proposed for the ETSI NFV MANO framework, VNFs would be used to implement not just common network infrastructure devices (e.g., routers, switches, firewalls…), but other network functions as well (e.g., AI/ML functions, monitoring functions, billing functions, etc.), either to implement the network specific functions (e.g., management functions, RAN functions…) or functions oriented to implement the service logic of those network services deployed on the network (e.g., application oriented network functions).

- The implementation would be based on modern cloud-native principles, i.e.,
  - Instead of relying on VMs to implement VNFs (as suggested in [NFV13]), they would be primarily implemented through lightweight containerised micro-services (or Containerised Network Functions -CNFs- as they are sometimes referred). However, it is also not excluded that other virtualization technologies could be used to implement these VNFs as well (e.g., legacy technologies such as VMs, or new virtualization technologies that may appear in the future).
  - Instead of based on strictly defined reference-points (like in [NFV13]), communication among VNFs would be also cloud-native, i.e., based on the micro-services exposed interfaces. This would make this approach "multi-domain by design", enabling services federation through multiple network domains by relying on the micro-services exposed interfaces.

The following sections provide more detailed information about the system components' design, as well as the main envisaged workflows and interfaces. Information is also provided on a preliminary implementation of this concept (which is being integrated as part of the PoC B of the project), as well as on the most relevant KPIs and KVIs that could result positively impacted by this sub-enabler.

3.5.2.1.1 System components

The application of the ETSI NFV MANO framework on the entire network continuum considers the deployment of multiple MANO blocks, with one of these blocks considered common (the so-called Common Infrastructure Management and Services Provisioning System – CIM&SPS), while the others would be tailor-made, and attached to each NS to provide the specific service assurance mechanisms each service may require. The idea behind is to decouple service onboarding (which is considered a main "common" feature of the network) from service assurance (which can be addressed specifically for each service, in a decentralized way, and with an approach that in many cases may be more lightweight than in the regular MNO-centric approach). The components considered for each of type of MANO blocks are described below.

Common Infrastructure Management and Services Provisioning System components

The CIM&SPS is envisaged to be composed of four so-called distributed network stakeholder support services, which were already introduced in [HEX223-D32] and [HEX224-D33]). They are the following:

- The Deployment Service (DS).
- The Infrastructure Registry Service (IRS).
- The Services Registry Service (SRS).
- The Infrastructure Status Prediction Service (ISPS).

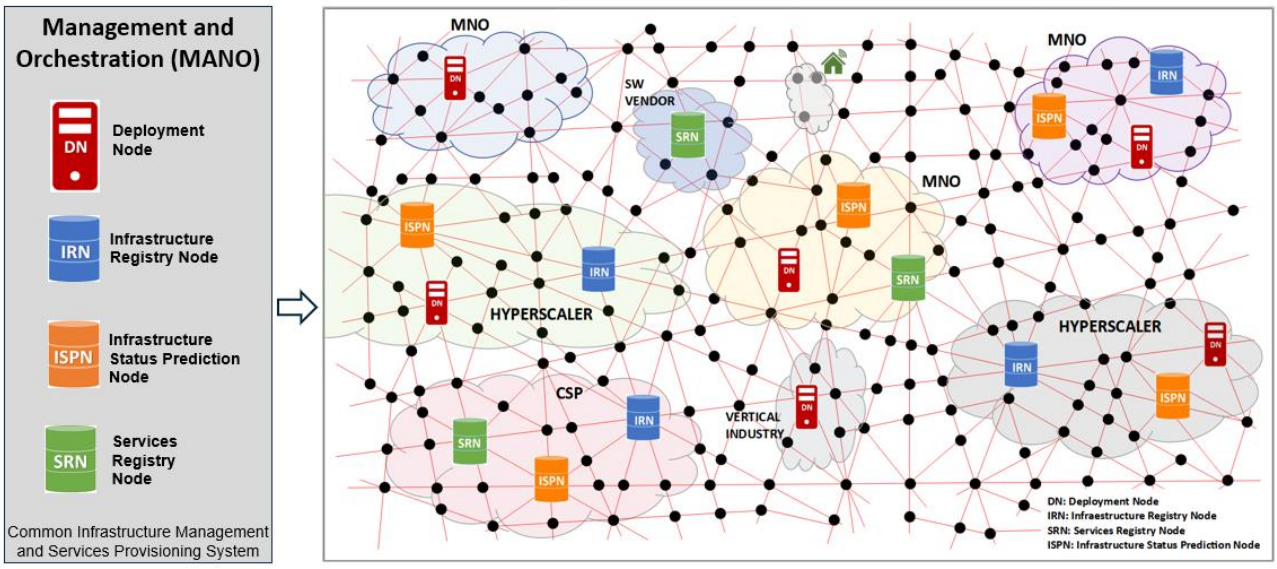


Figure 3-62: Common Infrastructure Management and Services Provisioning System.

Collectively, these four stakeholder support services facilitate the provisioning of new network services in the network, being implemented by four specific components associated with each service: Deployment Nodes (DN), Infrastructure Registry Nodes (IRN), Service Registry Nodes (SRN), and Infrastructure Status

Prediction Nodes (ISPN). Multiple instances of these components would be distributed through the network to implement each of the services (see Figure 3-62). As can be appreciated, these nodes would be issued as part of the facilities provided by different stakeholders (MNOs or others) with access to the network resources and with the capability to host and manage these network components.

The functionality of these four services (made of the corresponding kind of nodes) was already described in [HEX224-D33]. Below a summary:

- The IRS is basically a sort of distributed database playing a main role regarding resources orchestration on the network continuum. Its main objective is to keep an accurate and real-time updated registry of the network resources in the continuum, considering their heterogeneity and high volatility. However, beyond the registry functionality itself, the service is also provided with automatic infrastructure discovery mechanisms in charge of exploring and storing updated information on the actually available infrastructure resources (virtual or physical, in line with the NFVI definition). This service can be understood as something like the well-known Internet DNS service, but automatically updated and containing more detailed information about the infrastructure resources in the network, mainly in what regards their current availability, resources (computing, storage…), type (small/medium/high scale, fixed/mobile, battery powered…), physical location, etc.
- The ISPS is an ancillary service that would enrich/complement the information provided by the previous IRS by adding information regarding the "future" availability of the infrastructure resources, to enable proactive M&O mechanisms (e.g., this service could be used to predict if a certain infrastructure node, on which certain service components could be deployed, will still be available -or with the required amount of resources- after certain time). That future availability information would be provided based on applying data analytics algorithms (e.g., by means of AI/ML algorithms) on the infrastructure resources.
- The DN (Deployment Node) would be used to deploy the network services on the network, which would be defined as an aggregation of network service components (VNFs, in the form of microservices) for which specific deployment descriptors would be provided specifying the requirements of the infrastructure devices on which each VNF should be deployed. Based on that, and on the information provided by the IRS and the ISPS services, the DN would compute the "best match" between those requirements in the descriptors and the available infrastructure resources to deploy the network services in an optimal way (e.g., a NS consisting on different microservices could be requested to be deployed with certain components in the core network of an MNO, other components on a hyperscaler, and other components on a set of IoT devices with certain hardware requirements; based on that, the DN would be in charge of selecting the set of devices on which the complete NS would be instantiated). DNs would be also spread over the network, and would be able to aggregate VNFs provided from different parties (i.e., not only the party hosting the DN) to compose the network services [HEX224-D33].
- The SRS is an ancillary service intended to support the DN. It is also a registry, which would be implemented by means of a distributed database (as the IRS), but in this case, containing information on the current execution environment for the already deployed services. Since, considering the high volatility of the extreme-edge domain, that execution environment can dynamically change (certain nodes hosting service components can become unexpectedly unavailable), this SRS provides information on where the network services can be located at all times, even if they have been migrated from one node to another. The service is considered ancillary because certain network services could be designed to be partially or totally deployed on non-volatile nodes, so the use of this SRS would not be necessary in such cases.

As it can be noted, the internal composition of this common MANO block described here does not implement the well-known functional blocks in [NFV13], i.e., the NFV Orchestrator (NFVO), VNF Manager (VNFM), and Virtual Infrastructure Manager (VIM). That is intentional, and in fact, it is considered one of the main innovations introduced from this sub-enabler: while those NFVO, VNFM, and VIM were designed to be primarily deployed per-MNO (i.e., in an MNO-centric approach) the solution described here is inherently decentralised, with multiple instances of the above-mentioned blocks (i.e., DN, IRN, ISPN and SRN) targeting the M&O of the resources in the whole network continuum and the network services provisioning. However, this does not mean that these three functional blocks (i.e., NFVO, VNFM and VIM) are fully out of scope: they could be part of a particular case to implement some of the MANO blocks providing the service assurance mechanisms for the network services. More information on this is provided right in the following subsection.

Specific service assurance mechanisms

The ">1" cardinality of the MANO block in Figure 3-61 does not mean that all these blocks are different instances of the same class, but different implementations of the MANO functionality that could be done based on different approaches. One of those implementations is specific and well-defined (the Common Infrastructure Management and Services Provisioning System described right before), while the others would be tailor-made according to the specific needs of the network services to which they would be attached. Of course, a group of network services could rely on the same technical approach (e.g., by relying on a common MANO framework), but the key idea here is to give the NS developers the freedom to choose the most appropriate service assurance approach, adapted to each specific service needs.

Network services can be very different in size and shape: some could be quite complex, with many interacting VNFs, strict requirements, and with a big number of involved stakeholders. However, others could be simpler, requiring only basic service assurance mechanisms, e.g., to just keep a small set of VNFs running and granting access to a limited set of M&O operations. Based on this, this sub-enabler considers that, instead of imposing a common and complex "one-size-fits-all" solution for implementing the service assurance mechanisms, it is more appropriate to rely on ad-hoc tailor-made mechanisms, well adapted to each specific NS requirements. Those mechanisms would be part of each network service, i.e., each NS, would embed the specific functionalities to ensure the required availability of the service components, the migration mechanisms (if needed), the mechanisms to collect and communicate the service metrics, the access facilities to the involved stakeholders… and whatever other service assurance mechanism the service may require, providing in this way a high level of flexibility and scalability, with the M&O assurance mechanisms distributed through the network together with the services to which they were attached. Different SoTA technologies could be used for implementing these per-service M&O mechanisms, such as different containers M&O solutions (e.g., [K8S], [K3S], [NOMAD], [SWARM]), ad-hoc systems specifically developed for the services, or microservices choreographies (i.e., self-orchestrated microservices not relying on any external orchestration component) [CDT18], among others. This approach also keeps the possibility to rely on other technical solutions that may be available in the future. Besides, it can facilitate the integration of verticals or other stakeholders in the ecosystem since, instead of having to adapt to a pre-defined M&O system, they just could integrate their already available components with their specific service assurance mechanisms.

As mentioned at the end of the previous section, one of those specific approaches could be the application of the ETSI NFV MANO framework as defined in [NFV13], i.e., by relying on the well-known NFVO, VNFM and VIM components. This could be the case, for example, of those MNOs requiring implementing their own network services management solution aligned with such framework. In this specific case, the deployed NS in the continuum would be a "management service", i.e., a service able to manage other network services for the MNO and implemented according to the ETSI NFV MANO specification.

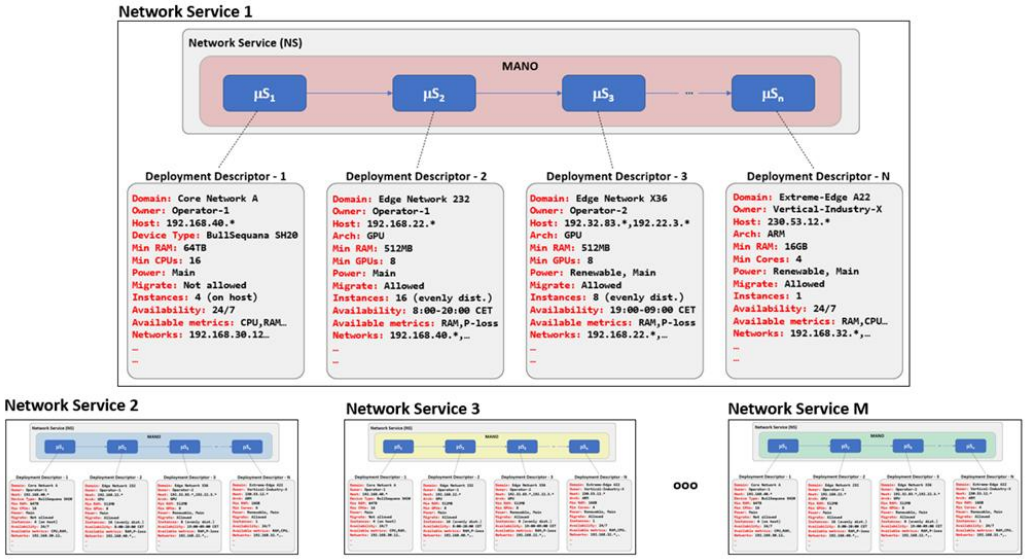


Figure 3-63: Network Service definition and deployment example.

Figure 3-63 illustrates the design approach described in this subsection with an example on how a network service made of different micro-services would be defined and deployed together with the associated MANO functionalities. As it can be appreciated, each NS, made of different micro-services for which their deployment requirements are specified, would be deployed with its own MANO resources embedded, which could be also specific for each NS (represented by the different colours of the MANO blocks). The different Deployment Descriptor blocks in the figure illustrate how a possible data model could be for specifying the deployment requirements, e.g., requiring devices in certain network domains, with certain computing or storage features, the number of instances to be deployed, the kind of power supply, etc.

3.5.2.1.2 Workflow

Figure 3-64 below shows a high-level sequence diagram describing how a NS would be deployed and managed using the decentralised orchestration approach. As it can be appreciated, there is a clear split between the service provisioning stage (steps 1 to 9) and the post-provisioning service management stage (steps 10 to 12), aligned with one of the main concepts associated with this decentralised orchestration approach described above, i.e., to rely on the four common infrastructure management and services provisioning components for the service provisioning (i.e., DN, IRN, ISPN, and SRN), and then, to delegate the subsequent M&O operations on the per-service MANO mechanisms.

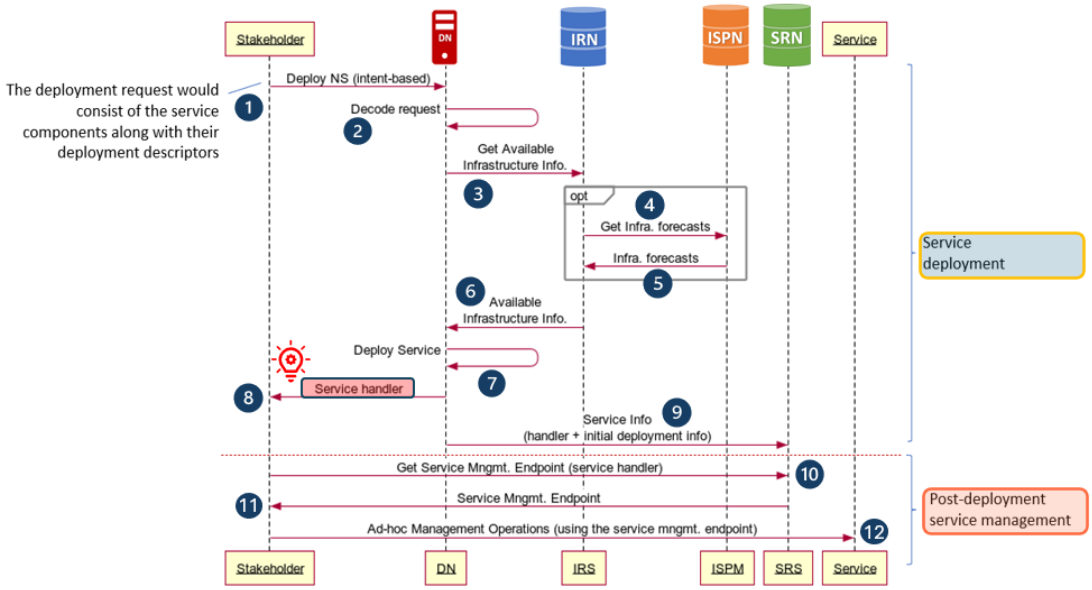


Figure 3-64: NS deployment in the decentralised orchestration approach [HEX223-D32].

It should be noted that the four components depicted in the figure (DN, IRN, ISPN, and SRN) represent four specific components out of all those that could be distributed through the network continuum, and which would not necessarily have to be deployed/operated by the same stakeholder, as represented in Figure 3-62. E.g., the specific DN depicted in the figure could be hosted by a specific MNO, the IRN by another one, and the ISPN/SRN could be hosted by other stakeholders in a distributed approach (the aggregation of all those components would constitute the four common services mentioned above: DS, IRS, ISPS and SRS, which would be accessible to all the stakeholders enabled to deploy and manage network services on the network)[5].

As Figure 3-64 shows, as a first step, a stakeholder (or group of stakeholders) would design the NS, to later request its deployment through the DN. For the service design, this could be done in different ways, e.g., by means of an SDK or a development framework enabling to integrate the different service components, to define their deployment requirements, and once ready, able to connect with the DN to request the deployment. Figure 3-65 shows an idealised representation of a GUI that could be used for this. As it can be appreciated there is a "Service Composition Dashboard" at the bottom, on which it would be possible to aggregate different service components (VNFs in the form of microservices) to compose a NS forwarding graph. Those components

[5] Of course, this is just an example. Other combinations are obviously feasible, including the one in which a single stakeholder may deploy one or more instances of each of these four nodes.

would be selected from the "Multi-stakeholder Components Marketplace" on top, where, besides own components (in red), it would also be possible to select components from third parties (e.g., Vertical Industries, MNOs, Software Providers, etc). As it can be seen, the Service Composition Dashboard would also offer the possibility to edit the deployment descriptor of the microservices, meaning that the service developer could specify here the deployment requirements for each microservice, as explained above (e.g., if it should be deployed on certain subnetwork, on a device with specific computing capabilities, etc). As it can be appreciated, this could be done also "intent-based", using even a natural language interface (bottom right). Once defined, the deployment could be requested from this interface using the "Deploy Now!" button. Of course, that deployment request could be done only by those stakeholders with the appropriate permit level.

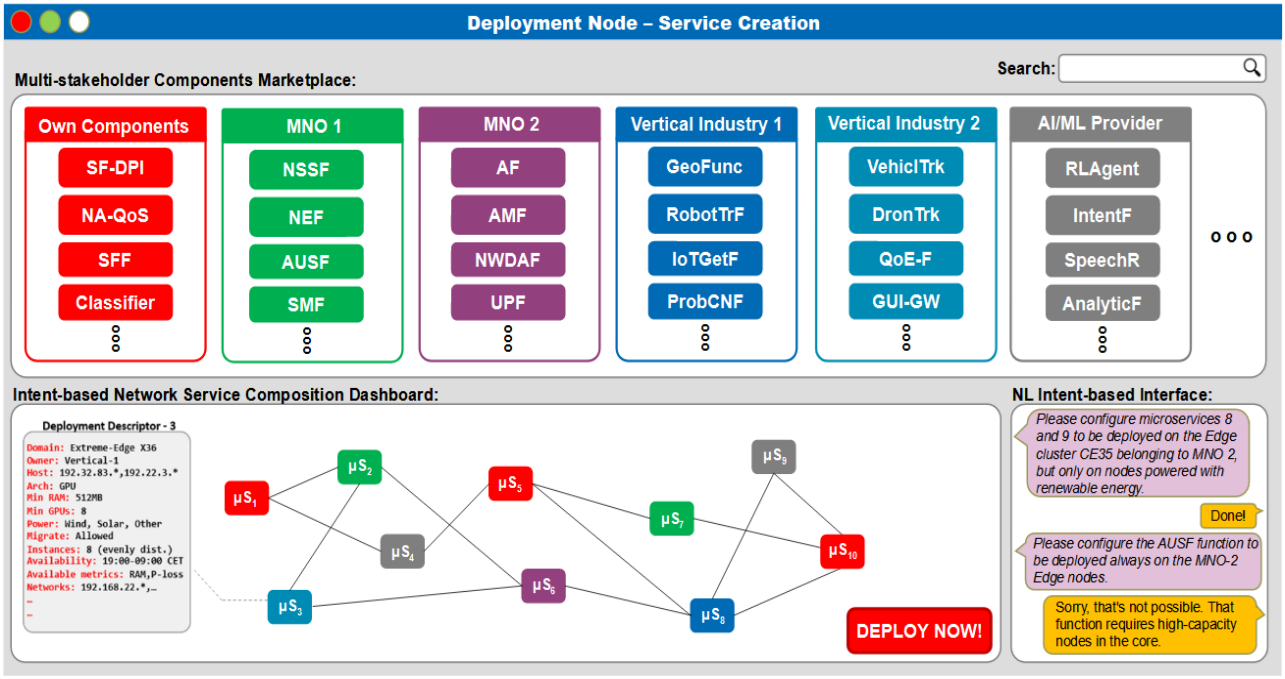


Figure 3-65: Deployment Node GUI example [HEX223-D33].

Indeed, it is not expected that all the design interfaces would offer the same level of complexity as this one in Figure 3-65. In some cases, the approach could be much simpler, e.g., based only on low-level command-line interfaces. The important thing, however, are the main concepts behind it, namely:

- The possibility to compose the network services by aggregating not only own service components, but also other components provided by a diversity of third parties also part of the network continuum (e.g., connectivity providers, data providers, AI/ML providers, vertical industries, etc.), which basically makes the implementation of the microservices federation concept possible.
- The possibility to specify the NS deployment status in a declarative way (either intent-based or not) by using the service components deployment descriptors, i.e., by specifying for each service component just the desired status of the network service once deployed, regardless of how that status is reached.

Following with the sequence diagram in Figure 3-64, once the DN receives the request to deploy the NS, it has to decode that request (step 2) and check if it can actually be performed. For that, and based on the information in the deployment descriptors of each service component, the DN would interact with the IRN to verify whether the requested infrastructure resources are actually available (step 3). As a whole, the DN processing would consist in finding the best-match between the deployment requirements and the infrastructure resources actually available, which basically would be a matter of solving an optimization problem that can be assimilated to the well-known bin-packing problem [Joh73] for VNFs. Also, trust indexes like those computed by sub-enabler 4.3 (section 0 – Trust Management System) could be used as well. Of course, though in the diagram this interaction between DN and IRN has been simplified, in reality it is assumed that the specific IRN to which the DN is interacting is just one of the many nodes parts of the complete IRS, and that would be enabled to provide information on all the diversity of the infrastructure resources in the network. As previously mentioned, such IRS would be a sort of distributed database, with different instances deployed in multiple

geographical locations that would be in charge of collecting and aggregating data from all the infrastructure devices connected to the network. Such information would be distributed through the IRS itself.

Besides, as it can be appreciated in the diagram, the information provided by the IRS could optionally be enriched with the predictions provided by the ISPS (steps 4 and 5), which could be used by the DN to make a more optimal selection of the infrastructure devices on which deploy the service, assigning a higher priority to those nodes for which a more stable behaviour is predicted.

Based on the information provided by both, IRS and ISPS (step 6), the DN would perform the deployment (step 7), and once completed, the deployment process would close the loop by returning to the requesting stakeholder a so-called “service handler” (step 8), to allow the requesting stakeholder (or any other granted) to access the deployed service for any future action. Also, as it can be seen in Figure 3-64, that handler would be communicated to the SRS as well (step 9), which would record it associated to the specific infrastructure nodes on which the service was hosted. That information in the SRS would be automatically updated every time the service components were migrated to other nodes (if that were the case). Of course, if for any reason the service could not be deployed, instead of that "service handler" the user would receive a "deployment error" message explaining the cause of the failure.

This mechanism based on the just mentioned “service handler” is considered a key concept in what regards the implementation of this decentralised M&O approach: since the functionality of the common orchestration system for all services is reduced to provide only the deployment operation, this service handler makes it possible for the stakeholder that requested the deployment of the service (or a group of selected stakeholders) to access the service later on to perform any further MANO operation that could be necessary (e.g., to update certain configuration parameters, access to service metrics, etc.).

For that, as it can be seen in the diagram, the stakeholder would communicate with an SRN using its Service Handler (step 10), which in turn would return the actual endpoint to access the service (step 11). It should be borne in mind that the endpoint may change due to the migration of the service components among the different nodes of the network (hence the need for the SRS service). However, this access to the SRS could be avoided if the service access node would be deployed on a non-volatile node (e.g., in the core network of an MNO).

After the previous, the sequence diagram finishes with step 12, where the stakeholder would execute any service assurance MANO operation that could be necessary. Of course, this could be done as often as necessary, using the Service Handler for that. As it can be appreciated, those MANO operations are not explicitly indicated in the diagram, because they would depend on the specific tailor-made MANO framework embedded in the network service itself (certain services could offer a quite rich interface, while others, with a more lightweight approach, would provide a more limited set of operations). Of course, also, besides those manual MANO operations that could be performed by the stakeholder, other service assurance operations could also be running automatically in the deployed service based also on the specific MANO framework embedded on it (e.g., for regularly collecting metrics, to trigger automatic scaling actions, etc.).

3.5.2.2 Preliminary Implementation and early validation results

The work being performed regarding this sub-enabler is planned to be demonstrated in the context of the PoCs defined in the Hexa-X-II project plan. An initial step towards that is being the implementation of a so-called Infrastructure Layer Emulator (ILE), which in sort, it is a software component that allows to emulate some of the main features of the infrastructure envisaged for the future 6G networks, namely:

- The deployment of a large number of computing nodes (the 6G orchestration systems should be able to operate on the complete network continuum, which is intrinsically large in size).
- The possibility to integrate different types of computing nodes (besides large, the network continuum is also heterogeneous).
- The emulation of different stakeholders.
- The emulation of the different network domains in scope, i.e., core, edge, and extreme-edge domains.
- For the extreme-edge resources, the emulation of its inherent high-volatility, with devices unexpectedly connecting/disconnecting and/or changing certain properties (e.g., memory or CPU occupancy, battery level in the case of battery power devices, etc.).
- The possibility to deploy realistic network services on such emulator.

This ILE can be seen as an implementation of the Infrastructure Layer in Figure 2-2, covering the whole network continuum. Its development is hence considered of paramount importance in order to have a kind of realistic environment in which to test and demonstrate the design concepts for the future 6G M&O systems. Although still under development, an early version is already available while this document is written, with most of the features described before. This early version has been released as Open Source, under the Apache 2.0 license (https://gitlab.com/decentralized-continuum-orchestration/infrastructure-layer-emulator). Its implementation is based on the LXD [LXD], a system container and virtual machine manager that provides a unified environment for running and managing full Linux systems inside containers or virtual machines. It supports images for many Linux distributions, from lightweight distributions (which are quite useful to be able to deploy a large number of Linux nodes on small-scale equipment) up to regular/complete Linux distributions (e.g., able to host and execute complex network services). LXD can scale from one instance on a single machine to a cluster emulating a full data centre, making it suitable to simulate different kind of scenarios.

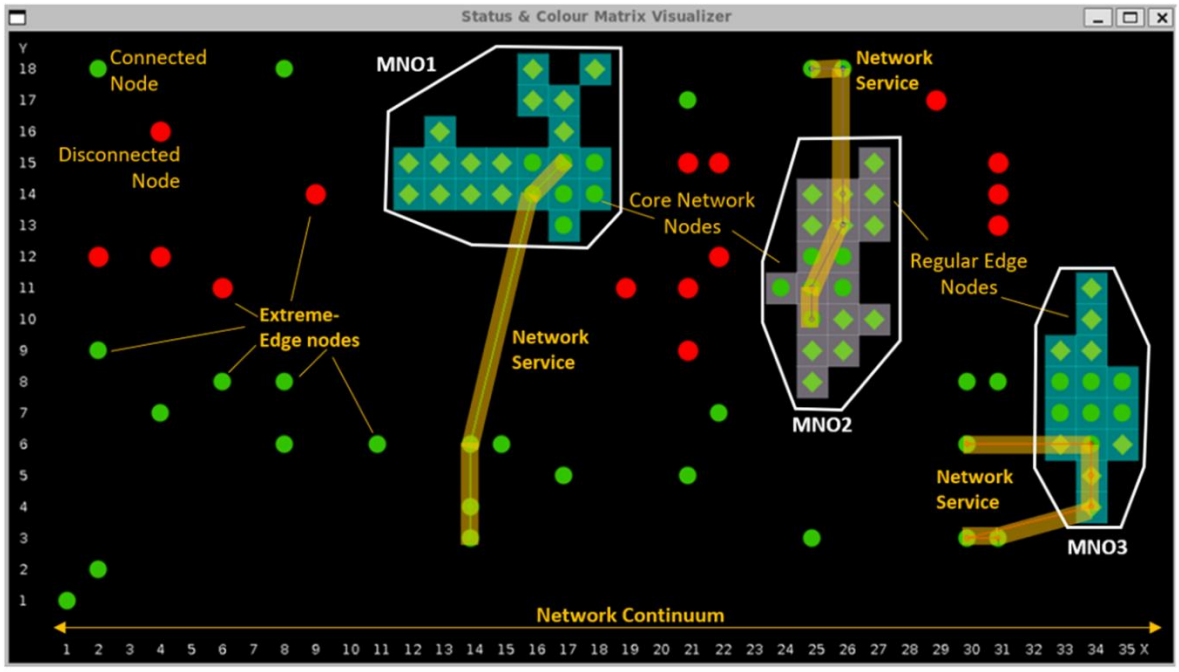


Figure 3-66: Infrastructure Layer Emulator (ILE).

Figure 3-66 shows a screenshot of the ILE, showcasing a set-up with three MNOs (white polygons) with core and edge resources each (circle and diamond nodes), and a set of extreme-edge nodes beyond the MNOs domains. Green/red circles and diamonds within and without the MNOs represent "actual" (containerised) Linux machines able to communicate each other and host network services. In the live set-up those nodes are shown asynchronously "blinking", i.e., switching between green and red depending on its availability status (connected or disconnected). As can be seen, three services are also shown deployed, represented by the three thick lines connecting certain nodes, and covering infrastructure resources withing and without the MNO domains. In the live PoC these lines evolve dynamically depending on the connection/disconnection status of the nodes at the extreme-edge, representing the service components' re-location in real-time.

Of course, the integration of the functionalities provided by this sub-enabler in the PoC goes beyond the ILE itself, i.e., the ILE is just a tool that is considered needed to demonstrate the decentralised orchestration concept. Based on that, the approach is twofold:

a) On the one hand, to demonstrate that it is actually feasible to deploy (on the ILE) a realistic network service with the service assurance MANO mechanisms embedded on it, i.e., as described in 3.5.2.1, and to demonstrate the functionality of those MANO mechanisms.
b) On the other hand, to provide an initial implementation of the functionality provided by the common infrastructure management and services provisioning system described in the previous sub-sections.

Regarding (a) initial positive results have been already obtained integrating the K3S microservices orchestration system [K3S] as part of a simple NS, and relying on the K3S high-availability feature to deploy

a new microservice instance at runtime when an instance that was running on an extreme-edge node becomes unavailable due to the unexpected disconnection of that node. However, the work continues to improve this scenario targeting the demonstration with a more realistic network service (the low latency network service developed for PoC#B.1), and also, to trigger the orchestration action (i.e., the relocation of the network service components) based also on energy-related measurements (certain network nodes will be configured as "battery-powered", and the VNF migration will be triggered based on the measured battery levels).

Regarding (b) the focus is on the implementation of an Infrastructure Status Prediction Node (ISPN) based on AI/ML techniques, in order to provide predictions about the extreme-edge nodes availability status (those deployed in the ILE). This ISPN will provide forecasts to the service referred in (a) to trigger proactive orchestration actions, i.e., to make K3S to be able to migrate certain service components "before" the specific node on which they would be deployed is actually unavailable. While writing this document an initial design for this has been already provided, though the prototype is still under development.

3.5.2.3 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this sub-enabler 5.2 are the following:

- **Scalability**. As a distributed system, the approach presented here is intrinsically highly scalable, which is considered quite relevant regarding the integration of the extreme-edge domain. The distributed orchestration network elements make it possible to handle an increased amount of network services and workloads of different shapes and sizes, and without requiring complex centralized systems that could become bottlenecks or single points of failure, and that would need to be upgraded to be able to manage more services and resources.
- **Latency**: The proposed approach relies on deploying service components on the whole network continuum, beyond the MNO own domain, and relying on network resources from multiple datacentres and in different geographic regions, so making it possible to move the necessary service components in close proximity to where they are actually requested by end users. This can lead to a large reduction in latency for certain time-sensitive applications, beyond to what can be done in 5G with regular edge nodes.
- **Flexibility**, mainly in what regards the integration of vertical parties that, instead of having to adapt to an external MNO-centric orchestrator could just integrate their own service components exposing their interfaces in a cloud native way. Also, in what regards relying on an extensive diversity of infrastructure resources, which would be abstracted by the DN.
- **Processing Capacity**, which would be considerably expanded by integrating the resources at the extreme-edge domain.
- **Automation**, which appears in different aspects of the model: the devices discovery processes and the registry of the infrastructure resources performed by the IRS would be a highly automated process. Also, the migration of the NS components through the volatile extreme-edge infrastructure nodes (which can be also performed proactively, based on the ISPS predictions). Also, the deployment of the network services from the DN: finding the most suitable infrastructure resources to deploy the network services (according to the requirements in the deployment descriptors) would be fully automated as well.
- **Services Creation Time** is a KPI that would be of course affected by this approach, depending on the facilities and mechanisms provided by the DS.
- **Integrated intelligence**. AI/ML algorithms could be embedded as part of the ISPS to enable proactive M&O actions. Also, as part of the service specific tailor-made MANO resources for the services assurance processes.
- **Reliability**. The sub-enabler has been designed to specifically target the high-volatility of the resources extreme-edge domain, considering that those resources could unexpectedly vary their capabilities, move or even fully disconnect, and providing a M&O framework to specifically manage such situations. This should therefore affect the reliability of the system as a whole, and of the deployed network services.
- **Programmability**. The proposed system is highly programmable by itself, relying on the cloud-native principles as a whole (e.g., all the network service components communicate using exposed interfaces, which enable programmability, and could be provided relying on highly automated DevOps practices).
- **Maintainability**. The system is designed to be easily maintainable, in what regards the on-boarding/off-boarding of the infrastructure resources, which would be done in a highly automated manner supported by the infrastructure discovery mechanisms associated to the IRS.

- **Intent expressiveness**. As anticipated in [HEX223-D22], network services definition could be performed intent-based. Intents, that could be expressed even in natural language, would be translated into the low-level regular language used to define the service components deployment descriptors.
- **OPEX** would be reduced for MNOs: instead of a large and complex M&O system within their premises to manage a big amount of network services and infrastructure resources, they could delegate on a wide set of external distributed resources and M&O mechanisms.

Regarding KVIs, it is considered that the most evident to be affected by this sub-enabler is **Sustainability** in what regards the following aspects:

- Extreme-edge nodes that would be already connected (so consuming Energy) but not hosting any service could be used to execute network service components, which would avoid having to put into operation new infrastructure nodes which would consume an additional amount of energy. Although the amount of energy consumed by the execution of the service components could be considered roughly the same in both cases, the baseline energy required to keep the nodes up and running would be saved. Considering the large size of the extreme-edge domain this could be a great energy saver.
- Relying on the extreme-edge devices to deploy network service components would also contribute to reducing the hardware to be deployed by MNOs (or other stakeholders) in their datacentres. If certain network service components would be deployed on extreme-edge devices that are "already there" (e.g., vehicles, industrial robots, home appliances, etc.), that could contribute to reduce the amount of specific infrastructure devices in datacentres, which would lead to direct energy savings, as well as to reduce the carbon footprint associated with the manufacturing and the installation of those devices.
- Energy used in data transmission would be also reduced, since certain workloads could be directly executed on edge and extreme-edge resources, without needing to transmit certain data to central datacentres (e.g., inference or certain training AI/ML algorithms could be performed right on the extreme-edge nodes).

3.5.3 Sub-enabler 5.3: Federated orchestration system

As the operating landscape of cloud networking becomes more disaggregated and more resource providers enter the market, it becomes more complex to establish SLAs to provide service continuity among them should the need arise. Moreover, given that it makes more business sense for the providers to modify the pricing for the use of their resources depending on demand, SLAs need to be dynamic to capture this characteristic.

Using technologies like NFV and Network Slicing enables the creation of vertical-specific services tailored to meet the unique requirements of each industry sector. Vertical-specific services are transformed into network services (NFV-NS), which are then deployed and orchestrated across both local and external domains. The orchestration of services across multiple administrative domains known as federation, presents a promising opportunity for future 6G networks. Specifically, federation of services occurs when a consumer domain instantiates NFV-NSs (or parts thereof) in external provider domains, thereby orchestrating their life cycle. A promising approach to tackle these issues is leveraging distributed ledger technologies or blockchains to define and execute smart contracts. This system ensures security, transparency as well as verifiability without the need of costly and slow third-party intermediaries.

3.5.3.1 Enabler Design

The key idea involves employing a permissioned blockchain where each administrative domain operates a single node within the blockchain. A unified Federation Smart Contract (SC) is deployed on the blockchain to serve as a decentralized authority, ensuring security and trust throughout the federation process. Each administrative domain running a single node operates the same instance of the Federation SC, ensuring synchronous execution of code across all nodes in the permissioned blockchain network, bolstering security and trust among participants.

The design of the Federation SC is pivotal in safeguarding the privacy of sensitive information for each administrative domain while overseeing federation procedures involving all domains. To join the blockchain network, a new administrative domain must register with the Federation SC, providing its unique blockchain address and administrative information along with its service footprint.

For seamless interoperability, administrative domains access the Federation SC via a REST API. Once registered, domains can participate in federation processes as consumers or providers. When a consumer domain initiates an announcement or federation offer, it is recorded as a new auction on the blockchain by the Federation SC, which then broadcasts the auction to all registered domains. To protect privacy, the consumer domain's address is concealed in the broadcast announcement, preventing passive data collection by other domains. As such, a single-blinded reverse auction mechanism is employed, where consumer domains anonymously create offer announcements, and potential provider domains bid for them.

The Federation SC serves as a facilitator rather than an authority, allowing the bidding process to be closed only by the consumer domain. This empowers the consumer domain to apply selection policies freely. Periodically polling the Federation SC, the consumer domain caches bidding offers, selects a provider domain, and closes the auction. The chosen provider is recorded as the winner by the Federation SC, which notifies the selected provider and broadcasts the completion of the auction to all domains. Following this, the negotiation and acceptance phases conclude.

Direct communication and information sharing occur between the consumer and selected provider domains. The federated service is deployed and integrated into the E2E service by the consumer domain, following legacy provisioning procedures. Upon deployment completion, the provider domain initiates charging for the federated service. The same permissioned blockchain network can be utilized, incorporating micropayment channels to facilitate unbiased charging records, immutable for both consumer and provider domains.

3.5.3.1.1 System components

This enabler leverages the monitoring and telemetry capabilities of Enabler 2 to obtain the data that serves as part of the input to trigger the smart contract execution. It is also closely aligned with the objectives of user-centric service provisioning given that it allows for dynamic SLA creation and enhances service continuity. By eliminating the need for a human in the loop for the business process of SLA creation and policing, this enabler contributes to the real time zero touch control of Enabler 8.

Figure 3-67 illustrates the design of the Digital Ledger Technology (DLT) federation system. Each cloud domain includes a node that implements a blockchain protocol that allows it to participate in the DLT network.

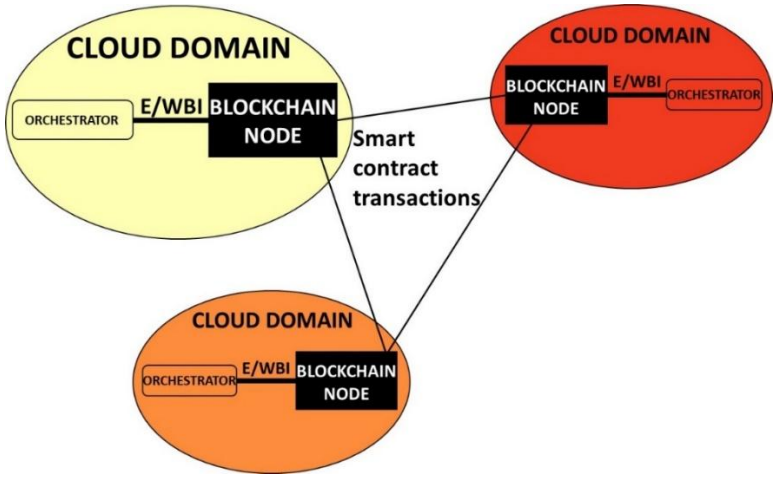


Figure 3-67: S/W Design of DLT based federation.

The resources, required to provision a service, of each cloud domain are managed by an orchestrator which determines where in the domain's network they will be deployed. An East-West Bound interface ensures that the orchestration mechanism is governed by the DLT by enabling it to connect to its corresponding domain's node that participates in the blockchain network. The smart contract consists of a set of conditions (based on the ad-hoc SLAs between the consumer and provider domains) that when met, trigger the federation process highlighted in Figure 3-68.

3.5.3.1.2 Workflows

In order that this proposed system works as desired, the available providers have to be first registered on the blockchain. When the need arises to exploit resources of a provider distinct from the consumer domain currently offering a service, it publishes an advertisement on order to discover the candidate provider domains.

This is followed by a negotiation of terms using such techniques as reverse auctioning. Once an agreement is reached, it is announced to all the bidding provider domains and the chosen one(s) proceed to provision the resources to deploy the service. The service is then deployed on the provider domains and the corresponding life-cycle management including the records for subsequent billing carried out by the consumer domain in a transparent manner over the blockchain. The sequence of the federation mechanism is depicted in Figure 3-68.

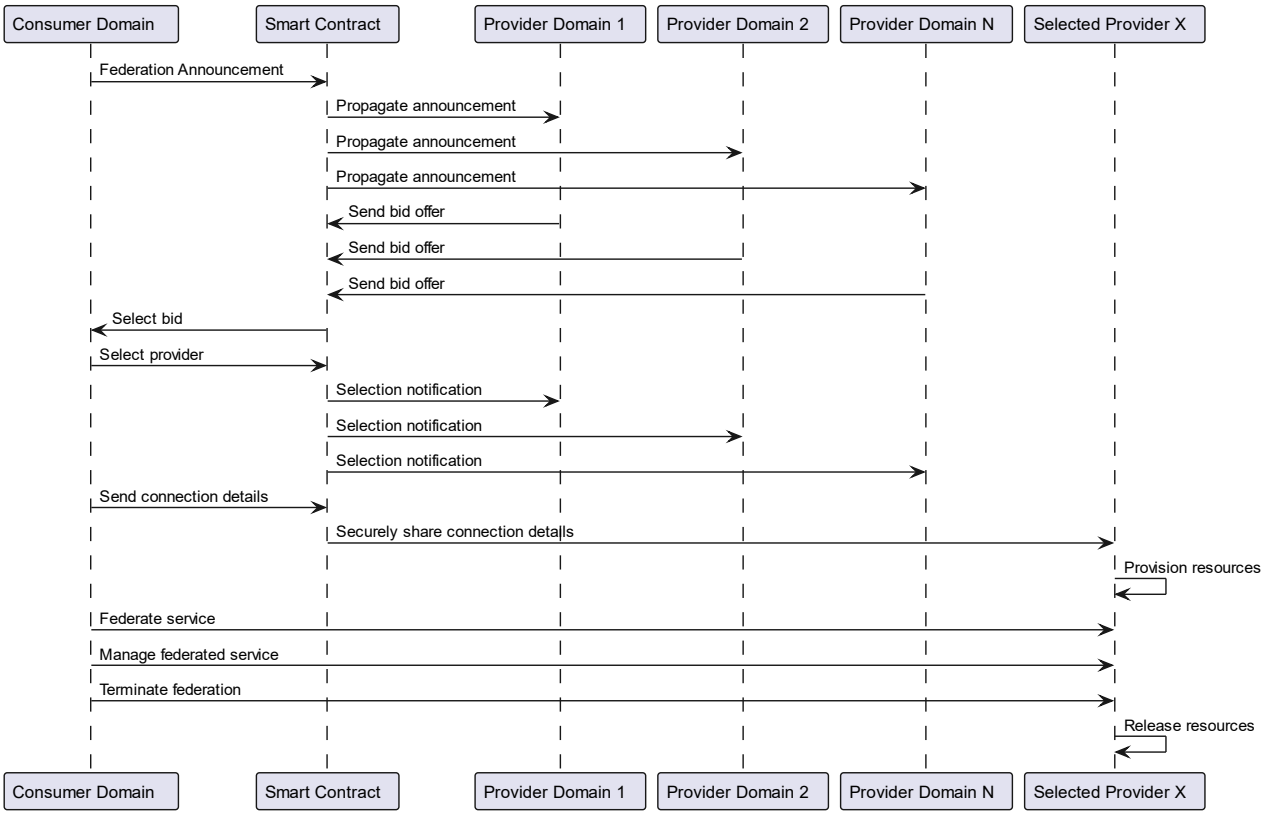


Figure 3-68: Sequence diagram of service federation using DLT.

3.5.3.2 Preliminary implementation and early validation results

A deployment of a preliminary version of the federation system using KVM as the VIM and Kubernetes as the orchestrator is depicted in Figure 3-69. Ethereum [ETH] has been used as the base DLT network allowing the use of smart contracts for triggering and managing the federation process. This implementation is aligned with PoC B.1 and facilitates the autonomous provisioning of the object detection service in third party domains.

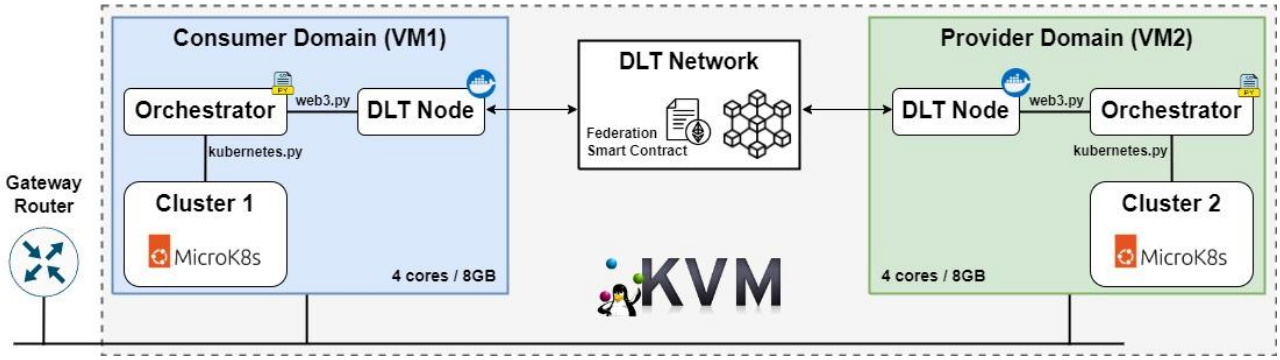


Figure 3-69: Initial implementation of DLT federation.

The consumer initiates a transaction in the smart contract to request the deployment of a desired service with specific requirements. The provider monitors federation events and responds by creating a bid-offer transaction that includes the service price... The consumer then goes on to select the winning provider which proceeds to deploy the requested federated service. Once the deployment is successfully completed, the consumer is informed by the provider and it furnishes the latter with pertinent service information, such as the connection

parameters. A load balancer is employed to allow the deployed service to communicate to the consumer domain.

These data were collected by recording the precise timestamps of transaction times as perceived by the consumer domain. Figure 3-70 depicts the average duration of the federation workflow steps. The transactions between the consumer and provider domain take longer than the service deployment which is executed only on the provider domain. The foregoing is as a result of the complex cryptographic mechanisms associated with blockchain transactions that underpin the authenticity, integrity and confidentiality of this technology.

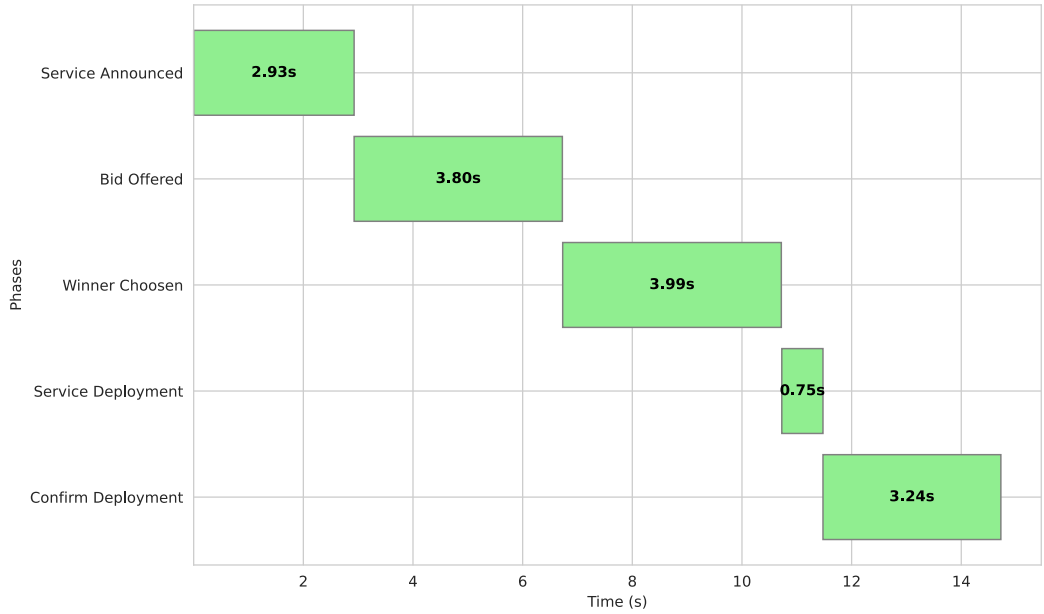


Figure 3-70: Average duration of service federation steps.

3.5.3.3 Impacted KPIs and KVIs

The main KPIs considered impacted by this contribution are:

- **Reliability**: The autonomous provisioning of a service in third-party domains ensures service continuity in the event of unforeseen roaming scenarios which in legacy systems would necessitate services to be dropped.
- **OPEX reduction**: Given that this system provides a trustworthy mechanism through which service level agreements can be reached, executed and policed without a costly intermediary legal framework, the operational costs of network operators is greatly reduced.
- **Scalability**: This mechanism is highly scalable given that all that is required for operators to participate in this scheme is a representative node in the blockchain, they need not establish prior agreements with every possible third-party provider as in traditional systems.
- **Automation:** This scheme fully automates the legacy BSS based SLA establishment process and integrates it into the OSS processes related to service provisioning.

3.6 Enabler 6: AI/ML algorithms

This enabler focuses on providing AI/ML-based mechanisms for the M&O to be included in the E2E system blueprint design. This enabler is split into two sub-enablers, the first one is the AI/ML-based control algorithms for sustainability which provides AI-based solutions that fulfil the objectives of improving the overall performance in terms of QoE metrics, energy efficiency, and increased zero-touch automation for reducing OPEX. The second sub-enabler is aims to improve the trustworthiness of the AI/ML based control system by protecting the AI/ML models against privacy and adversarial attacks, and by providing explainable decision outputs. The solutions developed within the enabler are part of the 6G architecture blueprint as a set of AI/ML-based solutions within the AI/ML framework for automated, sustainable and performant network control, together with the security, privacy and explainability mechanisms to ensure the trustworthiness of the system

and its decisions. The solutions can be deployed in a distributed manner, having different AI/ML models running and inferring close to or even as part of specific M&O functions or procedures. These procedures can also be part of a single layer seen in the blueprint Figure 2-2, from the network-centric application down to the actual computing infrastructure, or any combination of layers with resulting actions being performed in multiple layers. For example, a Reinforcement Learning policy for function placement and scaling can be part of an orchestrator in the network layer deploying and managing those functions.

3.6.1 Sub-enabler 6.1: AI/ML-based control algorithms for sustainability

In recent years, an increasing interest has been witnessed for environmental sustainability and energy efficiency, which have become one of the main goals for future network technologies. This is particularly relevant considering increasingly complex network environments that include edge resources and virtualization overlays that make it difficult to select the appropriate management action, in particular when energy saving management actions may conflict with network and service performance targets, or with other energy saving actions. In this context, this sub-enabler aims to provide AI/ML-based control solutions that help achieve 6G networks requirements in terms of environmental sustainability and energy-saving while satisfying performance targets. The solutions developed within this sub-enabler provide AI/ML-based network control solutions with the objective of optimizing energy efficiency, to be integrated into automated network and service management procedures. Namely, energy-efficient solutions are provided to perform dynamic resource allocation for service chains and multi-domain system federation, where the decision process optimizes network and service performance metrics such as latency as well as energy related indicators such as energy consumption, carbon footprint, and energy sources. To facilitate network management automation, AI/ML is employed to translate high-level energy-related requirements into decision recommendations. On the other hand, while using powerful ML models can improve efficiency, the training process of those models consumes significant amounts of energy and computing resources. Thus, sustainable AI/ML solutions are developed where the energy consumption of AI/ML model training is monitored, and a trade-off is made between performance and energy consumption when implementing AI/ML based solutions.

3.6.1.1 Sub-enabler design

This sub-enabler aims to provide AI/ML-based solutions for automating and optimizing M&O control in terms of sustainability and energy efficiency, while maintaining the service performance levels. Thus, this sub-enabler supports the different M&O lifecycle workflows such as service deployment, task scheduling, migration and scaling by providing the AI/ML models for making automated (zero-touch) decisions. The MLOps workflows for training, deploying and managing the AI/ML models are also supported and optimized to reduce energy consumption. Additionally, this sub-enabler interfaces with the monitoring functionalities from Enabler 2 to collect relevant data on energy consumption and service performance metrics such as latency and throughput.

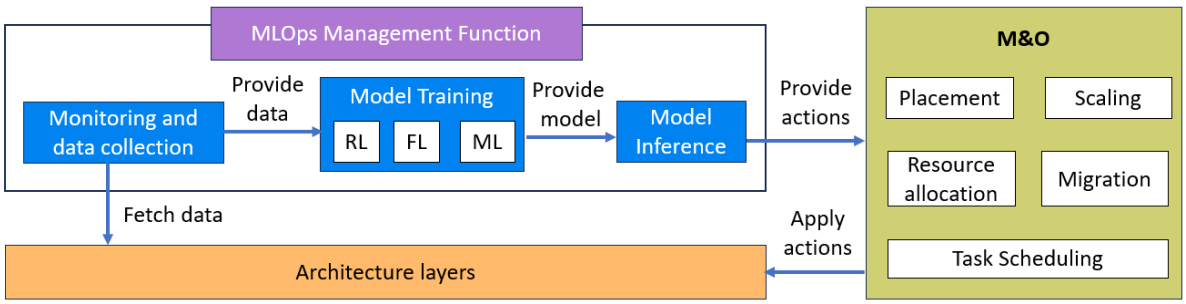


Figure 3-71: High-level architecture for AI/ML-based M&O.

The high-level architecture for this sub-enabler is illustrated in Figure 3-71 and designed as follows:

- **Monitoring and data collection:** The monitoring functionality interfaces with different layers and components of the architecture and collects data on the state of the network infrastructure and services. The collected metrics considered relate to service KPIs such as latency and throughput, as well as

energy consumption values. The data is provided as input to the AI/ML model to make decisions, as well as for evaluating the results of the applied decisions.

- **Model training:** This functionality is responsible for the training of AI/ML models for performing automated M&O actions such as service deployment, migration, or scaling while optimizing energy efficiency as well as different performance metrics such as latency and throughput. The model training may be centralized on a single instance, or distributed over multiple instances or domains such as in Federated Learning.
- **Model inference:** This component is responsible for performing inference of the trained models, and providing control actions to be performed by the M&O system.
- **MLOps Management Function:** This functionality is responsible for orchestrating the lifecycle of the AI/ML model, and may trigger model training, data collection, and deployment on the inference instances. This functionality also aims to monitor and minimize the energy consumption of the steps in the MLOps control loop of the models.

3.6.1.2 Preliminary implementation and early validation results

Multiple implementations for the sub-enabler are described below together with early validation results in the following sub-sections, where the implemented solutions aim to minimize energy and resource consumption while satisfying performance metrics. Six implementations are detailed:

1. AI/ML-based recommendation system for access node configuration.
2. Resource-efficient network function deployment in shared Edge environments.
3. Energy efficient computational workload placement.
4. Energy consumption assessment and optimization for MLOps workflows.
5. Federated Learning for decentralized resource allocation in multi-domain collaborative architectures.
6. Multi-agent Reinforcement Learning for dynamic scaling of resources.

All of these except 4 and 5 are different AI/ML models that fit in the blue elements of Figure 3-71, implementing (part of) the intelligence for different M&O procedures. The other two detail improvements to the MLOps Management Function with regards to energy and other costs related to model training and model aggregation across domains.

3.6.1.2.1 ML based configuration recommendation for energy saving

Despite the numerous energy-efficiency solutions already integrated into mobile networks, energy consumption continues to escalate due to the rapid expansion of both network traffic and data volumes. Research indicates [CKS+21] that further enhancements in energy efficiency can be realized by leveraging ML techniques, facilitating higher levels of automation. Through the recommendation of configuration settings applicable to base stations and other equipment, ML-based techniques enable the reduction of energy consumption in network elements without adversely affecting Quality of Experience (QoE). Through the recommendation of configuration settings applicable to base stations and other equipment, ML-based techniques enable the reduction of energy consumption in network elements without adversely affecting Quality of Experience (QoE).

The focus on improving energy efficiency in networks revolves around access nodes. An access node, often referred to simply as a node, represents the relationship between connected user devices and the network elements to which those devices are linked. Configuration settings for access nodes significantly impact node energy consumption [PLD+22] and potentially influence numerous observable network performance Quality of Service (QoS) metrics.

Recognizing that certain configuration settings may have varying timeframes for implementation, the necessity arises for accurate predictive models. These models should forecast when changes can be applied in advance, aiming to minimize potential disruptions to the network's operation. An optimal solution would identify numerous potential avenues to reduce energy consumption within the current functionality of network elements.

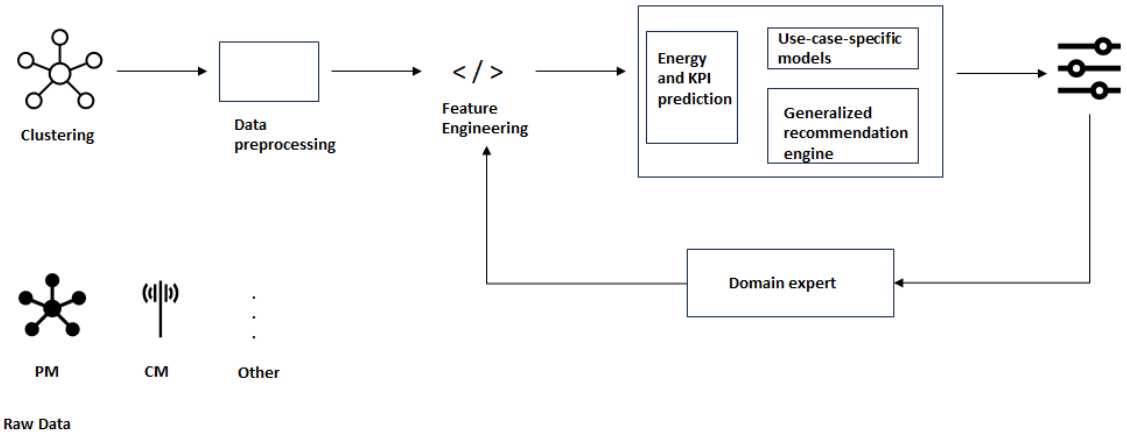


Figure 3-72: End-to-end energy optimization from power system to node to network [VHI+21].

Considering these aspects, a concept for E2E energy optimization has been formulated, spanning from the power system to the nodes and the network level. Figure 3-72 depicts the concept, emphasizing the energy recommendation engine that serves as its core [VHI+21].

Implementation

All data was derived from a live network The primary sources included performance management (PM) and configuration management (CM) data obtained from the base station. Energy measurements are inherently part of the PM-collected dataset, eliminating the need for deploying new hardware to gather the required information. The radio network performance counter data sets offer insights into cell performance, encompassing metrics like activity count in downlink (DL) and uplink (UL) directions, cell utilization, and units. Example of data collected includes transmit power of the base station, total energy consumption, important key KPIs such as throughput and latency. The environment is a real network with different types of traffics. As depicted in Figure 3-72, the collected data is first pre-processed to be ready for feature selection. Not all features are important for energy and KPI prediction and hence some of them are eliminated with feature engineering. Then, the solution described in the next Methodology section is applied to the final data and features where it recommends new network configurations, such as new transmit power level, and it also provides the prediction of this new recommended configurations on the network KPI, such as throughput or latency. Domain expert can also play a role in feature selection and in final network configuration to be applied to real network.

To ensure no adverse impact on QoE, the models were constrained by KPIs. While additional or alternative KPIs could technically be included based on operator preferences, the five of them are selected considering their influence on energy consumption: Number of connection attempts to a cell, Average number of users in a cell, Throughput, Latency, Interference.

Telecom networks are initially configured with parameters such as the number of cells and hardware unit types, but reconfigurations may occur due to issues like software glitches or hardware failures. Subtle changes and tuning over time can lead to varying energy consumption levels, either positive (less energy consumption) or negative (more energy consumption) for the same traffic volume.

The CM data set that is utilized comprises numerous configuration attributes for the sector, including radio cell settings (e.g., frequency in DL and UL directions) and installed hardware types. This information enables the recommendation of multiple configuration changes simultaneously, as opposed to focusing on individual adjustments. The output from the energy recommendation engine provides a set of configuration attribute changes for a corresponding node, capturing the interplay between different nodes and configurations rather than isolated fine-tuning at a per-node level, which could impact other nodes.

Methodology

The high-level definition of a generative model involves deducing the generalized distribution of observation features within the constraints specified. The recommendation engine, based on a Conditional Variational Autoencoder (CVAE) generative model [SLY15], comprises multiple components, including an encoder, decoder, prediction model for target KPIs, and a prediction model for the energy consumption target (see

Figure 3-73). The encoder model compresses the representation of the raw CM dataset into a low-dimensional matrix known as the latent space. This space consists of points, where each point in a 2D latent space represents a complete CM configuration setting constrained by energy consumption and a KPI value.

Given the nature of the CVAE, CM configurations with the same energy and KPI constraint categories are situated closely in the latent space. The decoder reconstructs the complete configuration file, potentially containing numerous CM attributes, from the embedded representation in the latent space. Latent variables representing the targeted KPI and energy-consumption levels are randomly selected from those representing the same category of targeted KPI and energy levels. The decoder utilizes these inputs to generate new configuration attributes.

During deployment, constraints are provided as input to the CVAE model's decoder, confined by the specified energy-efficient network parameters.

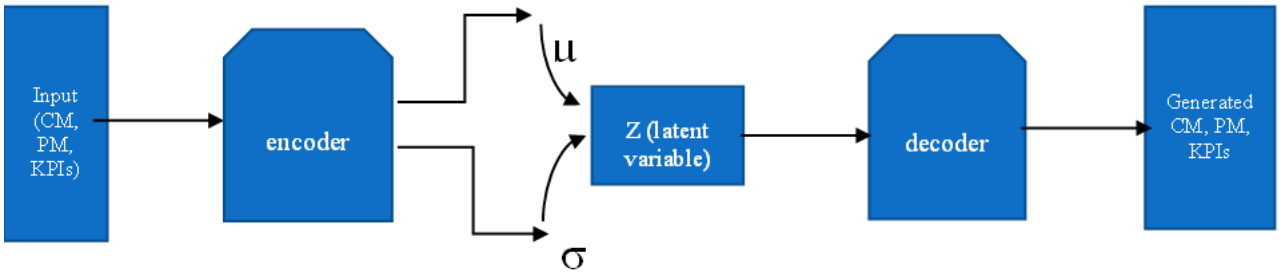


Figure 3-73: CVAE Structure.

Experimental results

By applying CVAE, a KPI versus energy trade-off curve is obtained as shown in Figure 3-74. In general, these curves tend to show that higher energy savings yield a higher negative impact on the KPI. For example, the operator can select a configuration from this map, depending on its priority. As the impact of different configurations to different KPIs is measured, the operator can decide that the KPIs coming with more valuable services are favoured, and the configuration reflecting this priority is selected.

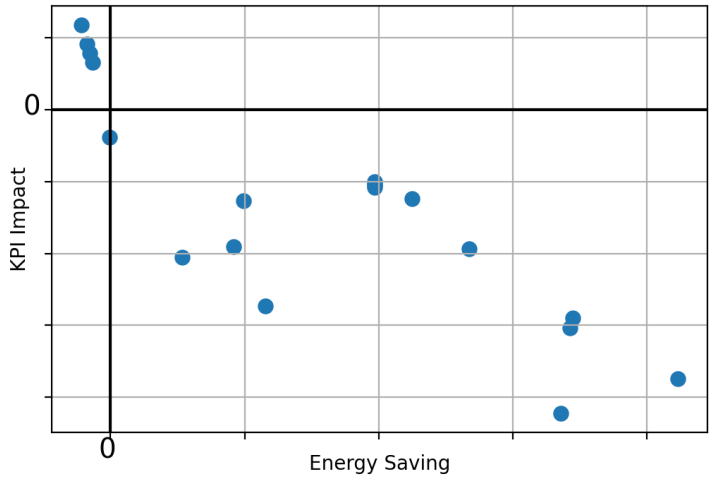


Figure 3-74: Energy savings and KPI impact according to the CVAE model.

An all-encompassing strategy for energy optimization is the most effective means to attain comprehensive energy savings. This approach guarantees that advancements achieved at one level do not counteract increased energy consumption at another level. The concept developed here for E2E energy optimization relies on an energy recommendation engine fuelled by artificial intelligence. This solution holds significant potential for automation, facilitated by specific interfaces capable of directly adjusting nodes without human intervention. Moreover, it can be entirely software-based, eliminating the need for additional hardware. From the blueprint architecture point of view, the idea can sit in Pervasive Functionalities block where it can be placed at AI function module. It can communicate RAN NF through Network Layer APIs located in M & O. For example,

it can be deployed as rApp in O-RAN architecture and can communicate with other rApps or different NFs (NWDAF) to cooperate or to obtain the required information or data.

3.6.1.2.2 Resource efficient network function deployment

Edge Computing enables the deployment of services closer to the end users, which reduces the communication latency, offloads the core, and allows more efficient resource management. Thus, services that are traditionally deployed on centralized servers can now be distributed over multiple Edge servers and their allocated resources can be adapted on-demand to adjust to local requirements. In this context, the same compute infrastructure can be used to host network functions and services whenever possible. To further optimize physical infrastructure usage, the physical resources of a node can be shared between the services and functions that the node hosts based on the current demand without a strict allocation scheme. However, this leads to the hosted functions and services competing for the available resources, and possible SLA violations in case of increased user demand. Therefore, an efficient resource allocation process should find a trade-off between user admission and SLA satisfaction. The solution described below allows to train a Deep Reinforcement Learning (DRL) [ADB+17] model for dynamic and intelligent network orchestration with resource allocation, user demand admission and routing in a resource-constrained edge environment with shared resources, where the objective is to maximize user demand admission and SLA satisfaction per user, while minimizing resource and energy consumption by performing service consolidation.

The solution developed in this work allows to train and provide a model for resource efficient network function deployment in shared Edge environments, the model is assumed to be trained by the AI/ML framework, and used in a centralized manner by the M&O control functionality of the architecture. This model would then be deployed for inference as illustrated with the blue elements from Figure 3-71, providing actions to the placement and resource allocation functionalities.

Implementation

As illustrated in Figure 3-75, the developed solution consists of an architecture comprising a DRL model which has been trained using the Deep Deterministic Policy Gradient (DDPG) algorithm [LHP+16], an actor-critic algorithm that comprises an actor for providing actions, and a critic for evaluating the decisions provided by the actor.

The environment for training the model is developed using a network simulator (OMNeT++) [OMN24] for deploying services on a network infrastructure comprising 5 interconnected Edge clusters of 10 nodes each with varying user demands, and observing the packet delay and drop for each of the user flows. Multiple scripts have been developed for integrating the model and the network simulator by generating the state input to be fed into the DRL agent, translating the DRL agent actions into simulation configuration and deployment files to be run by the network simulator, collecting the flow metrics from the simulator and calculating the overall reward to provide to the agent.

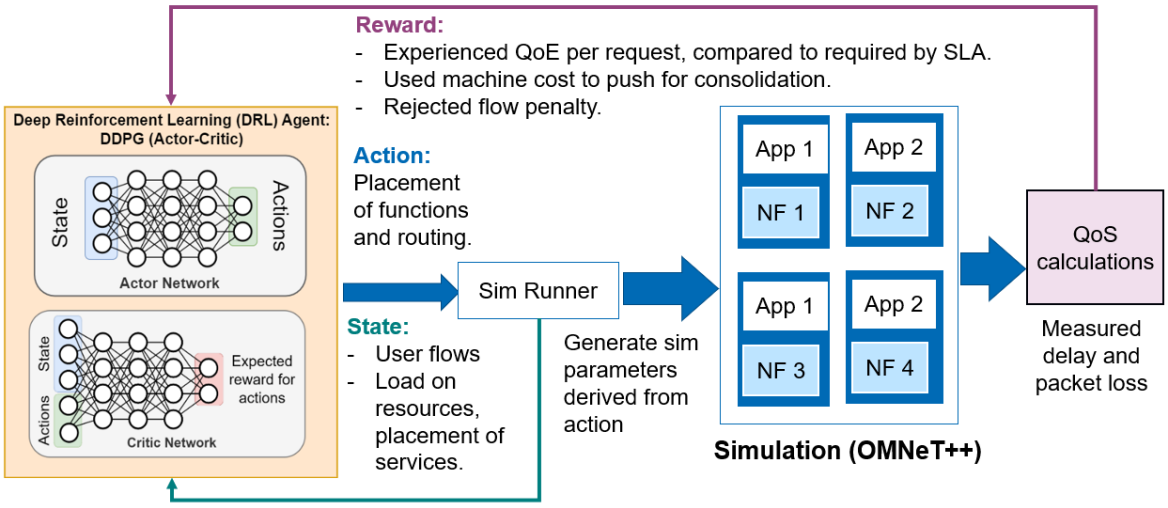


Figure 3-75: DRL model training environment.

To train the DDPG agent to learn a decision policy for energy efficient resource allocation, the following state, actions, and rewards were formulated:

- **State:** Includes information on the current user demands for services and their location, as well as the state of the network with information on the current load on computing and network resources, and the placement of the currently deployed services.
- **Actions:** The agent provides the new placement of services for the next step, which can be translated into the deployment of new instances, their migration to a different server, or perform scaling-out (adding a new instance and performing load balancing) or scaling-in (removing an instance). Additionally, a user admission and distribution matrix is provided for distributing the user demands coming from each edge over the instances deployed on the edges. User demands that have not been routed to any edge are rejected.
- **Rewards:** The rewards are calculated based on the QoS metrics per flow, where the packet delay and drop are compared to the maximum values allowed by the SLA requirements of the service. If they breach the SLA, the penalty for packet loss increases linearly with the loss while the one for delay increases with the cube of the experienced delay. Additionally, to push for consolidation and energy efficiency, a penalty is added for each server where an instance is deployed. There is also a penalty for each rejected user flow.

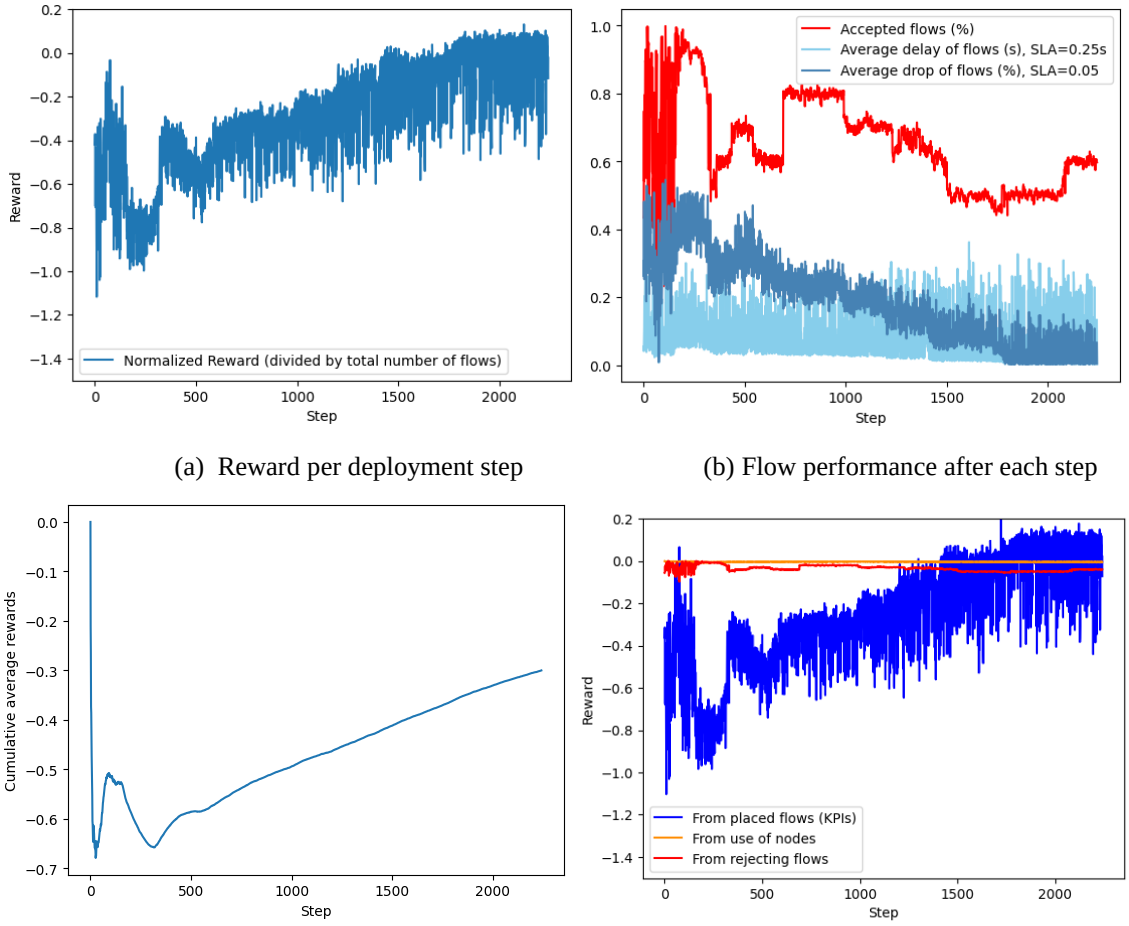
The model training process includes testing multiple hyperparameter combinations such as the learning rate and the DNN architecture parameters (e.g. layer number and size, activation functions) for both the actor and the critic until the model converges to a decision policy that satisfies the optimization objectives (i.e. satisfy a maximum number of user demands with QoS metrics that match the SLA requirements, while minimizing resource and energy consumption by minimizing the number of servers used).

Simulation results

The model is trained over a number of episodes. For each episode, 15 steps are performed where for each step, the state of the network is provided to the model, actions are obtained and applied, and the reward is returned to the model. Due to the problem setup, the actions of the algorithm do not need to consider subsequent actions (and rewards) into account but only the current environment state. Furthermore, there is no ‘end’ state in the sense that a goal is reached but the optimization process continues indefinitely, so the episodes are in fact quasi-episodes that keep updating the policy until the best one is found. As shown in Figure 3-76, preliminary results reveal a convergence towards a decision policy that reaches a trade-off by maximizing the number of accepted flows while keeping the KPIs at satisfiable levels. As can be seen in the figure, the user acceptance rate is very unstable at the beginning, oscillates around certain values during training but increases in jumps when better deployments are found (upper right figure red line). The packet loss (upper right figure dark blue line), on the other hand, after the initial oscillations keeps decreasing until it reaches values within the SLA. The average delay is kept below the SLA values almost constantly. Similar trends are also visible if we see the decomposition of the reward into its constituents, visible in the lower right figure (d), where the reward from the flows KPIs keeps increasing. It can also be seen that the penalty from using additional nodes was minimized as a result of the consolidation of the deployed VNFs, which leads to a reduced overall resource and energy usage while maintaining KPI levels.

These results are obtained with the following hyperparameters: the actor and critic architecture comprise two hidden layers of 300 and 400 neurons respectively, where the activation function is ReLU, and the learning rate for the actor is 0.01, and 0.0001 for the critic. The ADAM optimizer is used for updating the DNN weights using an adaptive moment estimation.

The developed solution provides a resource-efficient mechanism for service deployment and user admission and routing in edge environments by reaching a trade-off between user request admission and QoS per user. Furthermore, the server usage penalty allows the placement policy to converge towards consolidation by placing service instances on the same servers whenever possible to minimize resource and energy consumption.



(a) Reward per deployment step (b) Flow performance after each step

(c) Cumulative reward after each step (d) Reward dissection after each step

Figure 3-76: Reward and performance statistics.

Compared to other state of the art such as [SHR20] and [LCA+22], the previously described work provides a solution that optimizes user request acceptance, end-to-end latency, packet drop, and resource and energy consumption as opposed to minimizing delay and cost only. Additionally, the provided solution considers a shared resource environment as opposed to fixed reserved resources per service. This allows for a more efficient resource allocation where functions are only allocated the resources that they consume. SLA violations for latency and packet loss are minimized by adapting user admission and distribution over the service instances. Finally, as can be seen in Figure 3-76 for the penalties from use of additional nodes, the obtained policy minimizes server and energy use, which is an aspect that has not been taken into account by the cited works.

3.6.1.2.3 Energy efficient resource allocation

Managing services and workloads efficiently in future networks is crucial for minimizing energy consumption. This is especially important given the increasing number of wireless devices with diverse and variable requirements, as well as the need to support a variety of challenging, new services. The functionality allocation component (see also functionality allocation in enabler 5.1) described in this section is an optimization mechanism that is responsible for the optimal placement of computational functionality/workloads to the available compute nodes (e.g., edge/cloud servers, robotic units) with maximum energy efficiency and trustworthiness (combining enabler 4.3, as will be explained later). Energy efficiency is addressed through the minimization of computing energy consumption, transmission energy consumption, subtracting the energy generated possibly from RF harvesters (e.g., solar panels etc.) if available.

Implementation

The computational workloads placement problem (M&O placement model) studies energy efficiency and trustworthiness when computational workloads and tasks are distributed to the various compute nodes of a

system. The input to this algorithm is the computational workloads /tasks $\{1, \dots, i, \dots, n\}$ with their computation and physical requirements as well as the data size ds_i and the location of data generation corresponding to each workload. Also, the compute nodes $\{1, \dots j, \dots, m\}$ are considered, with their capabilities (e.g., available CPUs $cpuN_{j'}$, memory), power consumption when idle and when fully loaded ($W_{idle}^{j}, W_{max}^{j}$), the battery level if applicable and the trust indexes $trustN_j$ assessed by the trust evaluation function of enabler 4.3. The network topology graph consisting of the nodes and the communication channels (with their throughput $cap_{j,j'}$) is also input to the algorithm. The aim is to minimize the objective function:

$$\min_{x,z}(a_1 E + a_2 L - a_3 T)$$

which is weighted (a_1, a_2, a_3) and consists of the energy consumption (processing, transmission, RF harvester generated energy) term:

$$E = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{j'=1}^{m}\left(\left(W_{max}^{j'} - W_{idle}^{j'}\right)U_{CPU}^{j'}\left(x_{i,j'}\right) + W_{idle}^{j'}\right)\frac{ds_i}{cpuN_{j'}}z_{i,j,j'} + \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{j'=1}^{m} z_{i,j,j'}p_{j,j'}\left(\frac{ds_i}{cap_{j,j'}}\right) - E_{harvest}$$

the end-to-end (E2E) latency (processing and transmission):

$$L = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{j'=1}^{m} z_{i,j,j'}\left(\frac{ds_i}{cpuN_{j'}} + \frac{ds_i}{cap_{j,j'}}\right)$$

and the trustworthiness term (this term is maximized):

$$T = \sum_{i=1}^{n}\sum_{j=1}^{m} trustN_j\, x_{i,j}$$

The $U_{CPU}^{j}\left(x_{i,j}\right)$ term denotes the CPU utilisation rate of the compute node j which is function of the binary decision variable $x_{i,j}$ which shows if the workload i is placed in the node j. The $p_{j,j'}$ denotes the power consumption in the communication between nodes j and j' and $z_{i,j,j'}$ is the binary variable showing if workload i using data produced in node j is placed in node j'. The constraints that should be respected are that each workload should be allocated to only one node, the available resources of the nodes should be respected, and the functional requirements of the tasks should be also respected if applicable. The output is the near-optimal placement of the workloads to the available nodes.

For solving this NP-hard problem, a metaheuristic algorithm was developed based on the genetic algorithm paradigm. The flowchart of this algorithm is shown in Figure 3-77. In particular, the algorithm initializes a population of possible solutions which are called chromosomes. Each chromosome consists of a series of nodes. Each node is the proposed node for each workload to be placed. The fitness score of each chromosome is calculated based on the objective function described before. The parent selection, crossover and mutation operations are respectively the tournament selection, uniform crossover and adaptive mutation and are responsible for improving the overall fitness of the population at each generation. The number of generations varies since it is determined by a dynamic stopping criterion.

The functionality allocation algorithm is designed to allocate compute resources efficiently across the continuum being use case-agnostic. Additionally, it can optimise the allocation of device-specific functionalities within robotic use cases. This application specific physical task planning functionality focuses on the energy efficiency and the system's efficiency in general when planning the various physical and robotic tasks/roles. The input is the physical tasks/roles with their requirements which may be the robotic specific features like camera, wheels, and propellers as well as the physical distance where these tasks are to be handled. Moreover, various physical nodes are considered, which are mostly robotic units, AGVs, UAVs, with their capabilities (e.g., available CPU, memory), power consumption related information, battery level, robotic specific features like camera, wheels, etc., and their physical location.

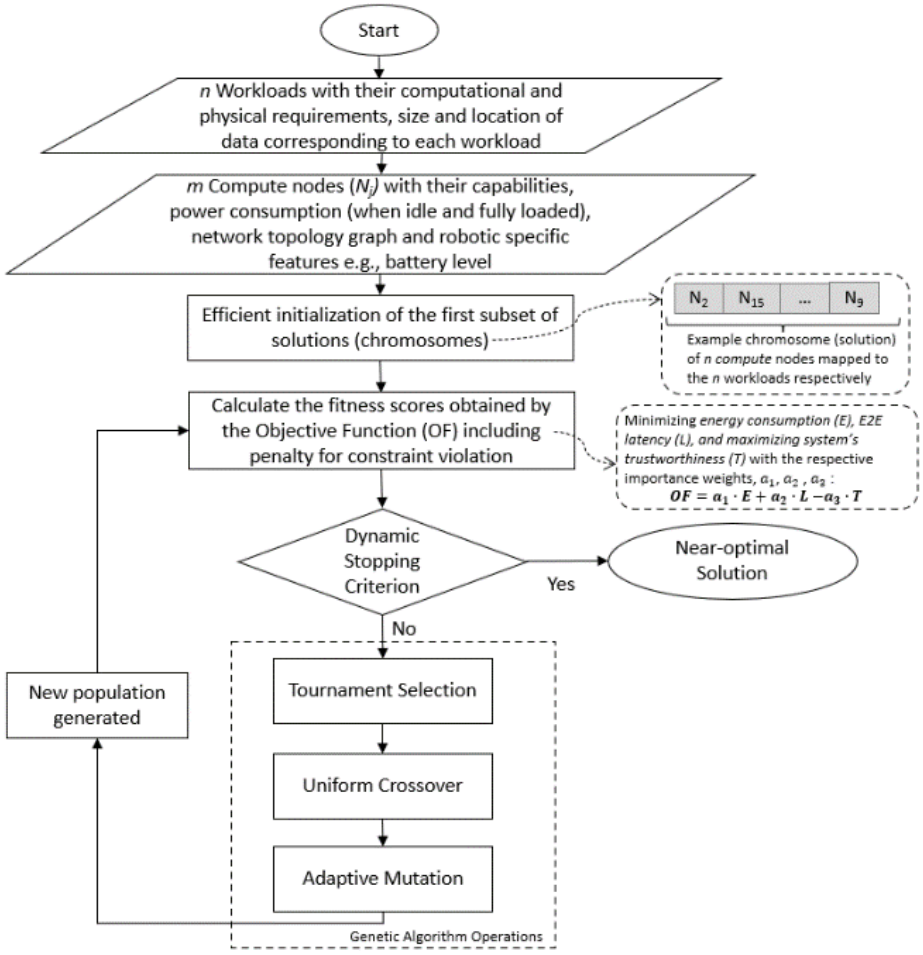


Figure 3-77: Flowchart of Functionality Allocation algorithm based on genetic algorithm paradigm.

The objective function aimed to be minimized consists of the cost of utilizing a physical node which is power consumption related, the total travel distance as well as the longest tour among the set of tours assigned to robots. The last term ensures that the workload is equally shared to the robotic units. The main constraints that should be respected are that each task should be handled only once by only one of the physical nodes. The battery level of the physical nodes, which indicates the maximum number of tasks a node can handle, should be respected and finally, the functional requirements of the tasks, if they need camera, wheels, propellers, should be respected as well. The output is the optimal planning of the physical tasks by the available physical nodes. In particular, the algorithm outputs a sequence of tasks assigned to each robotic unit. At this stage of development, the duration of each task is assumed to be the same no matter the robotic unit handles it. This problem has been initially solved with a Mixed Integer Programming solver and the analytic large-scale solution with the evaluations will be reported in the upcoming deliverable.

Simulation results

Functionality allocation component is integrated in PoC#A and is further enhanced within PoC#B and the inventory management use case. This implementation is described in the respective deliverables of E2E system PoCs' evaluations [HEX223-D21, HEX223-D22]. In this subsection, the focus is on presenting some preliminary simulation results.

Figure 3-78 shows the simulation results obtained by the functionality allocation mechanism (computational workloads placement) related to energy consumption. The energy consumption was calculated with the use of the energy consumption E formula presented above. In particular, the graph shows the decrease of the total energy consumption (percentage) obtained using the metaheuristic functionality allocation mechanism compared to the feasible round-robin placement algorithm. These measurements were taken at different weight levels of the energy term a_1: low, medium, and high. These correspond to an almost zero contribution, a moderate contribution and full contribution, respectively, compared to the other terms of the objective function.

Fifty fixed compute nodes were used having various capabilities and increasing number of computational workloads (10-170) with varied requirements.

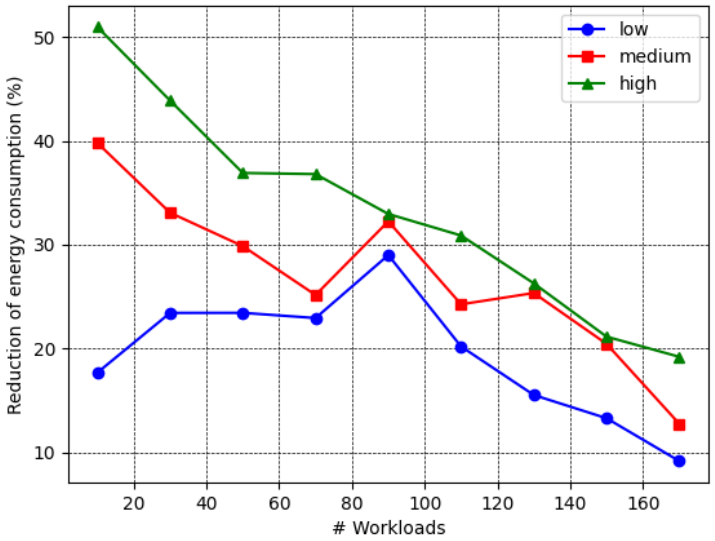


Figure 3-78: Energy consumption measurements of the functionality allocation mechanism.

As the graph indicates, the higher the weight of the energy term is, the higher the gains of energy consumption are, which is as expected. Also, as the number of computational workloads increases, the power consumption gains decreases. This is because when there is sufficient availability of compute resources, thus solution space and optimization potential, the described solution provides higher gains. The scarcer those resources become; the lower gain potential is observed. The energy consumption gains obtained in the described scenario by the functionality allocation algorithm compared to the round-robin placement are 9.9-50.9%. The trustworthiness related evaluations of this algorithm are presented in Section 3.4.3 (Enabler 4.3).

The described mechanism proposes a novel mechanism for optimally placing the computational workloads to the available compute nodes (edge, core, extreme-edge nodes) in a centralised manner, towards maximum energy efficiency and trustworthiness. The energy consumption is formulated as the sum of the processing and transmission energy consumption subtracting the energy generated possibly from RF harvesters and the trustworthiness in defined as the sum of the availability, reliability, security, and other metrics as described in Enabler 4.3. These two aspects that the optimisation focuses on, are not yet studied together in the literature as proposed here as far as we are concerned.

3.6.1.2.4 Sustainable MLOps

The path to a sustainable MLOps scenario involves the automation and optimization of the different workflows involved in the AI/ML artifacts implementation and deployment, e.g., regarding the data acquisition and processing, the AI/ML models design and development, and their training, testing, and validation, among others. Although MLOps is already a well-known technique[6], it becomes specially challenging in the telco environment due to the usual work procedures in this scope, in which software development is typically outsourced by the telco operators. The main features to be considered in this environment are:

- This is a **multi-stakeholder** environment, so several external software vendors could be hired to collaborative work to develop a network service (which may include AI/ML features). To make the service deployment/testing/monitoring possible, the proper communication resources must be issued among the different parties.
- In this multi-stakeholder environment, **data confidentiality must be preserved**. I.e., telco-grade MLOps requires to consider using techniques to ensure anonymity and confidentiality when sharing data between the operator and the software vendors involved in the network services development.

[6] The MLOps term was originally coined by Dr. Nisha Talagala back in 2018 (see https://www.linkedin.com/in/nisha-talagala-6a6b20, https://www.forbes.com/sites/nishatalagala/?sh=4b2ac4b63de9, or https://www.slideshare.net/NishaTalagala/ml-ops-pastpresentfuture).

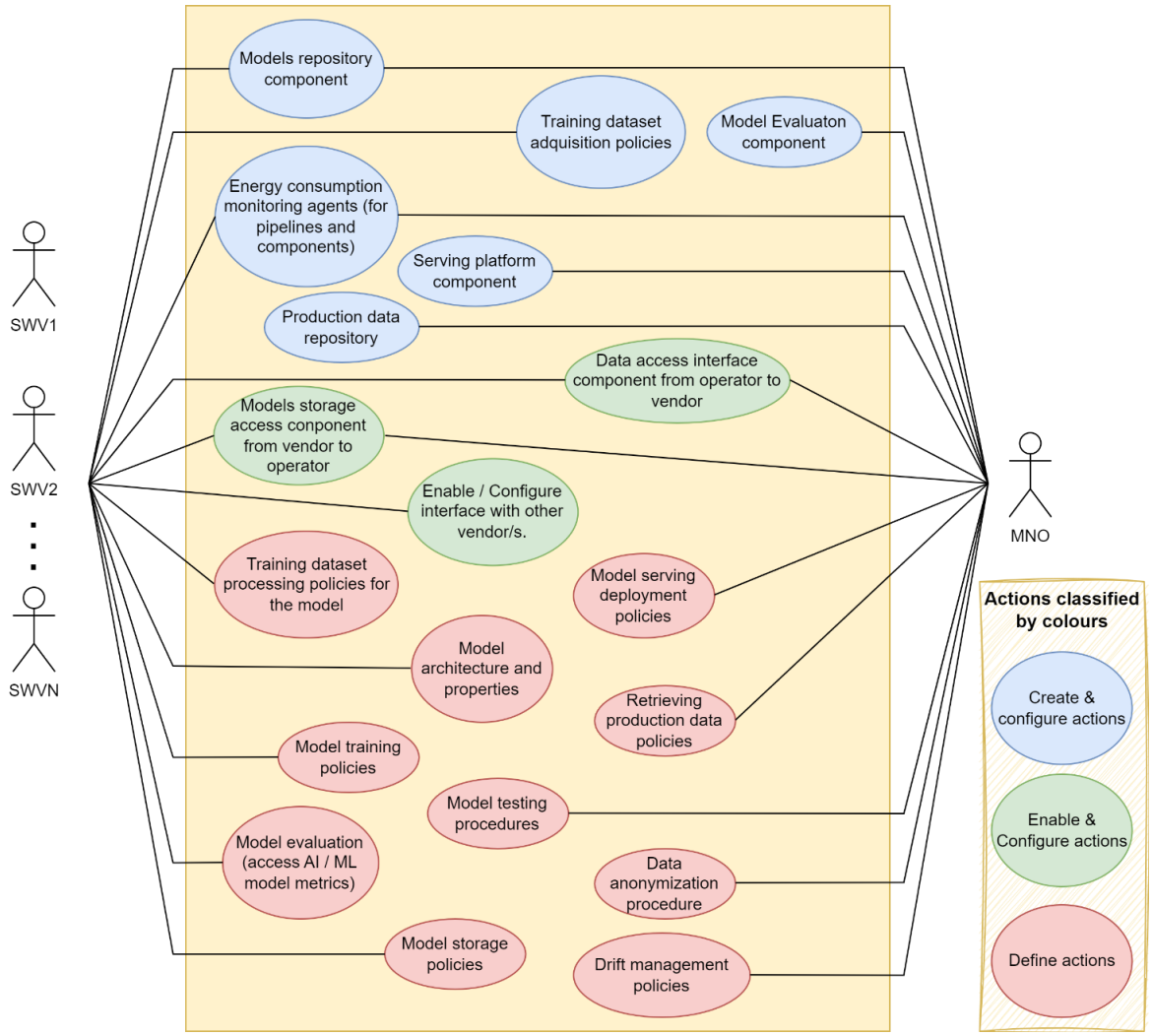


Figure 3-79: Actions use case diagram within the MLOps framework for vendors and operators.

The basis to address this MLOps concept in the telco scope were already set in the study carried out in the previous Hexa-X project [HEX23-D63]. This enabler builds on that initial work, but extending sustainability aspect in the approach, targeting to support the AI/ML development teams with energy related measurements, in order to help them to decide the more energy efficient MLOps workflows. The rationale behind this is that certain AI/ML models could require a non-negligible energy consumption for their training or operation, so it is considered appropriate to be able to design workflows considering that energy consumption. In line with this, the sub-enabler design considers a way of measuring the energy consumed in all the involved MLOps workflows, to identify those stages in the workflows consuming more energy (e.g., models training, inference, etc). This energy-related information could be leveraged to empirically check the trade-off between performance and energy consumption, and to take energy saving actions, as mentioned in section 3.6.1.

The sub-enabler implementation is envisaged as an asset allowing the users to rely on different predefined blocks to create a set of workflows that may suit with different AI/ML paradigms. This asset could be used by SW vendors and operators to allow them to define the workflows for them to design, train, and deploy AI/ML artifacts in a highly automated way (relying on DevOps techniques) in the telco operator environment, and at the same time, allowing them to select the most energy efficient MLOps workflows to deploy the AI/ML solutions. Figure 3-79 shows a high-level use case diagram with the main actions that could be performed by SW vendors and operators relying on this asset.

Metrics and measurements definition

To be able to take energy saving actions and apply the corresponding sustainability-based policies, it is necessary to define what needs to be measured, where it can be measured, and how the measurements and metrics would be taken and generated considering the different stages of the overall MLOps processes.

From one hand, the granularity of the information retrieved (resources level, application level, process level…) needs to be defined, first, considering the overall measurements corresponding to the different computing resources where the MLOps workflows are running, as well as those measurements from the network, considering the communications needs among computing nodes, and the data sources for modelling, training, and executing. With these measurements, different metrics related to the overall sustainability could be computed (illustrated in Figure 3-81) in order to take a clear and detailed picture of the context where the MLOps workflow is running, allowing the decision making based on that information.

Second, considering the specific measurements corresponding, for example, with:

- AI/ML applications and models.
- Data Management Frameworks.
- Pipeline scripts (e.g. data normalization scripts).

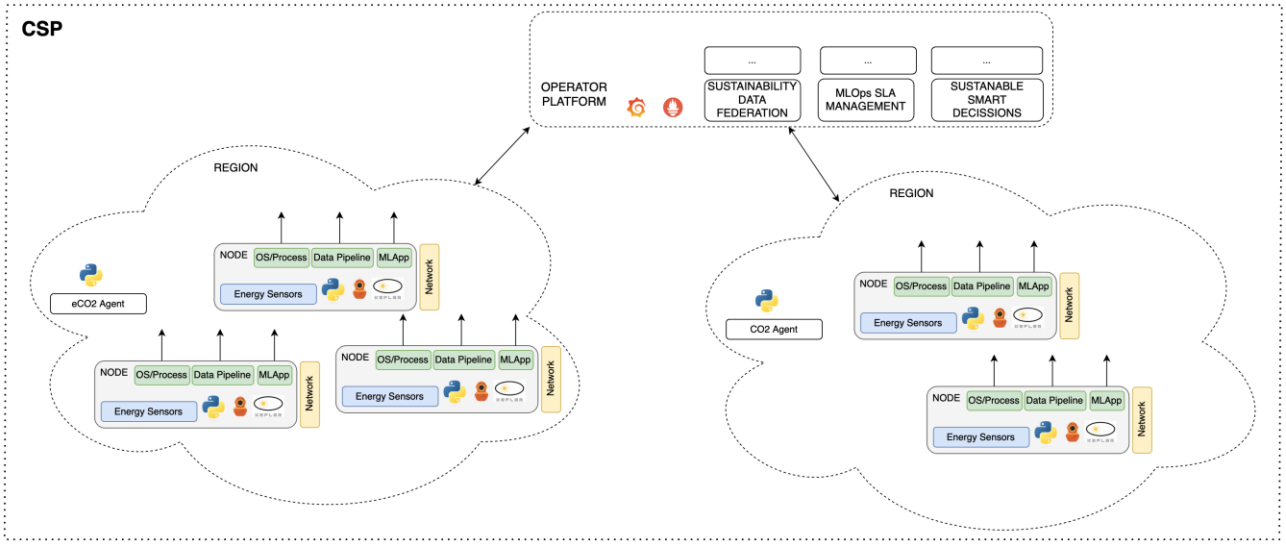
That would be used in the different phases and steps of the MLOps workflows, to obtain a detailed contribution of each process to the energy consumption, being capable to aggregate the information by stage and then, monitor and take decisions based on that information.

Third, considering the specific measurements that can be made inside the ML artifacts, if a more fine-grained information to take decisions were needed. The objective is to have more fine-grained information, to take decisions, regarding the MLOps lifecycle energy consumption adding the capability to compare the performance of different models in both, training and execution (Figure 3-83).

Figure 3-80 illustrates a high-level diagram with the network elements that could be used to retrieve the sustainability related information, considering the different stages of the MLOps workflows implemented by SW vendors and operators.The diagram considers the needed measurements level (related with each domain and step of the MLOps life cycle) and the relevance of having enough data for a clear view of the contributions in terms of sustainability (e.g. OS processes and common frameworks), power consumption at ML application level (MLApp represents the machine learning application/s in execution over the different computing nodes - e.g. a forecasting app that implements a model trained for execution -), and the equivalent in CO_2 emissions per region (depending on electricity production scenario). In the diagram is being considered too the operators IT platform where a centralized view and management functionalities related with MLOps is conceceptually placed.

To perform those measurements, a variety of energy sensors need to be deployed on each node where the different stages of the MLOps workflow are running, considering the different architectures capabilities, limitations and the additional requirement of exposing metrics calculated using normalized ontologies for each granularity layer. As well as the multi technology energy sensors, different instances of the $eqCO_2$ agent need to be deployed by region, to add information or estimations about the distribution of energy sources. In addition to this, global modules need to be considered for the management of the sustainability metrics as well as to implement the management of the work loads, SLAs and policies.

With the information retrieved from the different levels of granularity, even in a distributed scenario as the one in Figure 3-80, energy-saving decisions can be taken based on information retrieved from nodes and policies/slas defined, on the operator platform side (MLOps Sla Management & Sustainable Smart Decisions), such as moving part of the workloads to different nodes (inside or outside the same region, depending on the policies defined), or workflows re-design decisions, by comparing the efficiency of the models. Besides, it would be also possible to rely on temporal data analysis to timely start/stop different process in a more optimal way. An additional feature would be also the possibility to perform MLOps efficiency grade certifications, since all the measurements and metrics could be used to certify computing nodes, applications, or models, among others. This could be useful to compare different results and decide where/when to execute the MLOps workflows.



Icons left to right: Python & Measurement libs, Scaphandre, Kepler, Grafana and Promethus

Figure 3-80: High level architecture for sustainability measurements

Preliminary implementation and early validation results

To test some of the capabilities defined in the above paragraphs a couple of AI/ML artifacts (ML applications for base classification and clustering models) were implemented and deployed over a testbed node. Based on that, the following plots were obtained, illustrating the measurements retrieved from the sensors deployed using different levels of granularity, targeting the AI/ML models and the nodes on which they were deployed, and considering different activities related with the implemented MLOps workflow (considering training, evaluation, deploying, execution and swapping for a simple classification use case). This demonstrates the capability of the system to retrieve sustainability information with different granularity related with the different stages of a MLOps workflow. Specifically, Figure 3-81 shows an example with carbon production measurements for the computing nodes used during a particular MLOps workflow. Figure 3-82, on the other hand, shows power consumption measurements in mW, while Figure 3-83 shows the power consumption for the ML Artifacts, also during a particular MLOps workflow.

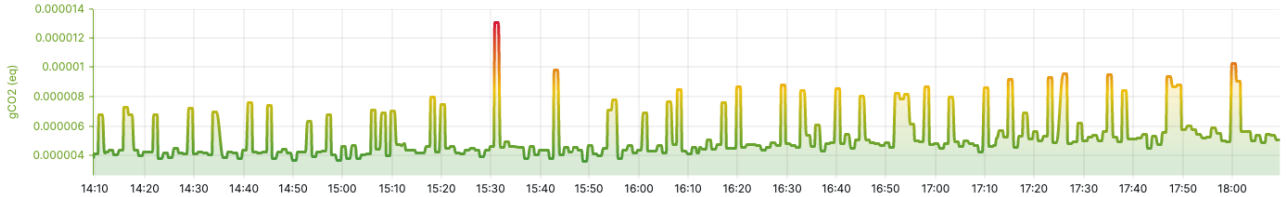


Figure 3-81: Example of carbon production equivalent measurements over a node, in gCO_2.

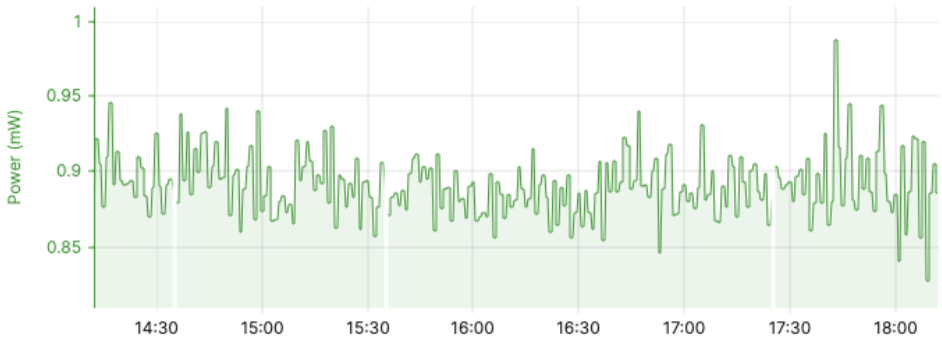


Figure 3-82: Example of power consumption measurements over a node in mW.

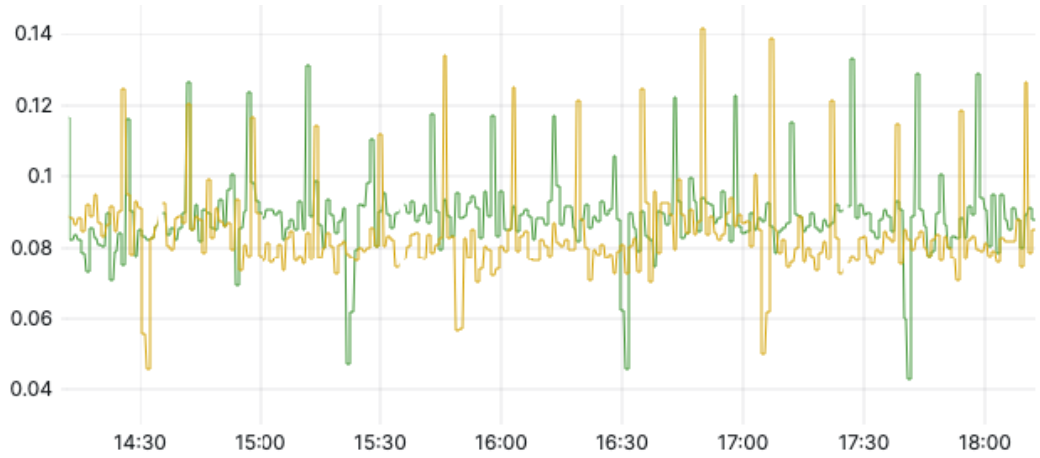


Figure 3-83: Example of power consumption measurements over different ML Artifacts.

Finally, and regarding technology support, some limitations were detected while performing the testing regarding some hardware and software restrictions that impacted the capacity to perform measurements without assuming adaptations and developing new sensors. These limitations ware mainly related with the CPU architectures (x86 or ARM), GPU drivers' availability and, in case of virtualization of the computing nodes, in the virtualization engine itself.

3.6.1.2.5 Multi-domain federated learning

The overwhelming resource pre-requisites of training AI models at scale has meant that only expensive, high performance compute nodes would be self-sufficient in these kinds of applications. Networked nodes, working cooperatively as a distributed compute fabric, therefore provide a more apt solution to this problem. Network management therefore comes to the fore as an important concept in ensuring that such demanding envisioned 6G services as AIaaS can be satisfactorily handled with little impact on co-existing services. Moreover, in the contemporary landscape of artificial intelligence, the distributed nature of data poses challenges for model training. Privacy concerns emerge when sensitive data is shared, and the resource-intensive process of transporting training data over networks requires energy [AMK+18]. Decentralized Learning (DL) emerges as a solution by training models across multiple entities, collecting data locally, and aggregating models from each entity. However, DL faces the delicate challenge of accurately identifying and integrating pertinent models given that errors in this process can result in inaccurate aggregated models. Recent advancements propose an E-TREE DL architecture [YLC+21] that organizes nodes into clusters, facilitating the hierarchical aggregation of model weights.

In collaborative scenarios involving multiple administrative domains collaborating on a training task, the objective is to define a DL logical training topology that determines interactions among domains. Decisions regarding optimal resource allocation (computing nodes, data sources, and computational/network resources) are made with the overarching goal of minimizing the overall energy consumption.

Preliminary implementation and early validation results

This optimized decentralised learning technique enhances the performance of the ML training component of the ML-Ops management function depicted in Figure 3-71. We define an administrative domain as a set of nodes (each consisting of a distinct data source and compute resources) that are individually managed and can interact with each other sharing model weights to produce a local model. The domains could be implemented as virtual machines and the compute resources as multiple containers within each. In the proposed system, multiple administrative domains collaborate within a Federated Domain Set (FDS) to collectively optimize the utilization of shared resources. The interconnection between these domains is via an End-to-End Federation Layer, administered by a Federating Orchestrator (FO) operating within each domain, as depicted in Figure 3-84. Each administrative domain maintains one dataset for training and another for testing. The latter is dynamically updated with input data collected within the domain, ensuring that it consistently reflects the current distribution of input data and accommodates any potential drift that might arise over time.

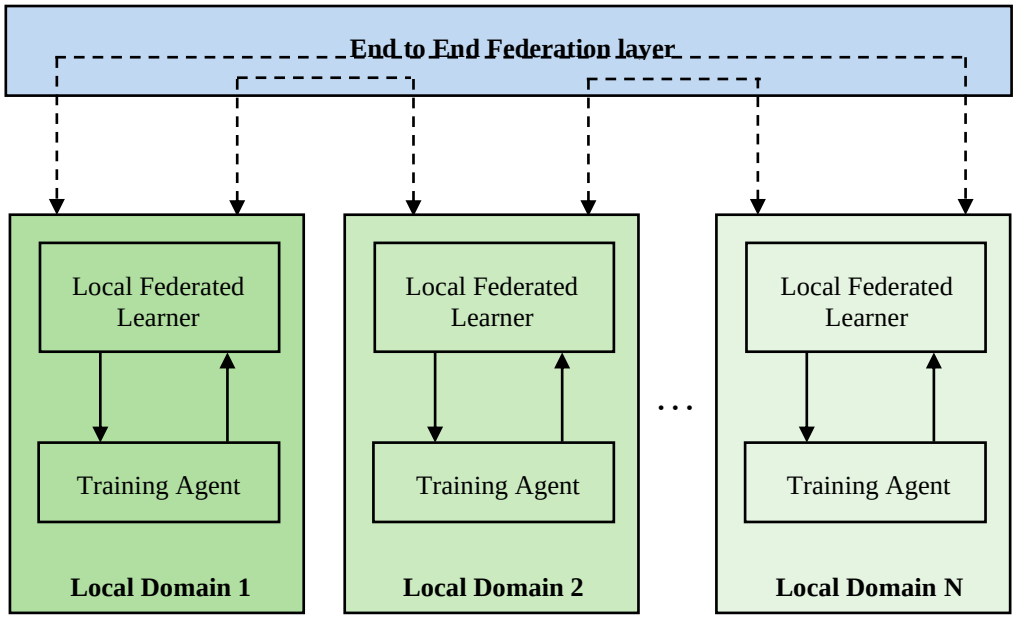
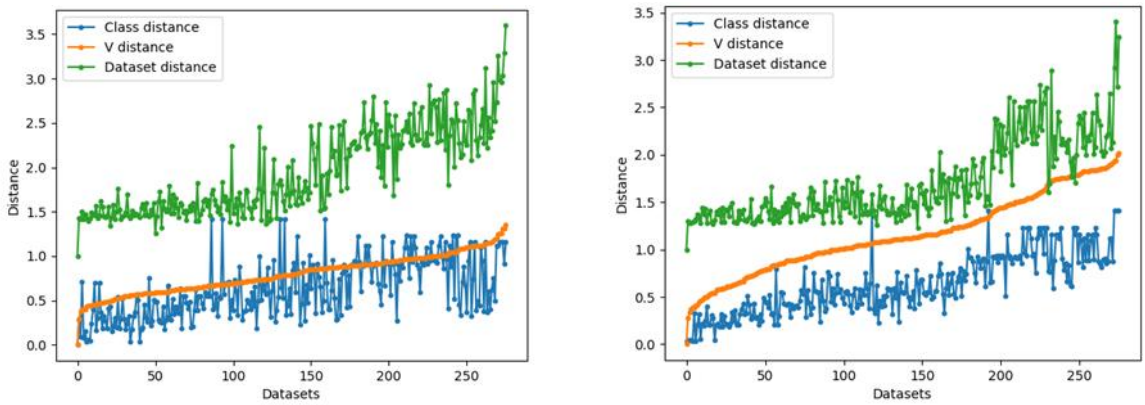


Figure 3-84: Schematic of decentralised learning.

To proceed with the problem and solution, a reliable way of selecting significant data for training the model is needed. To this end, we measure differences between datasets to identify unexplored samples. These measures are represented in Figure 3-85, for datasets with 500 samples (a) and 980 samples (b). Green lines are the distance between the raw datasets, treated as a matrix to compute the Frobenius distance; blue lines are the L2 distance between the class distributions of the datasets. Remark that the distances are normalized to fit in the plot.



(a) (b)

Figure 3-85: (a) 500 samples and (b) 980 samples.

The decentralised learning system proposed here is based on [YLC+21], which is enhanced by optimizing the selection of the participant nodes in the model aggregation. In this way only nodes that contain sufficiently uncorrelated data are enjoined in model aggregation thereby avoiding the unnecessary synchronization between relatively similar nodes.

3.6.1.2.6 Multi-agent Reinforcement Learning for adaptive scaling

The increased complexity of network services and application graphs deployed across the continuum does not fit well with centralised approaches such as optimization techniques or single-agent-based AI algorithms whose decision space grows exponentially with the addition of services or links. Handling resources or tasks in a decentralised manner becomes more and more crucial for the satisfaction of the service level objectives (SLOs) defined in a set of available deployment options. For implementing such a scenario, a Directed Acyclic Graph (DAG) -formed application graph with individual services communicating with each other is deployed in a multi-cluster setup. The traffic created by user requests is distributed across the graph according to the application's structure and so, each service needs to handle different loads at each time point. The developed

autoscaling mechanisms make the application adaptive to the variable incoming workload by spawning the necessary number of replicas for each service to handle the corresponding load, while considering the defined optimization objectives, i.e. latency and energy requirements, and the respective resource constraints.

Implementation

An initial implementation of the QMIX algorithm [RSS+18] is considered for service autoscaling towards latency minimization based on a Kubernetes setup. In specific, a three-service application (as shown in Figure 3-86) is considered for deployment on a single node infrastructure with sufficient resources (16 CPUs, 32GiB RAM). For each service, the corresponding agent monitors the environment and collects all relevant information for the service. During training, the agents aggregate their outputs and provide the input for an extra layer that provides an output for the training actions. This process is referred to as centralized training. For the inference part, the local models of the agents are used for action selection (decentralized execution).

The environment of each agent is defined as follows:

- State space:
  - Workload: Request arrival rate (reqs/s)
  - Predicted workload: Predicted request arrival rate (reqs/s)
  - Service computation latency: The latency currently measured (ms)
  - Service CPU/memory usage (MiB)
  - Node CPU/memory usage (mcores)
  - Service pods number deployed (#)
- Action space: The number of replicas spawned for the service.
- Reward: The reward is calculated as a function of the consumed resources (CPU) and latency measured:

$$Reward = w_{lat} * R_{lat}(lat) + w_{res} * R_{res}(CPU)$$

The design of the formula guarantees minimization of end-to-end latency, while considering the resource consumption constraints that may be existent.

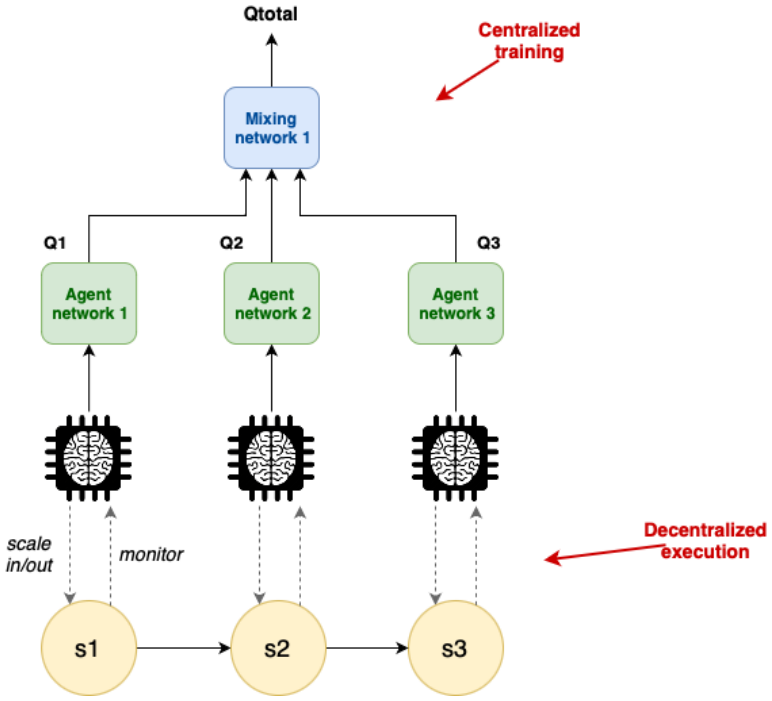


Figure 3-86: QMIX application on service auto-scaling.

The algorithm showcases how different agents in the continuum controlling different interacting resources can work together towards a common goal. The idea leverages on collaborative multi-agent systems theory as well as adaptive ML to identify features of the environment that influence individual decision making in coordination with similar agents that coexist ad influence each other. This work will provide a real-world

implementation of the QMIX algorithm applied in a real testbed and service to validate its applicability in unstable and evolving network environments. The goal is for a global reward (centralised training) to guide local decision making (decentralised execution) and enabling collaboration between independent entities with separate (but dependent) objectives in the compute continuum.

3.6.1.3 Impacted KPIs and KVIs

The main KPIs impacted by this sub-enabler are the following:

- **Energy efficiency:** AI-based resource efficient network function deployment minimizes resource use through consolidation by reducing the number of used nodes, which translates into energy saving, while optimizing end-to-end latency and packet loss metrics. Energy efficient resource allocation and physical tasks scheduling also leads to improvements in the total energy consumption of the system by collecting sensing, localization, traffic and mobility pattern data as input for the AI/ML algorithm. AI-based solutions can also be used to automate and optimize post-deployment orchestration operations such as service migration or scaling to minimize energy usage. On the other hand, the energy consumption of the MLOps workflow for the different phases of model training and maintenance can be assessed and optimized by retrieving metrics of different granularities from the workflow. When using Federated Learning in a multi-domain edge architecture context, each step to convergence bears a data communication load and energy cost as the FL members exchange data. Thus, optimizing the local clustering of edge nodes reduces data communication and energy cost.
- **Service KPIs:** Service KPIs such as the end-to-end latency and throughput are optimized through the AI-based M&O control procedures for resource allocation, task scheduling and post-deployment lifecycle management.
- **Automation:** This enabler contributes to automating the M&O control system and reducing OPEX through full zero-touch M&O decision mechanisms using AI-based solutions which provide more efficient decisions compared to manual or default decision mechanisms. Additionally, the developed configuration recommendation system allows for increased automation of the network and service M&O by translating high-level requirements into configurations that best match the performance and energy related requirements.

The main KVI impacted by this sub-enabler is **sustainability**, as the energy and resource efficiency targets in the different AI-based solutions lead to reduced energy consumption by the system for computing, data transmission, MLOps workflows, as well as the energy consumed for using hardware edge nodes.

3.6.2 Sub-enabler 6.2: Trustworthy AI/ML-based control algorithms

The purpose of trustworthy AI/ML is to ensure that AI/ML systems are created and used in a transparent, accountable, dependable, safe, and ethical manner. There are several fundamental characteristics connected with trustworthy AI/ML [S21], including security, privacy, and explainability. The security element of trustworthy AI/ML examines the AI/ML system's design to be resilient, dependable, and safe for both individuals and society. The privacy component of trusted AI/ML strives to construct the AI/ML system in compliance with data protection laws and regulations, respecting individual privacy and protecting personal data disclosure. The third important factor connected with trustworthiness is explainability. ML models that handle difficult computational tasks with almost no human intervention are naturally complex black boxes. This has heightened the need for accountability while also raising worries about AI/ML systems' decision-making processes. Explainability strives to increase the transparency of black-box ML models, allowing humans to understand the decisions made by AI/ML systems. The trustworthy AI/ML-based control enabler aims to provide conceptual solutions and implementations for more robust models by protecting against adversarial attacks, reducing leakage of personal and sensitive data, and providing clear and concise explanations and justifications for the AI's reasoning or decision-making process.

3.6.2.1 Sub-enabler design

This sub-enabler focuses on the development of mechanisms for improving the trustworthiness of the developed AI/ML-based control solutions by protecting the AI/ML models against adversarial attacks, data leaks and by providing explainability.

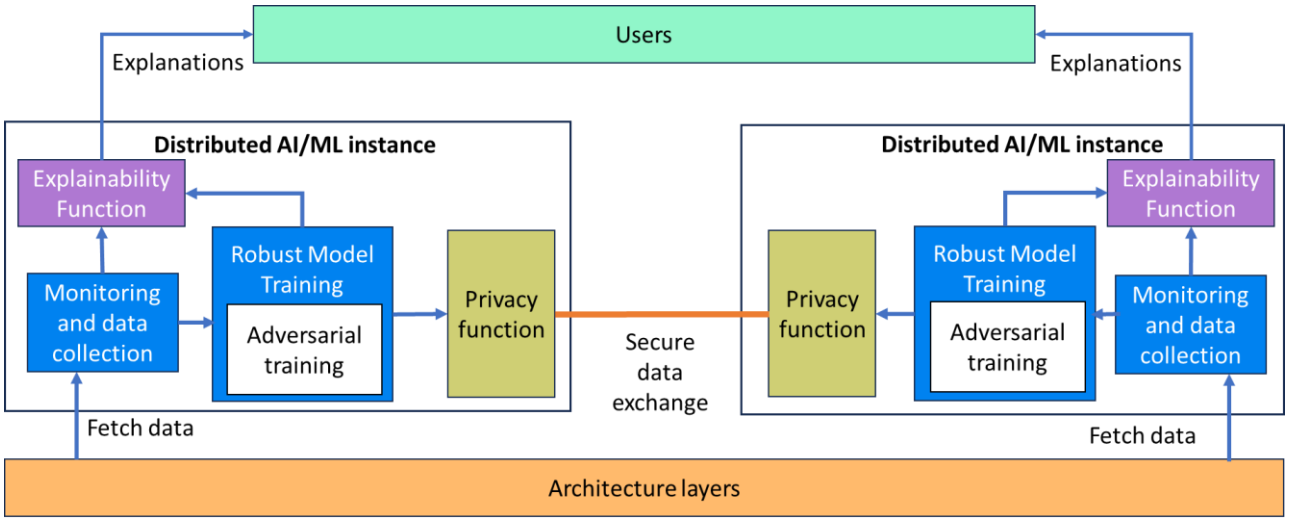


Figure 3-87: High-level architecture for trustworthy AI/ML-based control.

The envisioned high-level architecture for this sub-enabler is shown in Figure 3-87 and includes the following components:

- **Robust model training:** This function is responsible for performing AI/ML model training by including adversarial attack samples in the training dataset to create AI/ML models that are more resistant against adversarial attacks.
- **Privacy functions:** In distributed training scenarios such as Federated Learning, training data is exchanged between the components, which makes the data vulnerable to privacy attacks. The privacy functions are deployed with each distributed instance, and used to exchange data in a secure and privacy-preserving manner.
- **Explainability function:** This function applies XRL techniques to provide human-interpretable explanations of AI/ML-based decisions within the M&O framework. The AI/ML model-level explanations also serve as fundamental components for aggregation and abstraction, potentially contributing to achieving system-level explainability. While system-level explainability is not the main objective of this function at this stage, it represents a significant area for future investigation.

3.6.2.2 Preliminary implementation and early validation results

This section describes the preliminary implementation of an example of trustworthy AI/ML-based control, with particular reference to a secure AI/ML-based control for an Intent-based Management System.

3.6.2.2.1 Secure AI/ML-based control for Intent-based Management System

The advent of intent-based management (IbM) in overseeing telecommunication systems within the 6G landscape underscores the importance of recognizing potential threats and risks accompanying its adoption and execution. IbM relies on automation and AI/ML to administer networks according to overarching goals and requirements outlined through intents, in a high-level and abstract manner. While AI/ML models empower IbM systems to dynamically adjust network parameters and resources to fulfil intents, these models possess inherent vulnerabilities to adversarial attacks. Such attacks have the potential to degrade the IbM system's performance and prompt erroneous decision-making, consequently impacting network performance and incurring financial costs for the operator. Thus, it is important to be aware of the vulnerability of IbM system against adversarial attacks originated through malicious component and the mitigations to decrease the chance of such attacks.

In 6G networks, due to their complexity, the networks should be self-organizing and self-adaptive to maintain high performance under different circumstances. Thus, it is important to provide management solution with the knowledge about the expectations, requirements and constraints in order to enable these capabilities in the network. This knowledge can be defined by an intent for an autonomous system. The goals and the expected state can be given to the technical system through the intents in a high-level format. Thus, the network needs to take required actions to satisfy the defined expectations by the intents. According to the TM forum [TMF-

IG1253], the Intent Management Function (IMF) is responsible for handling intents and executing the intent-based operations. Information including the data gathered from the managed environment and network state, intents and their current state and expectation, as well as domain models and expert knowledge and rules are stored in knowledge base. It seems that the knowledge base is a key component in intent-based management, as it is a place to store the most important data, so it must be privacy-preserving and protected against adversarial attacks.

The source of adversarial attacks in IbM systems can stem from vulnerabilities in the underlying ML models which are foreseen to be used by agents to facilitate the predictions and the decisions that are taken by the IbM system. They may also originate from other sources such as an adversary can generate various randomly constructed inputs and inject them in an intent and then observe what happens to the network. In another case, the adversary can collect system information and gain knowledge about the managed environment, as well as disclosing the ML models by inferencing input/output APIs. Adversaries may create carefully crafted inputs to cause models to take incorrect decisions. In IbM system, adversaries may exploit the vulnerabilities in the models that are used by agents to propose actions, made predictions, and evaluate actions. The adversaries may craft adversarial examples by modifying the submitted inputs to the AI model in the agents and may deceive the agents to propose wrong actions, predict wrong impacts, or make wrong evaluations. In addition, adversaries can cause the system to take action, while there is no need for such action and vice versa.

Implementation

To observe the effect of adversarial attacks on the generated model which are used by the agents in the IbM system, experiments are conducted using an internal network emulator that simulates end-to-end network connectivity. The Intent Management Function (IMF) prototype developed in [BJZ+22] has been used. It incorporates a knowledge base, reasoner mechanism, and a set of agents. To facilitate connectivity and execute pertinent functionalities for data and service exposure, a set of user plane and control plane network functions are established. IMF uses these exposures to keep an eye on the network and adjust as necessary. A cloud-deployed conversational video service instance is set up as part of the trials. The QoE reported by the application itself based on a formulation is the primary KPI for this sort of service. As a result, the intention to target this kind of service is predicated on a need that establishes a minimum QoE threshold.

In the experiments, a topology was generated in the network emulator with 2 UEs consuming the video service and downlink video traffic flowing from application to the UEs via UPF and gNB as it is shown in Figure 3-88. Data were collected under different network configurations. To achieve this, the network was configured on the run by taking arbitrary actions such as modifying the priority of the video traffic flows or maximum bit rate~(MBR) set for each UE. The data set for training a QoE prediction model consists of measurements and recent configurations of the environment that are collected from both network and the application. The data set consists of 15 different features (e.g., number of received frames, throughput).

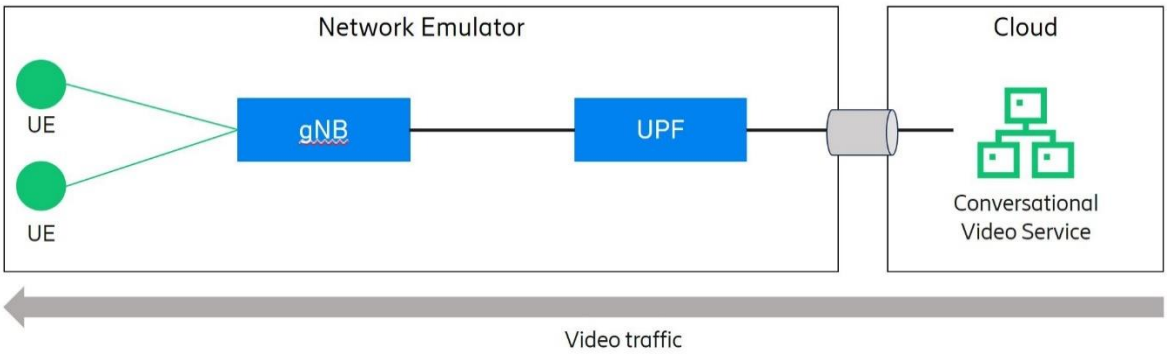


Figure 3-88:Topology in our in-house network emulator

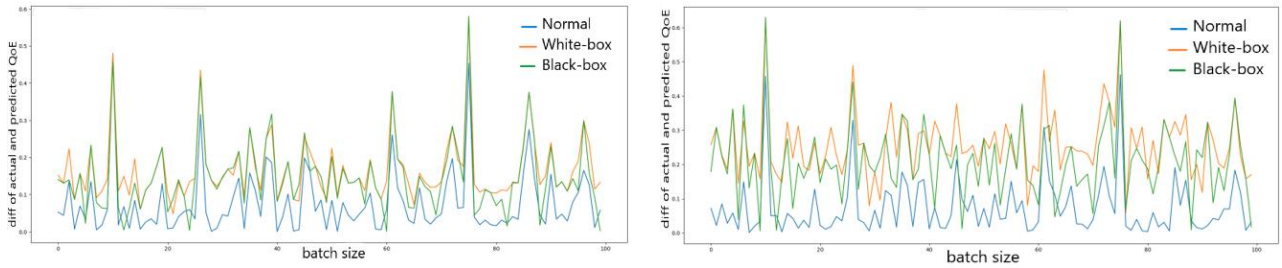
It should be noted that even though there is a QoE formulation integrated into the conversational video service application, some of the input features are found irrelevant and redundant due to the modelling of the network emulator. In practice, the QoE is estimated as QoE = f(x) + noise, where f(x) is the QoE formulation. Since the traffic from application towards UEs flow in real-time, the queue management might create noise in the data.

The aim of the experiment is to show how adversarial attacks can degrade the performance of the model and impact the decisions which are taken by IbM system as well as the actions taken by the reasoner. A Neural Network was used to generate a regression model for QoE using measurements collected from both the network and the application. There are several parameters in the Neural Network to be configured which impact the performance of the model. In the experiments, the number of layers is set as 3, and the activation function is selected as Rectified Linear Unit (ReLU), the batch size is 100, and number of epochs is 15. The choice of loss function for the regression problem is also important: Mean Squared Error (MSE), which is a commonly used loss function for regression tasks, is used in the experiments. The training and test losses for a regression problem using MSE are numerical values that show how well a prediction is made by regression model. The dataset for the experiments contains 13437 samples where 80% of the data is used for training and the rest for the test. A surrogate model was also created using the same distribution of the main dataset with a different neural network architecture.

Experimental results

In order to show how adversarial attacks can degrade the performance of the model, a set of experiments have been conducted, testing the performance of the model against both types of attacks, white-box and black-box attacks with two different attack methods, Fast Gradient Signed Method (FGSM) and Basic Iterative Method (BIM).

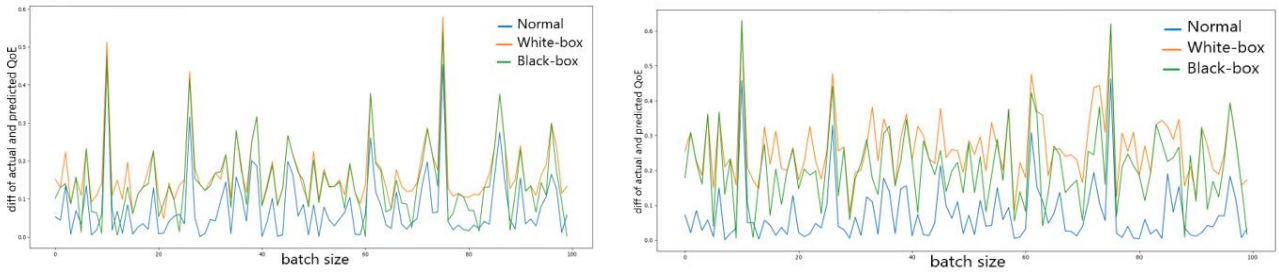
FGSM experiment for both types of white-box and black-box attacks. In FGSM method, the magnitude of the perturbations added to the input data is controlled by the epsilon parameter. Larger epsilon values produce more pronounced alterations in the input data, whereas smaller epsilon values provide less obvious disturbances. The selection of epsilon is important because it establishes the trade-off between the adversarial example's imperceptibility and its capacity to trick the model. In the experiments, as shown in Figure 3-89, the epsilon value is selected as 0.1 and 0.2.



a - FGSM EPS=0.1 b - FGSM EPS=0.1

Figure 3-89: Adversarial attack using FGM method.

BIM experiment for both types of white-box and black-box attacks. In the BIM method, the magnitude of the perturbations added to the input data is again controlled by the epsilon parameter. Alpha is the step size and commonly is set to a fraction of epsilon. The number of iterations determines how many times the perturbation is applied to the input. Two other parameters, alpha and number of iterations are respectively set as epsilon*0.2 and 15 (Figure 3-90).



a - BIM EPS=0.1 b - BIM EPS=0.2

Figure 3-90: Adversarial attack using BIM method.

Table 3-4 shows the MSE value obtained when adversarial attacks are applied. The value of the MSE before the attacks are applied is 0.01379. Adversarial attacks can lead to an increase in MSE if the perturbations

introduced by the attack significantly distort the input data in a way that the model's predictions are further from the true values. It is important to note that the effect of adversarial attacks on MSE can vary based on several factors such as robustness of the model, the nature of the data, type and strength of the attack.

Table 3-4: MSE values of the original model after applying adversarial attacks.

Attack Type	Black-box	White-box
FGSM Attack	0.03183	0.03768
BIM Attack	0.03239	0.03856

Mean square error increases may be less for models that have been specifically trained or altered to be resistant to adversarial attack types. Also, the effect of adversarial assaults on MSE may also be influenced by the qualities of the input data and the model's sensitivity to adjustments in that data. The impact of adversarial attacks on the mean square error of a machine learning model is contingent upon both their type and strength. Stronger adversarial attacks, characterized by larger distortions to the input data, often lead to higher MSE values. Furthermore, the kind of attack used—such as targeted, untargeted, black-box, or white-box—can have varying effects on the model's performance and, in turn, the MSE.

In the experiment, to improve the performance and make the models resistant against adversarial attacks, adversarial training methods are employed. In the experiments, the BIM technique is used to create adversarial examples and include these samples while training the machine learning model. FGSM and BIM adversarial attack methods are applied to evaluate how much the performance of model is improved against these attacks. As expected, the loss value of the model generated using augmented training dataset is decreased, which shows that the model is more resistant to adversarial attacks. The comparison of MSE values for the trained model before and after augmenting adversarial examples and applying FGSM and BIM adversarial attacks are demonstrated in Table 3-5 and Table 3-6.

Table 3-5: Comparison of the MSE values of the original and robust model after applying FGSM attack.

	MSE without attack	MSE black-box	MSE White-box
Original Model	0.01379	0.03183	0.03786
Robust Model	0.01430	0.02130	0.02401

Table 3-6: Comparison of the MSE values of the original and robust model after applying BIM attack.

	MSE without attack	MSE black-box	MSE White-box
Original Model	0.01379	0.03239	0.03856
Robust Model	0.01430	0.02158	0.02451

As it is shown in Table 3-4, the MSE of the original model on clean data should be relatively low, assuming the model has been trained well on the given task and data. Applying FGSM or BIM attacks to the original model led to an increase in MSE. In general, BIM tends to generate adversarial examples that are closer to the decision boundary of the model compared to FGSM. This iterative approach of BIM often results in more potent perturbations, which can lead to higher distortion in the input data. Consequently, we would expect BIM to result in a higher MSE compared to FGSM. The robust model trained using adversarial examples should also have a relatively low MSE on clean data similar to the original model. As shown in Table 3-5 and Table 3-6, the robust model exhibits greater resilience against FGSM and BIM attacks compared to the original model. While the MSE still increases after applying the attacks, the expected increase is smaller compared to the original model.

3.6.2.3 Conceptual solutions

3.6.2.3.1 Privacy Protection Framework for data analytics in M&O

ML is expected to be used in 5G and 6G where one of the centralized frameworks responsible for ML-based analytics is Management Data Analytic Service (MDAS)/Management Data Analytic Function (MDAF) in the management and orchestration (M&O) layer. MDAS/MDAF functionality enables a service consumer to obtain management data analytics. The primary role of the MDAF in a 5G network is to perform data analytics

and provide insights to efficiently managing and optimizing network resources, ensuring high-quality service delivery, and enabling intelligent decision.

One of the aspects which may be needed to be considered in this framework can be data confidentiality and privacy. Although federated learning which is a privacy enhancing mechanism can be used among different MDAFs to train a global model without disclosing the local data trained at each MDAF, the trained local ML models by analytics functions may be vulnerable against privacy attacks. Thus, secure aggregation protocols or some other solutions are needed to hide the individual local model updates from the aggregator/server but allow the aggregator/server to learn the aggregated result of the individual local model updates.

The proposed solution introduces a new privacy protection framework for data analytics in M&O. This framework involves two entities called Privacy Management Function (PMF) and Privacy Operation Function (POF). The PMF has functionalities of receiving the list of FL clients and a privacy operation, generating required parameters and keys for the privacy operation for the FL clients, and sending the required parameters and keys to the FL server and FL clients. At FL client, the POF receives parameters and key(s) for the privacy operation, receives a local model update, and performs the privacy operation on the local model update and returns the result. At FL server, POF receives parameters and key(s) for the privacy operation, receives privacy-operation-applied local model updates, and performs aggregation in a privacy enhanced way and then returns aggregated result. The general overview of the proposed architecture is illustrated in Figure 3-91.

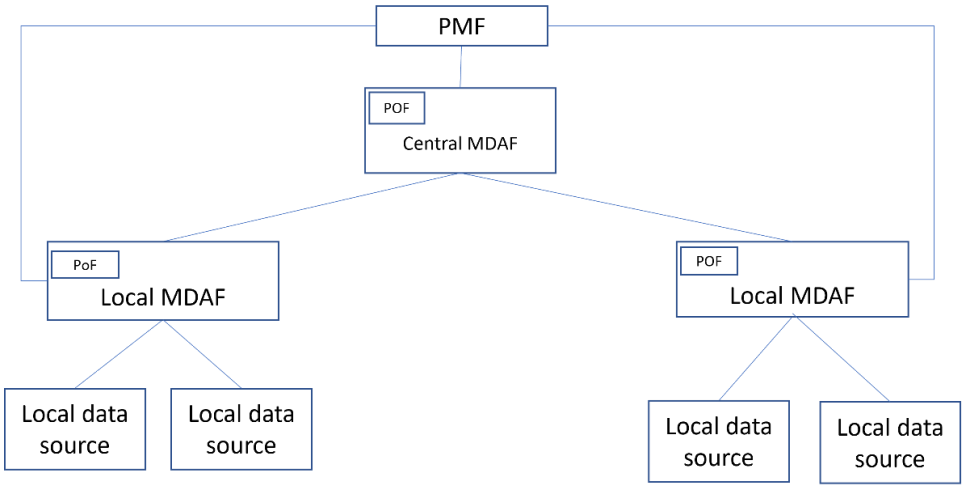


Figure 3-91: General Overview of privacy protection framework.

As illustrated, there could be several local MDAFs which collect data from management environment and execute local training. The privacy protection (i.e., confidentiality protection of the local training against the central MDAF) may be necessary in some deployment scenarios (e.g., client MDAFs may place in different MNOs). To provide privacy, the local MDAF (as an FL client) sends the local model updates to the privacy operation function and gets processed result. The processed result can be generated by applying any privacy enhancing technologies like secure aggregation and differential privacy to the local model updates. Local MDAFs send the privacy processed local model update to the central MDAF (FL server). Central MDAF sends the received local model updates to the privacy operation function and gets the aggregated result in cleartext. For the management of privacy related operations, such as distribution of parameters, the privacy management function communicates with the privacy operation functions.

3.6.2.3.2 Explainability for RL-based Control in M&O

Reinforcement Learning (RL), as a branch of AI/ML capable of closed-loop control, represents a key technology to achieve the full automation and optimization support envisioned for the 6G system. In the literature, several works have demonstrated the superior performance of RL-based optimization for the optimization of network parameters, such as the antenna electrical downtilt [VJP22], compared to traditional rule-based approaches. Nevertheless, state-of-the-art RL algorithms suffer from a notable lack of explainability and transparency, making it challenging for humans to comprehend the rationale behind RL agents' decisions and reducing trust in the fully automated system. In recent years, several eXplainable Reinforcement Learning

(XRL) techniques have been developed by the research community to address and mitigate this problem. Some works have also considered telecom applications [RTI+24].

The trustworthy AI/ML-based control sub-enabler, through the integration of XRL techniques in the RL-based automation solution, would enhance the M&O framework with human-interpretable explanations of RL-based decisions in network optimization.

Figure 3-92 shows an overview of RL-based control enhanced with XRL techniques. While the RL agents control the telecommunication network by observing it and applying actions in a closed-loop manner, the XRL techniques generate explanations by analyzing the RL agents and their decisions. These explanations can be consumed by different users, such as AI engineers/developers for debugging and improving the RL agents, or NOC engineers to monitor the network through a dashboard or to query and understand the RL agent's behavior. This design is aligned to deliverable D6.2 [HEX223-D62] (section 3.8.5).

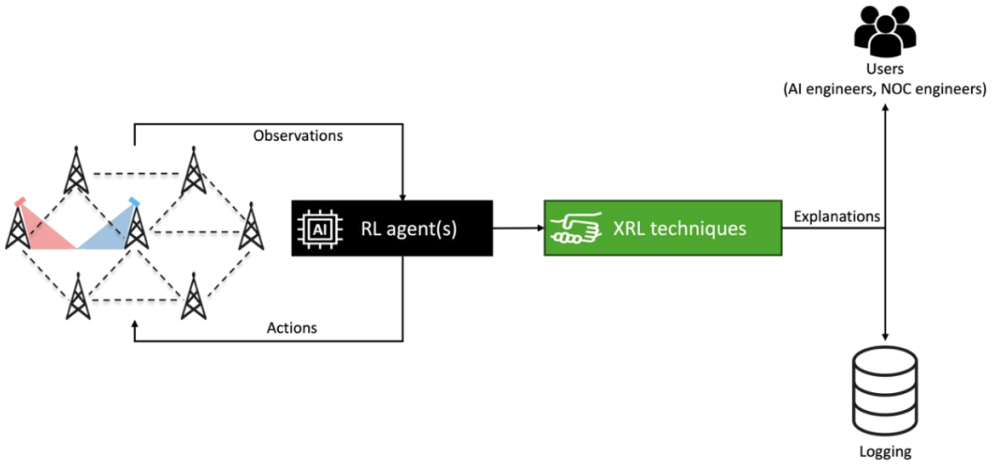


Figure 3-92: XRL integration with RL-based control.

As described in section 3.6.1.2.6, addressing most telecommunication challenges requires a decentralized approach to effectively scale to large networks. Within the RL framework, this requirement leads to the utilization of cooperative Multi-Agent RL (MARL), with multiple RL agents collaborating to optimize a shared objective. This sub-enabler proposes the integration of single-agent and multi-agent XRL techniques.

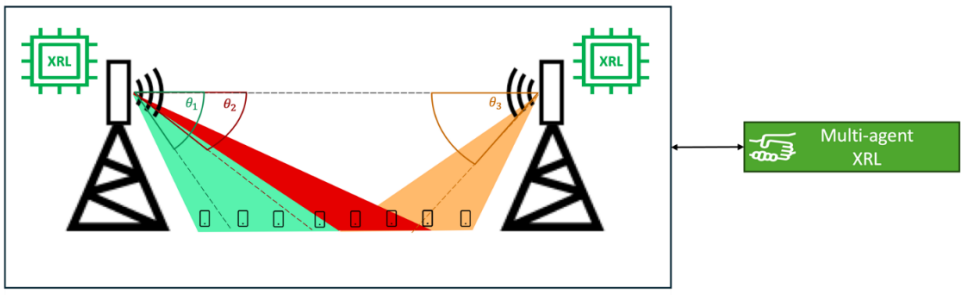


Figure 3-93: Single- and multi-agent XRL in network parameter optimization.

This concept is illustrated in Figure 3-93 in the context of network parameter optimization in RAN (specifically, the optimized parameter is the antenna electrical downtilt). Following a decentralized approach, each antenna is considered as an agent controlling its own parameter. Single-agent XRL techniques analyse each RL agent (antenna) individually to provide explanations about its behaviour and decisions (i.e., parameter updates). Several single-agent XRL algorithms are available in the literature, each addressing different aspects and producing complementary explanations [PVH+20]. A study also applied two XRL techniques – reward decomposition and linear model U-trees – to the antenna tilt optimization problem [RTI+24]. On the other hand, multi-agent XRL takes a higher-level perspective and analyses the impact and contribution of each RL agent in relation to the others. While this problem has been studied less extensively than single-agent XRL in the literature, Shapley values represent a powerful tool to fairly assign credit among the agents [HCD22].

Finally, the same conceptual design can be adapted to Intent-Based Management (IbM) automation. In this case, each RL agent would be responsible for a specific intent while collaborating with the others to jointly

satisfy multiple intents. The contribution plan for beyond SoTA is to experiment with a simulated environment where multiple RL agents control network parameters and cooperate to satisfy multiple dynamic intents. In such experiments, the goal will be to enhance the RL explainability with both single- and multi-agent XRL techniques.

3.6.2.4 Impacted KPIs and KVIs

The main KPIs impacted by this sub-enabler are the following:

- **Reliability:** This sub-enabler allows to develop mechanisms to protect the AI/ML models from adversarial attacks that may affect decisions made by the system and reduce its overall performance.
- **Privacy:** This sub-enabler allows to protect the system against privacy attacks designed to access and leak sensitive data collected and stored in the system.
- **Explainability:** Model explainability can improve the interpretability and transparency of the model and its decisions, and increase the trustworthiness of the AI/ML decisions by providing human readable explanations for each provided output.

The main KVI impacted by this sub-enabler is **trustworthiness** as the solutions developed within this sub-enabler protect the AI/ML decision system against performance degradations and data leaks, and providing human readable explanations for the reasoning of AI/ML-based decisions.

3.7 Enabler 7: Network digital twins creation mechanisms

This enabler aims to develop mechanisms for creating accurate Network Digital Twins (NDT) of the infrastructure and the deployed functions while taking into account the impact of virtualization. At present, there does not exist a unified definition of NDTs from SDOs. A recent draft memo from the Internet Research Task Force (IRTF) [ZYD+24] proposes the definition of a NDT as a virtual instance of a physical network that provides a real-time representation of its state. These four elements (data, mapping, model, interfaces) are the key constituents of the NDT as conceived by this IETF working group. The twin includes a *model* that is constantly synchronized with *data* from the real network via well-defined *interfaces* between them. An interactive *mapping* is thus maintained between the real network and its twinned counterpart. Unlike legacy simulation, the Digital Twin (DT) sends *control* data back to the real network thus allowing for closed loop network life cycle management. Through such closed loop, zero-touch network management is made robust by testing the ramifications of proposed optimizations or reconfigurations, (made via the *intent input* interface), on the digital twin before deciding on the most apt deployment strategy on the real network. However, this may also involve other functions of the M&O framework rather than directly send control signals to the real network. One of the main challenges in designing reliable NDTs is high-fidelity network modelling that accurately captures the intricacies of diverse topologies and scales well to various network sizes.

3.7.1 Enabler design

As illustrated in Figure 3-94, the NDT obtains data from monitoring tools present in the real network through a data collection interface (e.g., through enabler 2). These data are stored in a repository within the NDT and are leveraged for training and updating mapping AI models. The latter can be a mix of machine learning techniques such as supervised, reinforcement learning etc. When provisioning a new application, the resources available can be gleaned by the provider using the capability exposure interface. Any new configurations required to provision the service can be tested on the NDT using the *intent input* interface. Once sane configurations are established by the NDT, they are then updated on the real network via the *control* interface. The key constituents of the enabler are elaborated in the following section.

3.7.1.1 System Components

To enable Network Digital Twins, the system architecture ought to support mechanisms for data collection and storage, as well as NDT lifecycle management operations such as creation, storage, monitoring and update. The system architecture includes the following components:

Real Network: may include various components such as a mobile access network, a transport network, a mobile core, or a backbone. Additionally, it may also be constituted by diverse infrastructures like a data centre network, a campus enterprise network, or an industrial Internet of Things network. This network may span a

singular network administrative domain or multiple administrative domains. It could also include both physical entities and virtual elements such as virtual Switches and NFVs, cooperatively deliver network services.

Data Collection: This interface leverages Enabler 2 to obtain data from monitoring tools present in the real network.

Data Repository: is tasked with gathering and preserving diverse network data to construct different models. This involves gathering and refreshing real-time operational data from various network elements via the data collection interface. Additionally, it offers data services such as rapid retrieval, concurrent conflict resolution, and batch services, along with unified interfaces to the Service Mapping Models subsystem.

Service Mapping Models: provide model instances of network applications to facilitate the programmability of services in line with Enabler 1. Basic models provide the real time representation of the actual network elements, topology and link states. Functional models refer to those that deal with network analysis, fault diagnosis and prediction. Functional models can also be leveraged for lifecycle tasks such as network planning, optimization and maintenance. Should new models be required, they can be generated using data from the repository.

Digital Twin Network Management: this component deals with the life-cycle management of the digital twins. It tracks their performance, resource consumption as well as the models that constitute the twins. The latter may be updated or replaced to enhance the accuracy of the digital twin.

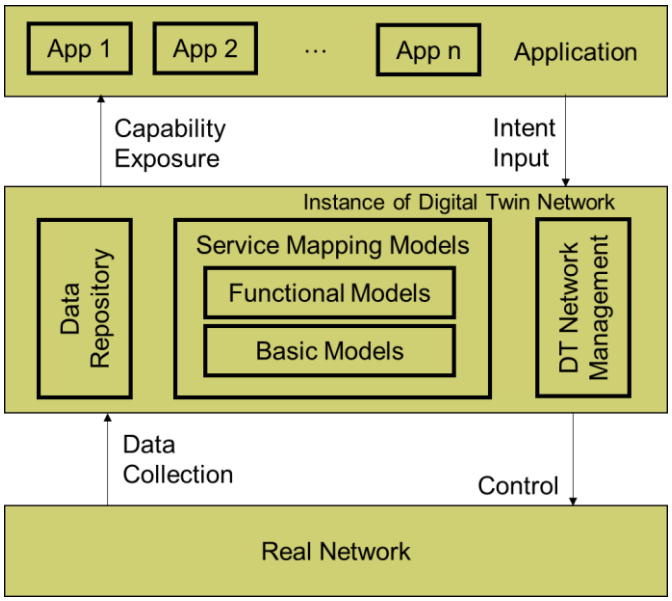


Figure 3-94: IETF proposed Reference Architecture of Network Digital Twin.

3.7.2 Preliminary implementation and early validation results

Among all possible ML mechanisms for creating a NDT, Graph Neural Networks (GNNs) [SGT+09] seem suitable to produce network performance predictions. Graph Neural Networks are a type of neural network designed to operate on graph-structured data, capturing relationships between interconnected nodes. These models are a type of NDT which falls under the Basic Models category in the reference architecture above, due to the fact they capture the essentials of the topology and environment, and can be used to characterize a real network. The work on this implementation will result in certain impacts that would generate bottom-up updates to the reference architecture in the next iteration.

Ferriol-Galmés et al. proposed RouteNet Fermi [FPS+23], a model based on Graph Neural Networks that describes the behaviour of communication networks using flows between nodes, queues at each node, and links between adjacent nodes. It employs a three-stage message passing algorithm to represent how different network components, such as topology, queues, and traffic flows interact with each other, enabling accurate prediction of performance metrics (delay, jitter, packet loss) at flow level granularity. However, RouteNet Fermi was trained on models of physical networks using traffic models and, as a result, cannot account for

several factors pervasive to increasingly virtual networks, such as the effects of multi-tenancy of Virtual Network Functions on a single node, implementations requiring differing Qualities of Service (slicing), or bottlenecks due to CPU, thread usage, or memory.

On the other hand, Whiteaker et al. [WST11] determined that virtualization can add up to 100 ms in round-trip times, with the type of virtualization used having a significant impact. Even in simple scenarios virtualization is found to increase jitter. Tsugawa and Fortes [TF09] further identified that multithreading does not necessarily provide beneficial effects and could in fact be detrimental if the number of threads for VNFs is greater than the number of available CPU cores. Therefore, a revised Digital Twin for virtual networks must account for these effects to remain accurate.

Wang et al. [WN10] determined that packet level flow information is a prerequisite to digital modelling of virtual networks, and created a similar GNN model to RouteNet Fermi to represent a sliced network. However, they use this model only to predict end-to-end latency, and model VNFs independently of the underlying infrastructure.

In this context, the objective of this work is to develop mechanisms for creating Network Digital Twins that can model the effects of virtualizations and VNF behaviour.

3.7.2.1 Applicable datasets / environment

The dataset for creating the NDT is generated using the ***O**bjective **M**odular **Ne**twork **T**estbed in C++* (OMNeT++), which is an open-source simulation software that is commonly used for undertaking network simulation. Due to its freely available nature, it has become a popular alternative to proprietary network modelling solutions such as MLDesigner or QualNet. OMNeT++ makes use of its own topology description language, Network Description (NED) which allows for network specification as elements of a graph, along with a set of flows between elements of the graph. OMNeT++ has been extended to a number of disparate network types such as the automotive Controller Area Network and cellular LTE. This simulation software forms the basis for the Barcelona Neural Networking Centre's BNNetSimulator, a computer network dataset generation and performance evaluation tool. The dataset generated using the BNNetSimulator serves as the ground truth for the development and evaluation of RouteNet Fermi.

3.7.2.2 Modelling virtualization effects

Although the impact of virtualization has been shown in some studies for VNFs [RTC+20], this has not been done for the impact that "thinner" CNFs have on each other in terms of network performance when running on the same physical host. This is necessary to be quantified before it can be included in the network model (Digital Twin). Two inter-related lines of inquiry were pursued to arrive at an improvement on RouteNet-Fermi for the case of virtualized networks:

- Modelling the effects of virtualization on a given network node that utilizes current state of the art container-based virtualization techniques.
- Adapting RouteNet-Fermi's architecture to capture virtualization artifacts and improve performance.

A network test bed was created with two physical servers, one running the TRex network traffic generator [TRE24] and the other, the test server, running a Virtual Machine (VM) with the Docker Engine [DOC24], connected in a ring by means of two 10 Gbps switches (see Figure 3-95).

The VM running Docker acted as a receiver and re-transmitter of the packets sent by the TRex application, by means of a number of identical container-based UPFs. An experiment on this test bed consisted of packet generation and reception and retransmission, given a set number of UPFs, varying the packet rate from 0 to 3 million packets per second bidirectionally, for a 60 seconds per experiment. The number of UPFs varied from 1 to 20 in a logarithmic manner. Each such experiment was repeated 30 times. This framework was used to measure the packet loss and packet round-trip delay.

In order to emulate virtualization environments typically used in production environments and focus on the containerization overhead, the test server was configured with CPU pinning for the Virtual Machine acting as container host, with Single Root Input Output Virtualization (SR-IOV), and HugePages enabled. Further, the eXpress Data Path was used for the UPF containers.

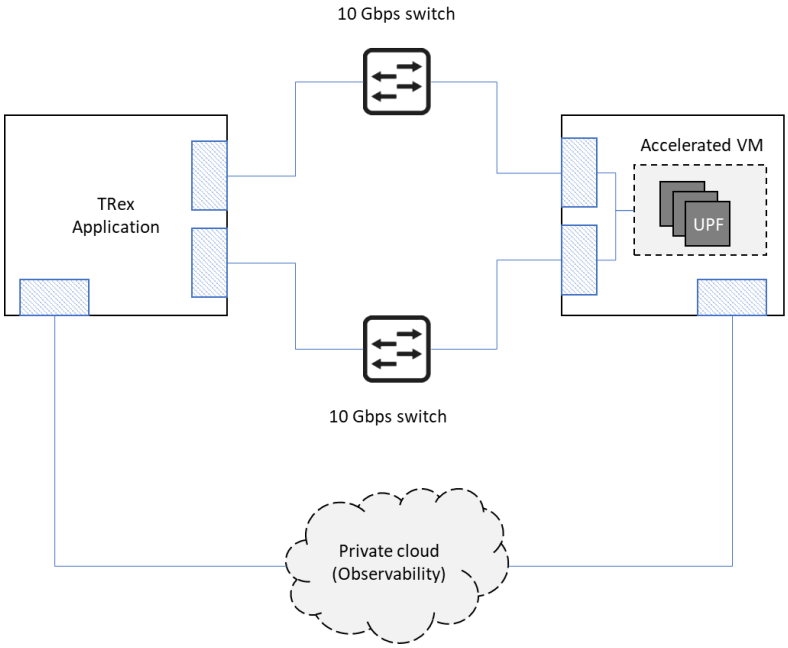


Figure 3-95: Experimental test-bed setup.

Based on experiment results (see Figure 3-96), it was found that with such container-based virtualization there is no large overhead that can be attributed to virtualization. The principal determinant of both packet loss and delay was found to be the packet rate, with an overhead of around 2% additional loss at 3 million packets per second in the case of 20 concurrent UPFs, compared to the case of 1 UPF. The effect on round-trip delay attributable to increased UPF count is around 2 milliseconds, with a maximum latency of around 25 milliseconds, primarily attributable to packet rate.

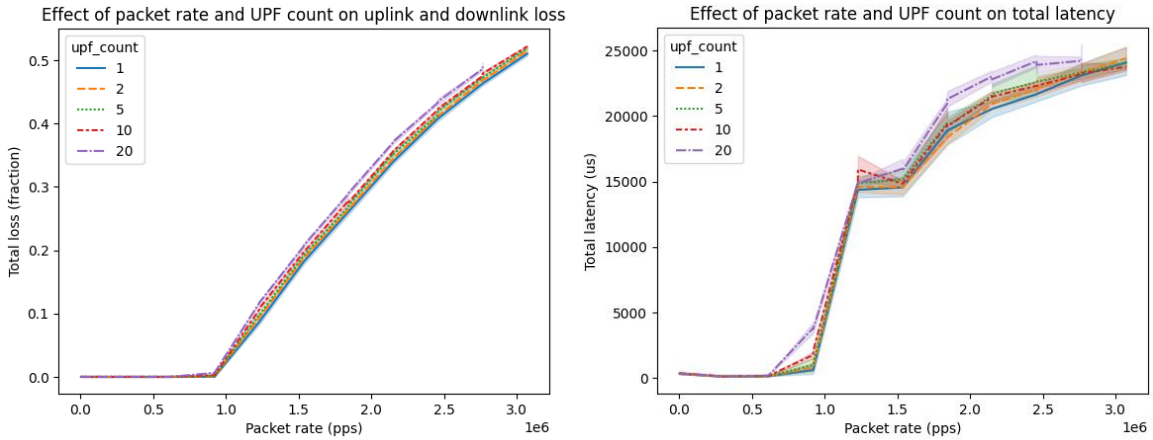


Figure 3-96: Effect of container-based virtualization on packet loss and round-trip delay.

The overhead observed from container-based virtualization is limited compared to the effect of traffic intensity, i.e. packet rate, at a given node. Hence, further analysis is required before a conclusive statement can be drawn. The overhead of 2% found at 20 concurrent UPFs is currently within the margin of error of RouteNet-FeArmi model. Further, alternative avenues of possible research were determined to be retraining the existing RouteNet-Fermi architecture on simulated virtualized workloads or performing hyperparameter optimization to offset the effects of container-based virtualization. A second approach would be the use of other types of Graph Neural Networks to better capture dependencies between network nodes, i.e. Graph Attention Networks (GANs) or Graph Isomorphism Networks (GINs).

3.7.3 Impacted KPIs and KVIs

The following KPIs are impacted by this enabler:

- **Service KPIs:** This enabler provides the mechanisms for creating NDTs that best reflect the state of networks, which means that advanced algorithms for network and service management can be trained in a more efficient manner, which improves their performance and decision quality. This results in improvements for the use case KPIs such as energy efficiency, latency, or bit rate.
- **Automation:** The use of advanced models that are true copies of the infrastructure hosting NFs complements the work on closed management loops. It allows for autonomous prediction of the impact certain management actions will have, thus removing complex policy decision trees made by network operational teams.

The KVIs impacted by this enabler are **trustworthiness** by providing a more accurate representation of the network state and thus improving the performance and accuracy of the decision mechanisms, and **sustainability** as the NDTAI/ML models for optimizing energy consumption.

3.8 Enabler 8: Real-time Zero-touch control loops automation and coordination system

Zero-Touch Closed Loops (CL) are expected to play a primary role in the management of 6G networks. They provide the foundations to bring self-configuration, self-adaptation, and self-optimization capabilities in the network logic. This makes complex control and orchestration tasks fully autonomous in highly scalable, dynamic and multi-technology environments. The automation mechanisms introduced with CLs limit the complexity of the network operation. CLs are able to handle heterogeneous and multi-layer objectives that can reduce the infrastructure OPEX (e.g., maximizing resource utilization, reducing global energy consumption, etc.) while guaranteeing the performance and QoS/QoE level required for the services. This could be achieved through pervasive M&O functions that cooperate following a data-driven approach, often supported by AI/ML techniques and other complementary technologies like Digital Twins and Intent Management. These CL functions contribute to the global M&O procedures and they can be specialized for several scopes. They can automatically update the resource allocation at the infrastructure level, the network configuration and/or the service and application settings to guarantee service continuity, compliance with user intents and continuous re-optimization based on monitored data and target objectives. A set of concrete CL implementation examples will be presented in this section, together with a system for CL governance, implementing dynamic provisioning and configuration of several CL instances.

CLs can be applied to different domains and layers, even with different scopes, e.g., with CLs specialized for a given service, a given network slice or, more broadly, for a given tenant. Multiple CLs can run in parallel, each of them with its own objectives and time scaling, but often operating over the same set of resources. CLs can be characterized by interdependencies where the actions triggered by one of them may impact the decisions of others, or actions from multiple CLs should be executed in a proper order, or more CLs can cooperate together to achieve a common objective in more efficient manner. Moreover, the decisions of concurrent CLs may be in contrast and lead to conflicts that need to be promptly identified and mitigated or resolved through suitable arbitration strategies before actuating their actions. This enabler also addresses these aspects proposing a system for the coordination of multiple CLs to guarantee a consistent end-to-end, cross-layer and cross-domain network automation. The design of the enabler (see section 3.8.1) is strongly aligned with the current work on ETSI ZSM ISG ([ZSM-009-1], [ZSM-009-2], [ZSM-009-3]) and the preliminary implementations are intended to provide concrete examples of ETSI ZSM CLs' models and functions.

3.8.1 Enabler design

The closed loop (CL) approach introduced in D6.2 [HEX223-D62] is based on four stages (Monitoring, Analysis, Decision, and Execution), which are chained together with the support of a transversal Knowledge function for the sharing of common information among the CL stages. The CL concept would be applicable at all the three layers of the Hexa-X-II E2E System Blueprint (ref. Figure 2-2 in Section 2), i.e., at the Infrastructure, Network and Application layers, with CLs that may consume monitoring data from various layers and operate with a cross-layer scope. Each CL starts from the collection and pre-processing of detailed information on target metrics, from service, network or underlying infrastructure layers. These data are then analysed to build advanced insights or predictions on relevant KPIs and/or detect anomalous conditions. In cognitive closed loops, ML techniques can adopt continuous learning mechanisms and adjust the CL behaviour

according to the results of previous actions. Processing the analysis output, the decision stage identifies the actions needed to continuously meet the CL objectives and triggers their execution (e.g., resource re-configuration or network/service lifecycle management commands) on the entities managed by the CL. Enabling a seamless integration with the management system of the mobile network, these stages can be implemented through a composition of cloud-native functions, provisioned and orchestrated dynamically over the cloud/edge continuum of the infrastructure. The management of CL functions is handled through the CL Governance service, delivered through one or more functions in the M&O framework. The following sections provide further details regarding the architecture, functional blocks and workflows supporting this enabler.

3.8.1.1 Internal Architecture of the system components

The internal architecture of a CL includes the CL functions (Monitoring, Analysis, Decision, Execution and Knowledge) and the function providing the mechanisms for its governance, i.e., the CL Governance system. In scenarios with multiple CLs, additional functions for CL coordination are also required. The functional representation of a CL can be implemented with several deployment models, e.g., with CL functions split in multiple components or, vice versa, aggregated in single elements. CLs can be specialized to guarantee different objectives or to operate over different managed entities (e.g., infrastructure resources, network slices, service instances, etc.). As such, several CL instances of different types can be instantiated dynamically to deal with the whole set of objectives defined by the operator or derived from the users' intents. CL functions can be deployed and configured on-demand or initially pre-provisioned and re-configured at runtime. Moreover, they can be shared among CL instances as it happens for VNFs in NFV network services. The CL provisioning, lifecycle management and runtime is handled by the CL Governance function, which includes a mix of generalized and CL-specific logic and procedures. To efficiently handle the CLs diversity, multiple CL Governance functions can exist, each of them specialized for one or more CL categories. Moreover, more instances of the same CL Governance function type can run in parallel for scalability reasons, under the coordination of an upper layer entity where needed.

The list of components with possible technologies for their implementation is provided in Table 3-7. The table refers to generic CLs. A survey of the particular CLs under implementation in the project is provided in section 3.8.2.

Table 3-7: CL components.

Component	Description	Possible technologies
CL – Monitoring	Represents the first stage of the CL and collects data required for the system automation from system itself (e.g., MDAF or an equivalent NF in 6G) or external sources.	Prometheus, ELK, Grafana, Monitoring Platforms developed for enabler #2.
CL – Analysis	Analyses the data collected by the Monitoring stage and derives information on what is happening and or happened in the monitored systems. Can make use of information resulting from external AI/ML process e.g., belonging to the 6G system	TensorFlow, KubeFlow, Seldon.
CL – Decision	Takes decisions based on the information derived by the Analysis service. Such decisions aim to produce possible corrective actions that would project the system towards a desired state. Can make use of information resulting from external AI/ML process e.g., belonging to the 6G system	
CL – Execution	Enforces on the system the decision(s) taken at the decision state, if any. This can include interactions with different management functions (e.g., CSMF, NSMF), NEF, etc.	Orchestrators, SDN controllers, RAN controllers, any other entity that can accept and apply a configuration in the target domain.

CL – Knowledge	Stores data (e.g., configuration) used by the other stages of the CL or by other elements such as 6G system's AI/ML functions. Information stored can also include AI/ML models to be employed at Analysis/Decision stage.	For ML models: TF Serving, Model Registry in MLFlow, Seldon. For monitoring data and events: InfluxDB, MongoDB, MINIO. For configuration and status information: SQL databases.
CL Governance Service	Allows the management of a CL by external entities such as Orchestrators and CL Coordinators. In this regard, it exposes a number of interfaces for CL life-cycle management, configuration, operations, status, and information retrieval.	OSM, Service Orchestration Platforms [HEX23-D63]
CL Coordination Service	Allow the overall coordination logic for groups of registered CLs. It interacts with the various CL Governance to receive notifications related to status, evolution and decisions of single CLs and to send commands for their configuration or activation. Internally, it invokes specialized coordination functions (see Table 3-8) implementing the logic associated to a group of interdependent CLs.	Not applicable.

CL Governance

Figure 3-97 shows the high-level view of the interactions between CL Governance and CL functions (represented in the green boxes) and other potential components of a 6G architecture (represented in the blue boxes) during the provisioning of a CL instance. The picture assumes a single CL Governance Service and a single CL instance for simplicity, but it can be easily extended to scalable scenarios without changing the global principles.

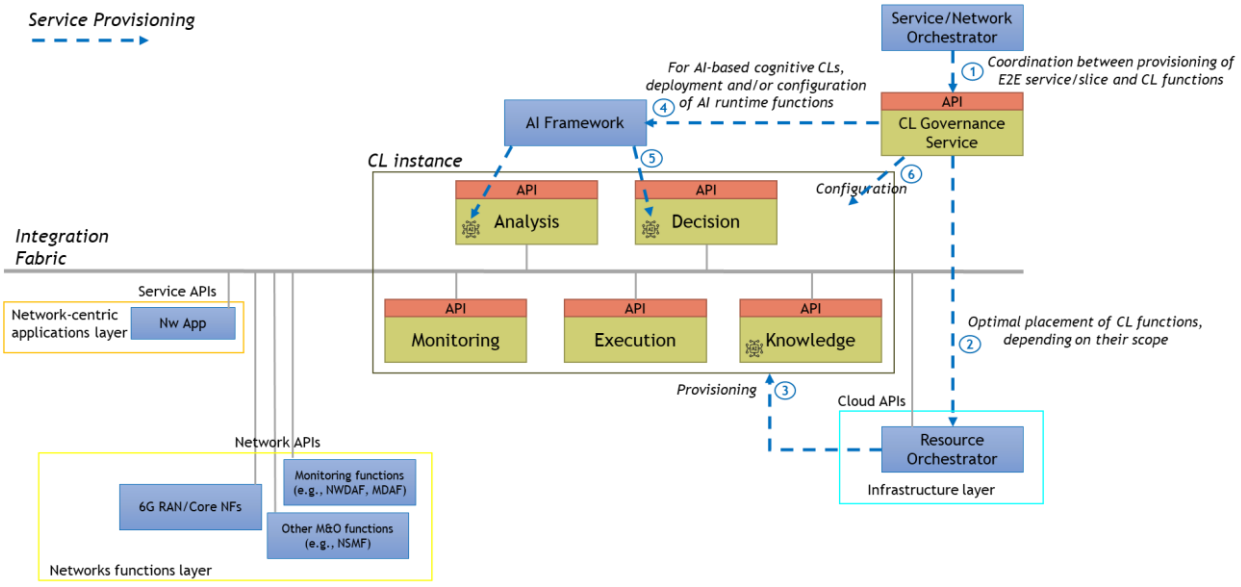


Figure 3-97: CL functions and CL Governance during CL provisioning.

CLs associated to specific service or network slice instances are deployed during the provisioning of their "parent" element, under the coordination of their orchestrator (step 1). The creation of the CL instance is triggered through a request to the CL Governance function, which identifies the CL functions to be provisioned (or re-configured in case of pre-existing ones) and their optimal placement in the target infrastructure (step 2). The actual allocation of the CL functions could be performed through a Resource Orchestrator (see also enabler

5.1) operating at the Infrastructure layer of the Hexa-X-II Architecture Blueprint (step 3), triggered by invoking the Cloud APIs exposed by this layer.

It should be noted that the CL functions, depending on their scope, managed entities and required input metrics, shall be able to interact with some entities or functions running at the Infrastructure layer, Network function layer, or Network-centric applications layer of the Hexa-X-II Architecture Blueprint. This could be enabled through the consumption of the Cloud APIs, Network APIs and Service APIs, respectively, for both monitoring and configuration purposes. The communication is supposed to be mediated through an Integration Fabric, i.e., a possible implementation of the Management Capabilities Exposure Framework (see enabler #3 in section 3.3), which provides the required mechanisms for access control, security, message brokering, tracing, etc.

In case of cognitive CLs based on AI/ML techniques, the CL Governance may interact with the AI Framework to request the deployment and/or configuration of the required AI functions (step 4). These AI functions can be considered as logical components of the CL functions they assist, mostly CL Analysis or CL Decision (even if ML techniques, in principle, can be applied also to monitoring and execution processes). The ML models are assumed pre-trained and they can be selected from the ML models' catalogue exposed by the AI Framework. However, during the CL runtime, a new training can be requested in case of poor performance of the active models. This would allow the cognitive CLs to autonomously adapt to the dynamic environment and to the new conditions generated by applying the CL actions. The target placement for the deployment of the AI functions exposing the ML model (step 5) is CL-dependent. They can either be instantiated within the AI framework in a centralized manner, or be co-located with the associated CL functions or with the Knowledge function, which may have a component dedicated to the exposure of ML models to be shared across CL functions.

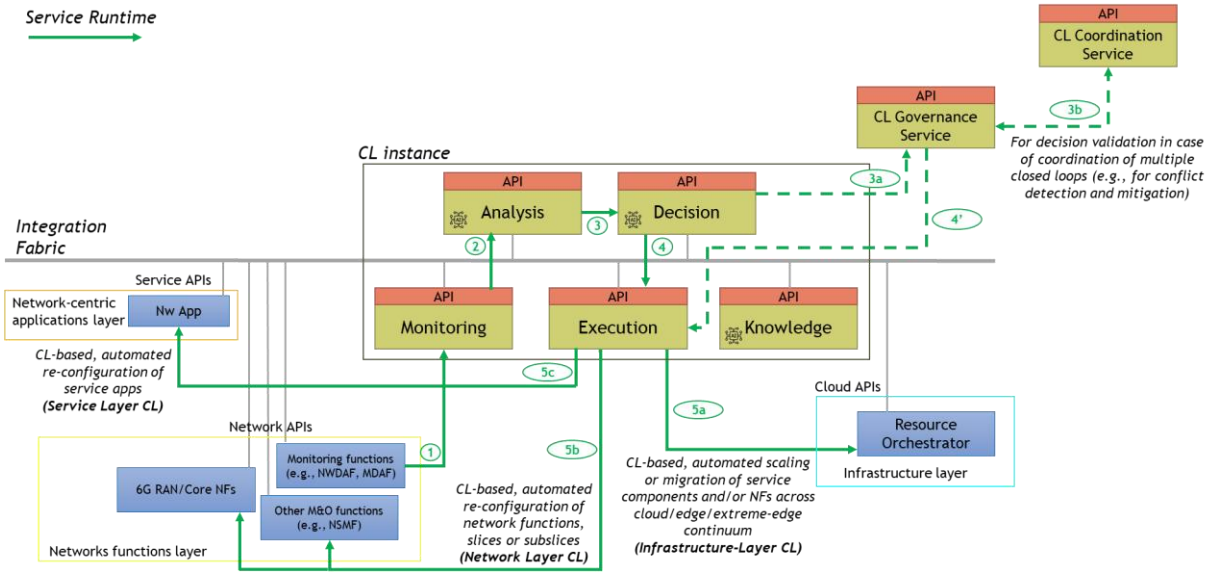


Figure 3-98: CL functions and CL Governance during CL runtime.

Figure 3-98 shows the interactions during the service runtime, when the CL is active. The cycle starts with the collection of monitoring data (step 1) at the CL Monitoring function. The example in the picture assumes to retrieve data from monitoring functions operating at the network layer (e.g., from the NWDAF or the MDAF, or equivalent functions in 6G), but additional sources, from different layers or even external services, can be considered. The incoming data are unified, aggregated and pre-processed at the CL Monitoring function and made available for the following analysis stage (step 2). The CL Analysis function produces an output (statistics, predictions, events, etc.) that feeds the decision stage (step 3), where a set of actions is elaborated depending on the target objective and the configured policies.

In case of fully autonomous CLs, the decided actions are sent to the execution stage (step 4). Here, the CL Execution function would coordinate their actuation interacting with the managed entities at the various architecture layers using the respective APIs. This is represented in the picture with step 5a for an Infrastructure-Layer CL, step 5b for a Network-Layer CL and step 5c for a Service-Layer CL. However, as further discussed in the CL Coordination section below, CLs are usually working in concurrency with other

ones and their decisions may conflict. For this reason, some CLs need to be coordinated together, evaluating the impact of the decisions taken by each of them before permitting their actuation. In this case, the CL is configured to operate in a controlled mode and the decision is not immediately transferred to the CL Execution function but notified to the CL Governance (step 3a) and, from here, to the CL Coordination service (step 3b). The CL Coordination is in charge of guaranteeing the consistency of the actions of multiple CLs, in compliance with the policies established by the operator. In this case it validates the actions proposed by the CL to identify and mitigate potential conflicts with concurrent CLs and it grants the permission to proceed with the action execution, which is triggered from the CL Governance to the CL Execution function (step 4'). It should be noted that in this stage the CL Coordinator may need to interact with other CLs and the conflict detection procedure may result in denying the execution of the proposed actions or modifying them.

CL Coordination

The CL coordination approach has been introduced in D6.2 [HEX223-D62], considering two main different models for the interaction between CLs. Peer-to-peer interaction allows to coordinate CLs operating at the same layer, e.g., in different domains with visibility on their own set of resources, or applied to different services to guarantee service continuity, performance or SLAs. In this case the objective is to guarantee the consistency of the end-to-end resource allocation, without incurring conflicting configurations. The hierarchical coordination model, e.g., based on vertical delegation and escalation actions, is particularly suitable for scenarios where CLs are applied to different layers. For example, a "parent" CL in charge of service automation can be assisted by multiple lower-layer CLs, where each of them receives a delegation to focus on a specific aspect of the service or a subset of its resources (e.g., the distribution of service components on the edge resources, the transport network connectivity, etc.).

The interoperability and the coordination among CLs whose components are provided by different vendors can be considered as a challenge that can be partially mitigated with (i) unified and standard information models for the definition and description of CLs, CL functions and groups of CLs to be coordinated together, and (ii) standard APIs to be exposed at the level of single CL functions, CL Governance and CL Coordination. Standardization work in these aspects is still in progress (e.g., in 3GPP SA5 [28.536] and in ETSI ZSM ISG [ZSM-009-1]) and the information models and interfaces (see section 0) defined in the project and validated through PoCs can contribute to the effort.

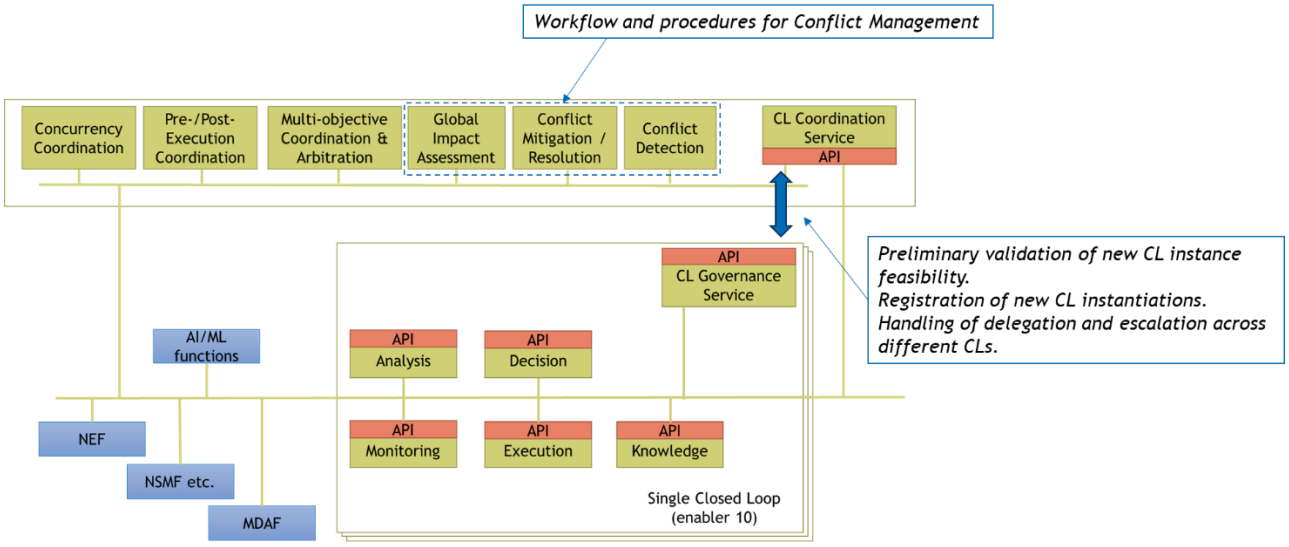


Figure 3-99: CL coordination functions.

Figure 3-99 shows the system components for the coordination of multiple CLs, with a main function responsible for the overall CL Coordination Service and exposing the related external APIs. Internally, it interacts with a set of functions specialized for the various aspects of the CL coordination, e.g., the definition of the order of the actions to be executed by concurrent CLs, the coordination of CLs that can collaborate towards a joint multi-objective target, as well as the management of conflicts. The scope of the single functions is described in Table 3-8. The interaction between the CL Coordination and the single CLs is mediated through their CL Governance function. It should be noted that CLs may belong to different administrative domains. In

this case, the interaction between CL Coordination and CL Governance would be based on an interface between different stakeholders, with the CL Governance function responsible to regulate the proper level of visibility on the internal status of the controlled CLs.

Table 3-8: CL Coordination internal components.

Component	Description
Concurrency Coordination	Specialized coordination function responsible for deciding and regulating the execution order of the actions decided by concurrent CLs, in order to avoid temporary inconsistencies of the overall configuration.
Pre-/Post-Execution Coordination	Specialized coordination function responsible for deciding and triggering commands (e.g., to temporarily pause one or more CL functions, to reconfigure them, etc.) before or after the execution of a CL action. This can help to avoid unexpected and undesired re-configuration decisions due e.g., long convergence times after CL actions execution.
Multi-objective Coordination & Arbitration	Specialized coordination function responsible for coordinating the decisions of CLs with different objectives, possibly combining them by adopting joint decision strategies. In case of contrasting objectives, this function can arbitrate among multiple CLs depending on global policies.
Global Impact Assessment	Specialized coordination function responsible to elaborate the global impact of the decisions coming from multiple CLs before their actual execution. This function may be assisted by network digital twins, for a preliminary evaluation of the consequences of several commands over a virtual copy of the network.
Conflict Mitigation / Resolution	Specialized coordination function that takes as input a detected conflict and tries to find alternative decisions to mitigate or resolve the conflict.
Conflict Detection	Specialized coordination function responsible for the detection of conflicting decisions taken by concurrent CLs and, possibly, the identification of the origin of the conflict. This may help in the mitigation or resolution step.

3.8.1.1.1 Closed Loop common information model

The information model of a CL Descriptor shall contain the elements to:

- Describe the main objectives of the CL. This shall enable the specification of a human-readable description of the CL goal and the definition of a set of statements to be either targeted or enforced by the CL.
- Describe the scope of the CL in terms of the entities (e.g. infrastructure resources, services, network slices, etc.) a specific CL is able to manage. Moreover, a CL shall state the type of management operations it can perform on top of these resources.
- Detail the functions which compose the CL in terms of type (e.g. analysis, decision, execution, monitoring, knowledge), description, configuration parameters required and the templates to be used for the provisioning and orchestration of the software components in a virtualized environment.
- Define runtime policies which can govern the behavioural pattern that a closed loop follows. For instance, these policies may be used to determine the communication mechanisms between the functions of the closed loop, or the decision which shall be tracked in the knowledge base.
- Define the reporting to be associated to the CL, for the generation of periodical or event-based reports with information on the history and the status of the given CL.

A preliminary proposal for the information model of a generalized CL Descriptor is reported in Appendix 8.1.3. Starting from this template, CL-specific descriptors can be derived, e.g., specializing the CL goals depending on the particular type of CL. For example, a CL for SLA Assurance may specify the SLA objectives on the *Goal Condition*, while a CL for KPI Assurance may list the target KPIs, how to calculate them from a set of measurable metrics and the range of acceptable values.

3.8.1.2 Workflows

Figure 3-100 represents the workflow for the provisioning of a CL instance, handled by the CL Governance function upon request of a Service or Network Orchestrator.

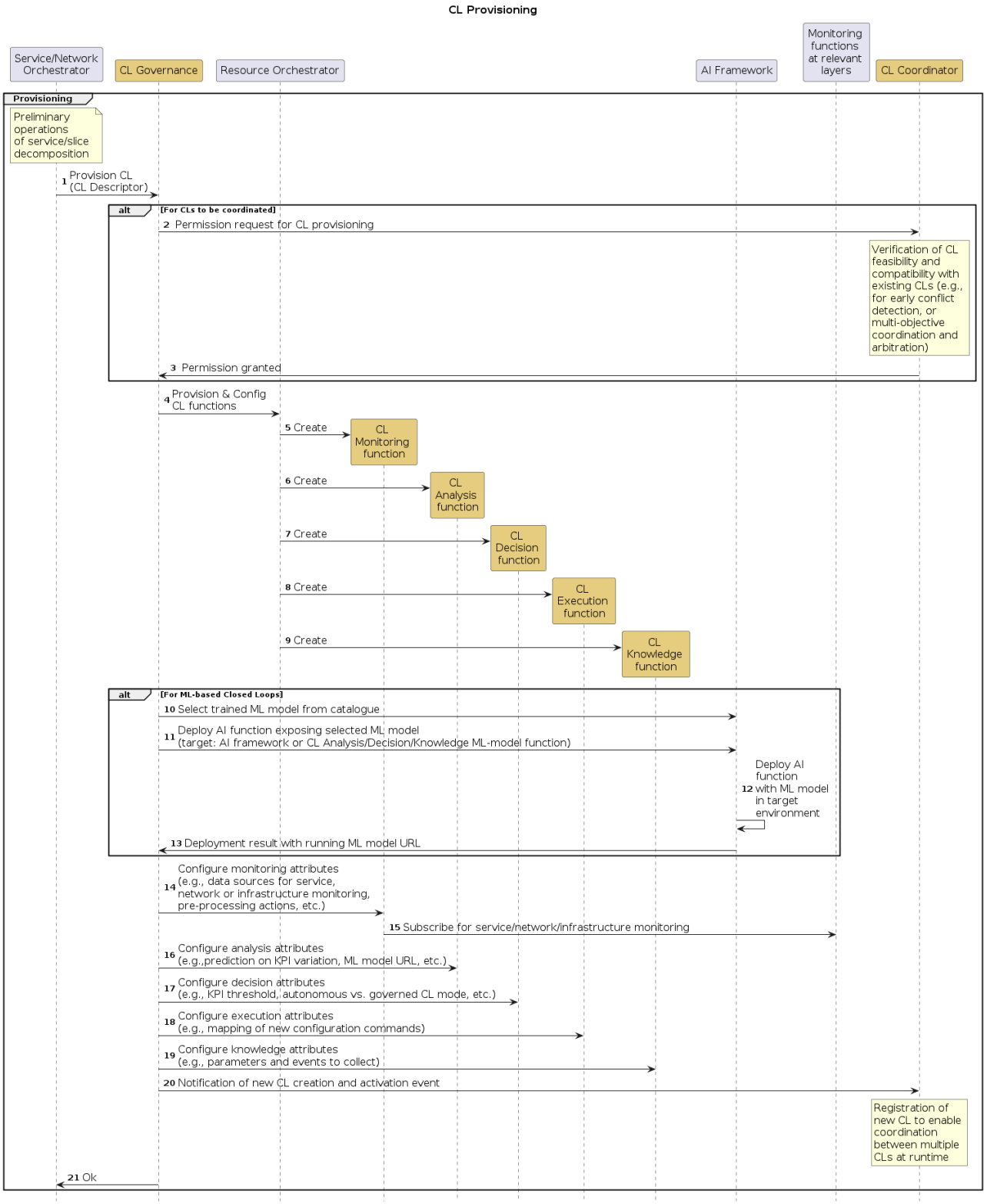


Figure 3-100: Workflow for CL Provisioning.

In case of CLs to be coordinated with other ones, a preliminary feasibility check is performed between CL Governance and CL Coordinator. This step may be used to detect early conflicts with pre-existing CLs that

may prevent the successful activation of the CL or to identify correlated CLs that would be worth be handle or merge together in a wider multi-objective CL.

The CL instantiation proceeds with the creation of the CL functions declared in the CL descriptor. As previously mentioned, one or more functions may be needed for each CL stage and some CL functions can pre-exist and/or can be shared with other CL instances. For AI-based cognitive CLs, the CL Governance interacts with the AI Framework to request the instantiation of AI runtime functions, selecting the related ML model from the available catalogue. This selection can be driven by information defined in the CL Descriptor and additional considerations on the capabilities of the target nodes chosen for the deployment of the AI functions. The following step involves the configuration of the CL functions, setting the attributes defined for each function in the CL descriptor. The procedure terminates with a notification from the CL Governance to the CL Coordinator to notify the creation and activation of the new CL instance, which is registered on the CL Coordinator side.

The workflow for the CL runtime is represented in Figure 3-101, showing the interactions between the CL functions, as well as the two mechanisms (fully autonomous or governed) to trigger the execution of the decided actions. The CL Knowledge function is not represented for simplicity. However, each CL function can access to the information stored there whenever needed during the entire cycle execution; moreover, notifications, events, status changes, etc. generated by the various CL functions are collected and maintained updated there. The notifications are also used to keep a continuous synchronization between the CL instance and the CL Governance.

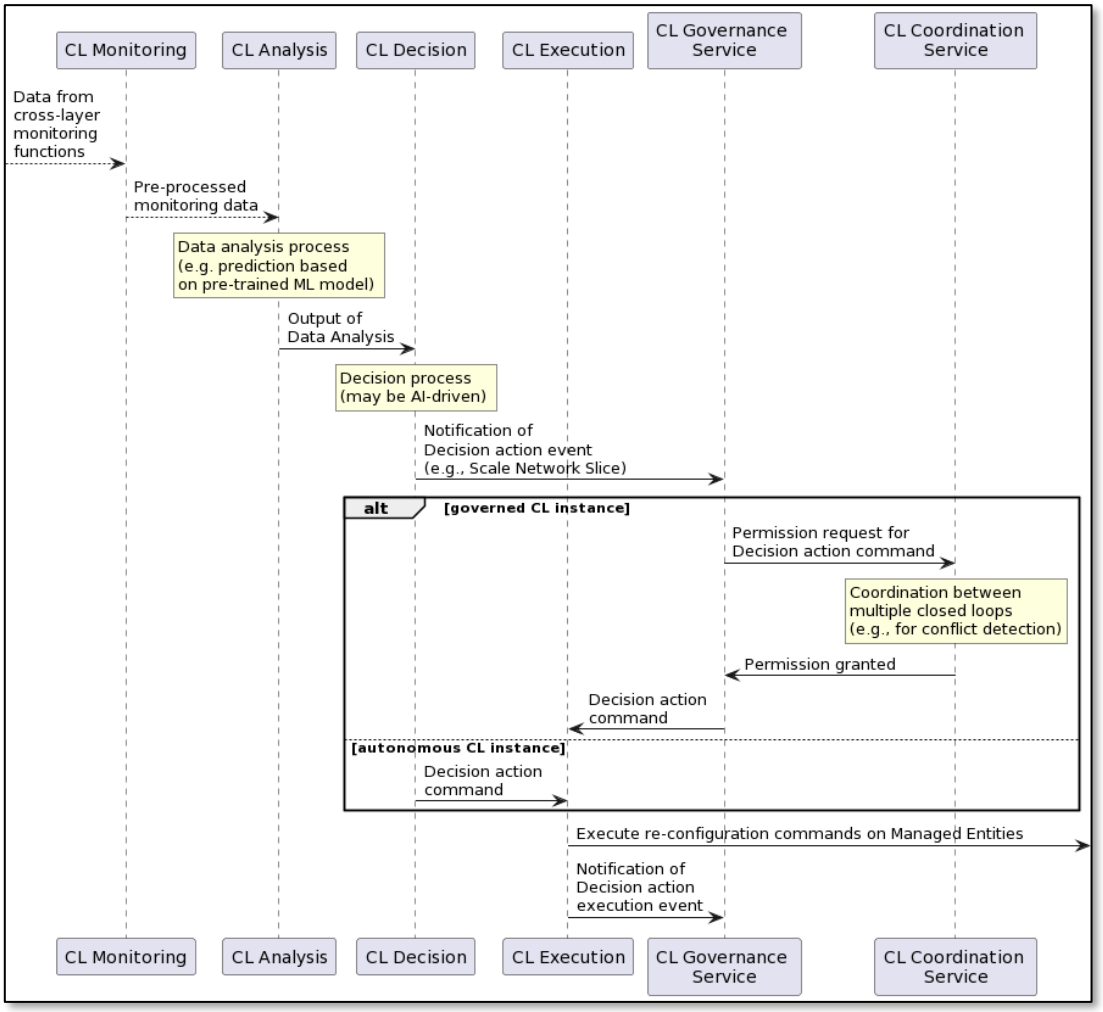


Figure 3-101: Workflow for CL Runtime.

The Closed Loop Coordination Service receives one or more action plans (set of activities that should be executed to guarantee the goals and KPIs of a specific closed loop) from the interacting closed loops. The

Conflict Management component can identify possible conflicts and to determine the combination of actions plans that contributes best to the different coordination goals (see Figure 3-102).

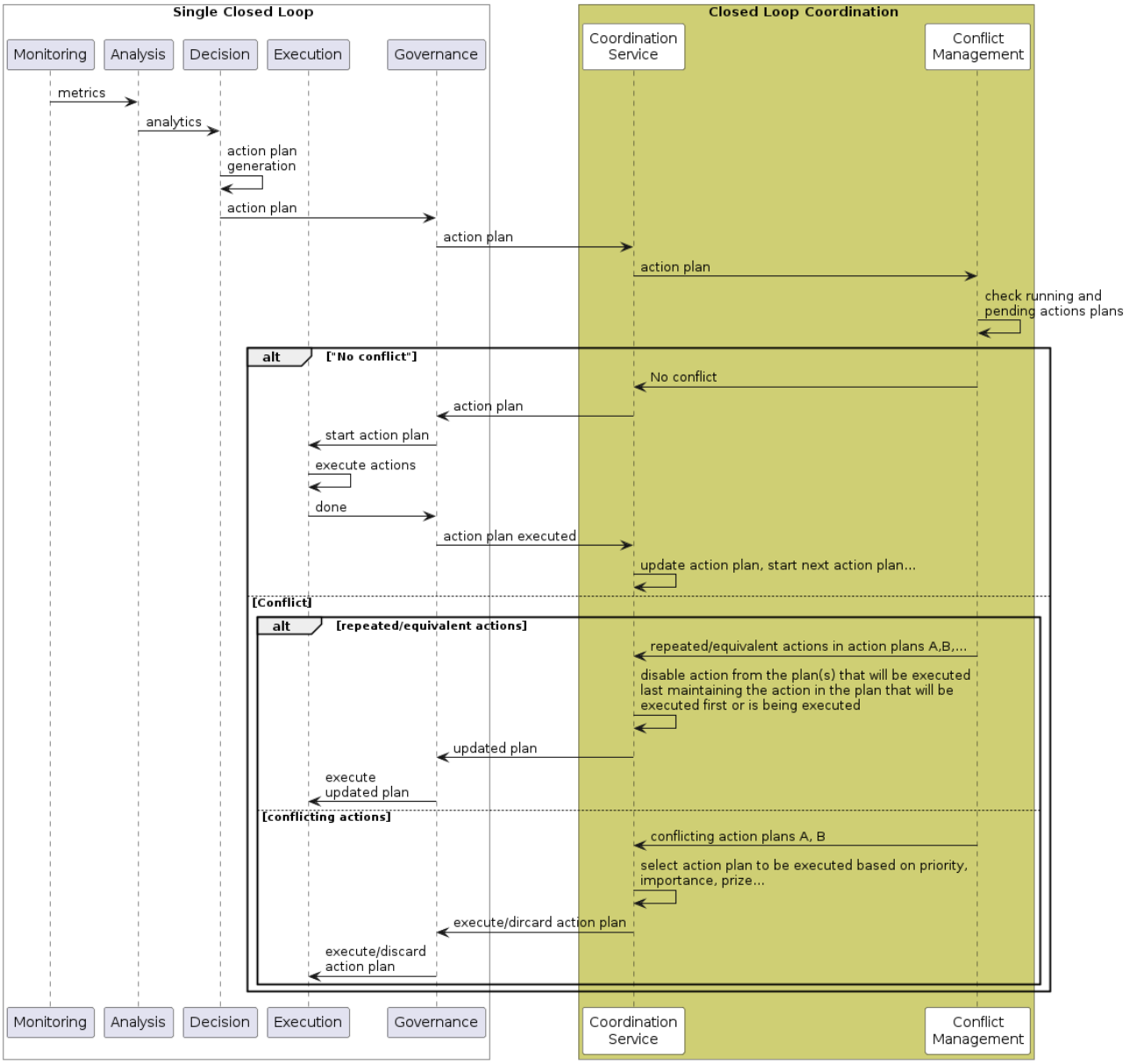


Figure 3-102: Conflict Management Workflow - sequence diagram.

From the notified action plan to be executed and those that could be already running, the Conflict Management checks if there are actions that act over the same resources and could produce conflicting activities, with no desired results. In a simplistic approach, there are three main situations (see Figure 3-103):

- No running or pending action plans, action plans that act over different resources, or action plans that act over the same resources have no conflicting activities: the action plan can be executed.
- Already executed or queued action plans that act over the same resources requesting equivalent actions: action plans should be modified avoiding duplicated activities.
- Action plans that act over the same resources with conflicting activities: coordination service helped by conflict management should select the action plans that should be executed, discarding conflicting plans partially/completely depending on if the plan can be executed without the selected action or not. Selection can be performed following policies based on priority, prize, penalties, etc.

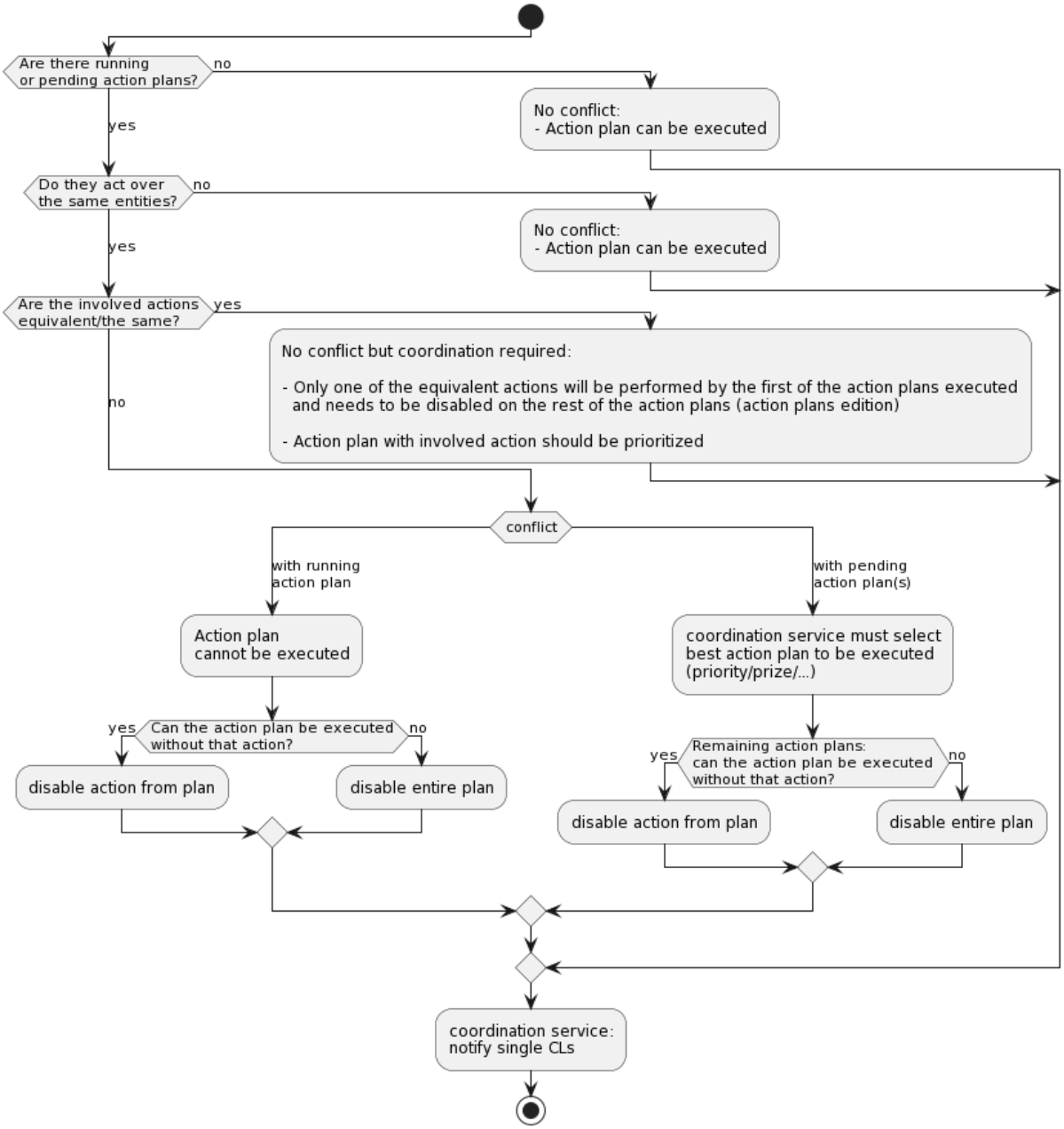


Figure 3-103: Conflict Management Workflow - Activity Diagram.

The interfaces of the CL Governance and Coordination services are reported in Appendix 8.2.4.

3.8.2 Preliminary implementation and early validation results

This section describes the current status of the components under implementation for zero-touch network automation based on closed loops providing, where available, some early validation results of the developed technology or solution. In addition, conceptual solutions of CLs specialized for different objectives (not yet implemented) are reported in section 3.8.3.

In detail, section 3.8.2.1 describes the prototype of the CL Governance and Coordination functions for the orchestration of multiple closed loop instances (with reference to PoC B). Section 3.8.2.2 provides an example of CL operating at the network level for automated path recovery. The conceptual solution presented in section 3.8.3.1 also addresses a network layer CL, but specialized for TSN/DetNet domains. Another particular case of CL, used for continuous validation of AI/ML models and NDTs is reported in Section 3.8.3.2. Section 3.8.2.3 provides an example of implementation for the coordination and conflict management of concurrent CLs, where CLs are prioritized and arbitrated following a penalty-based criteria.

When dealing with CLs operating at the service layer, it is interesting to observe the close relationship with the sub-enablers related to the synergetic orchestration in the continuum (enabler 5). Examples are provided in Section 3.8.4, with CLs for decentralized orchestration (sub-enabler 5.2, cf. Section 3.8.4.1) and service autoscaling in the continuum (sub-enabler 5.1, cf. Section 3.8.4.2), respectively.

3.8.2.1 Closed Loop Governance and Coordination Functions

The preliminary implementation of the Closed-Loop Governance Function is depicted in Figure 3-104. The Closed-Loop Governance Function is a microservice-based framework where each module contributes to implementing the functionalities of the "real-time zero-touch control loops automation and coordination system" enabler.

The Closed-Loop Descriptors Catalogue provides a repository for closed-loops and closed-loop functions descriptors implementing the closed-loop information models described in section 0. The Closed-Loop Descriptors Catalogue allows to on-board, update, delete and retrieve the closed-loop descriptors to be used for the provisioning of closed-loops. The Closed-Loop Descriptors Catalogue has been implemented as a REST Java Spring Boot Application (i.e., that exposes a REST-based CRUD interface) that leverages a PostgreSQL database to persist the information about the closed-loops and closed-loop functions descriptors.

The Resource Allocation module provides the capabilities to compute the placement of the closed-loop functions, once their instantiation is requested, leveraging the pool of compute resources exposed by the Resource Orchestrator's Resource Inventory (i.e., REC-EXEC, described in section 3.5.1.2.1). Given a list of compute requirements (i.e., CPU, memory, and storage) and constraints (e.g., the device type) for each closed-loop function, the Resource Allocation module returns the clusters and cluster nodes (i.e., selected from the ones exposed by the Resource Orchestrator) suitable to host each closed-loop function. A preliminary implementation of the Resource Allocation has been realized as a REST Java Spring Boot Application that leverages a PostgreSQL database to persist the information about the computed allocations of the Closed-Loop functions whose deployment has been previously requested.

The Closed-Loop LCM is one of the major components of the Closed-Loop platform, delivering a preliminary implementation of the closed-loop provisioning workflows described in section 3.8.1.2. The Closed-Loop LCM is responsible for managing the lifecycle of the closed-loops and their functions through three main entities: Closed-Loop Record Manager, Closed-Loop LC Manager and Closed-Loop Runtime Manager. The Closed-Loop Record Manger works as a repository of Closed-Loop instances: once a closed-loop deployment is requested, an associated Closed-Loop Instance is created and used to log all the relevant information of the operations performed for the corresponding closed-loop (e.g., instantiation, configuration, termination, etc.). The Closed-Loop Instance details the status of the closed-loop and its functions, highlighting, for each of the latter, the current configuration, placement and deployment reference (i.e., generated with the deployment of the closed-loop function). The Closed-Loop Instances are updated by the Closed-Loop LC and Runtime Managers in the context of the execution of the lifecycle functions. The Closed-Loop LC Manager is dedicated to the management of the instantiation and termination workflows of closed-loops. To perform the provisioning of closed-loops, after having verified the feasibility of the provisioning of the closed-loop itself through the Closed-Loop Coordinator, the Closed-Loop LC Manager retrieves the corresponding Closed-Loop Descriptor and Closed-Loop Functions Descriptors from the Closed-Loop Descriptors Catalogue; then, for each closed-loop function, the compute allocation and placement are requested consuming the interface of the Resource Allocation service. The instantiation and termination operations of closed-loop functions are performed through the Resource Orchestrator's Service Deployer and the AI Framework; the latter is involved in case of AI-based functions. Once instantiated, a closed-loop is registered to the Closed-Loop Coordinator. Since the instantiation and termination operations are long-running tasks (i.e., considering also the health-check performed for each closed-loop function status), the Closed-Loop LC Manager dispatches such executions to a dedicated thread-pool. The Closed-Loop Runtime Manager is responsible for the management of runtime operations performed on deployed closed-loop functions, providing the capabilities to start, stop and configure them; in particular, the configuration and re-configuration of closed-loop functions are performed by the Closed-Loop Runtime Manager after the functions have been deployed by the Closed-Loop LC Manager and when explicitly requested by higher-level entities (i.e., such as closed-loops coordinators). The Closed-Loop Runtime Manager also takes care of listening for managed closed-loop functions events

(e.g., closed-loop decisions to be validated, closed-loops escalation, etc.), received on a dedicated internal Kafka message broker, to forward them to the Closed-Loop Coordinator.

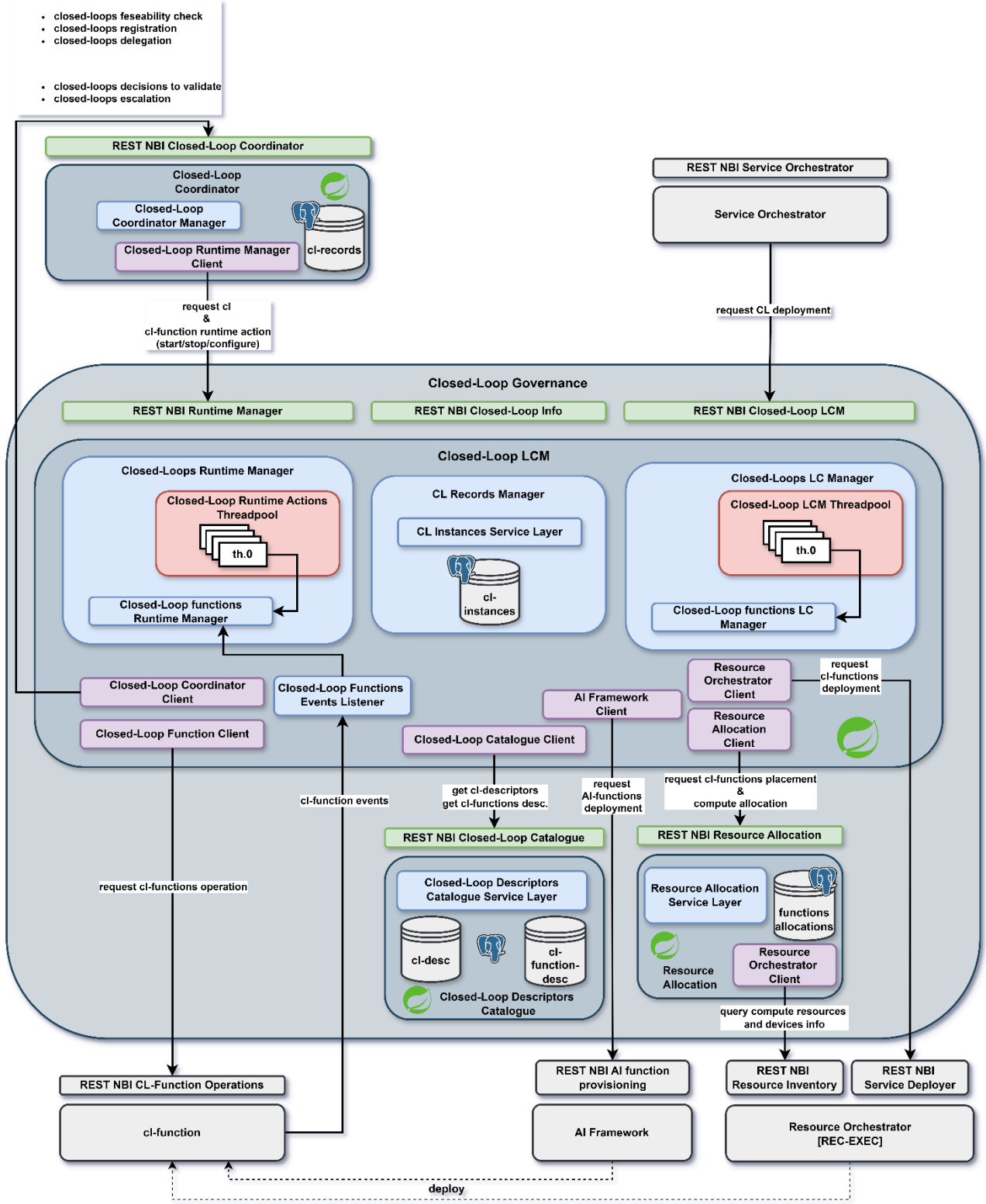


Figure 3-104: Closed-Loop Governance Function software architecture.

Like in the Closed-Loop LC Manager, long-running tasks (e.g., configuration of closed-loop functions) are dispatched to a dedicated Thread-Pool. The Closed-Loop LCM has been implemented as a REST-ful Java Spring Boot Application that leverages a PostgreSQL database to persist the Closed-Loop Instances. The REST-based APIs exposed by the Closed-Loop LCM are consumed by Service Orchestrators, to request the

deployment of closed-loops, and by the Closed-Loop Coordinator to request runtime actions to be performed towards specific closed-loops and closed-loops functions.

Together, the Closed-Loop Descriptors Catalogue, Resource Allocation and Closed-Loop LCM services realize the Closed-Loop Governance Function.

The Closed-Loop Coordinator is responsible for the coordination of closed-loops taking care, as described in the previous paragraphs and detailed in section 3.8.1.1, of providing the capabilities to check the feasibility of the provisioning of new closed-loops, maintain the records of instantiated closed-loops (i.e., exposing APIs to register the latter and consumed by the Closed-Loop LCM) and validate the closed-loops decisions (i.e., events) forwarded by the Closed-Loop Runtime Manager identifying the best runtime action to request to the Closed-Loop Runtime Manager itself (e.g., stop or activate a closed-loop/closed-loop function, re-configure a closed-loop/closed-loop function, approve/deny the occurred closed-loop decision, etc.). An early implementation of the Closed-Loop Coordinator has been carried out developing a REST-ful Java Spring Boot Application that leverages a PostgreSQL database to persist the registered closed-loop instances. The development of the internal components and coordination logic is currently in progress.

In the context of the PoC-B, the Closed-Loop Governance Function is being used for the lifecycle management of the autonomous closed-loop (i.e., closed-loop decisions not validated by the Closed-Loop Coordinator) associated with a network monitoring application deployed in a cobot (i.e., worker node of a dedicated Kubernetes Cluster). Such a closed-loop has the goal of migrating the application (i.e., in another cobot) when the battery level of the cobot hosting the network monitoring application goes below a predetermined threshold. Three functions are involved in the closed-loop: analysis, decision and execution. The analysis function continuously fetches the real-time battery level information of the cobots from the Monitoring Platform (section 3.2.2.3) and sends them to the decision function through a closed-loop dedicated Kafka message broker. The decision function simply compares the received battery level of the cobot hosting the application with the threshold, and if the battery level is below the threshold, it sends a request (i.e., through the dedicated Kafka broker) to the execution function to start the migration. Upon receiving a migration request, the execution function forwards such a request to the Service Orchestrator to perform the migration action. The involved closed-loop functions are deployed by the Resource Orchestrator' Service Deployer in the PoC-B dedicated Kubernetes cluster avoiding the cobots worker nodes [PBM+24].

As reported in section 3.5.1.2.1, the average time needed for the deployment of the three closed-loop functions, from the Service Orchestrator to the functions up and running in the PoC-B Kubernetes cluster (i.e., RUNNING pods' status), is ~16 seconds.

3.8.2.2 Specialized CLs: Automation of Transport Network Slices

This section describes an ongoing implementation of a CL-based system for service assurance in transport network domains. The design of CL functions for automated re-configuration of Transport Networks, e.g., for path recovery due to failures in network nodes or links, is based on extensions to the ETSI TeraFlowSDN (TFS) [TFS24] controller, as depicted in Figure 3-105, with new or updated components. The CL Governance function is implemented as an external element, enabling the management of different types of closed loops, not necessarily associated with the transport network and the SDN controller itself. After the provisioning of a network slice in the transport domain, the Transport Network - Network Slice Subnet Management Function (TN NSSMF) triggers the creation of the closed loop interacting with the CL Governance function. This, in turn, instantiates (step 1) and configures (step 2) the new closed loop activating the related TFS modules and/or setting their configuration. When the closed loop is up and running, it is registered with the CL Coordinator (step 3).

The CL Functions can be mapped to the following TFS components:

- the CL Monitoring Function is implemented through the TFS Monitoring\Telemetry component;
- the CL Execution Function is represented by the TFS Slice component;
- part of the CL Decision Function can be mapped into the TFS Policy and PathComp components.

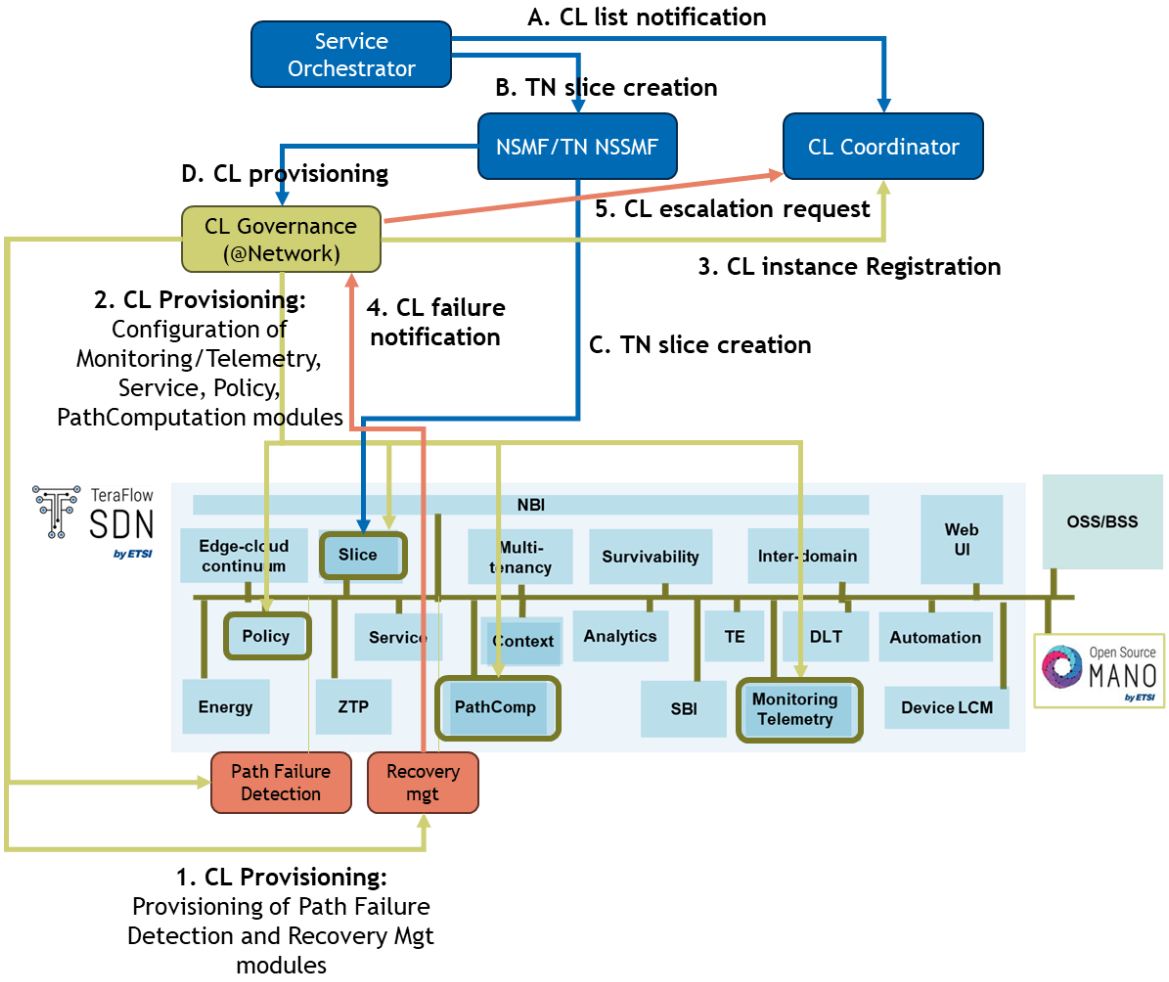


Figure 3-105: Closed-Loop for automation of transport network slices.

The prototype integrates different recovery strategies, which can be selected according to the network slice requirements in terms of availability and resiliency. For example, uRLLC services are associated with protection mechanisms, where both primary and backup paths are reserved and activated at provisioning time. In case of failure on the primary path, the connection is immediately moved to the backup path, which is already established, guaranteeing minimum service down time. However, this implies overprovisioning, with resources that are reserved but not used and this model should be applied only in case of strict constraints on reliability. eMBB services, instead, are associated with re-routing mechanisms, where only the primary path is initially established. In case of failure, a new path must be computed and created, increasing the recovery time but also the probability of service block. Intermediate options may involve the computation of backup paths which are entirely or partially shared among multiple primary paths, with an increased efficiency in resource utilization. This logic is implemented in the new Recovery Management module which selects the most suitable strategy and coordinates all the related procedures interacting with the other TFS modules. Moreover, a new Path Failure Detection component is proposed to enhance the failure detection with prediction mechanisms (e.g., to handle periodical congestion) for proactive re-optimizations. From a functional perspective, the Path Failure Detection can be considered part of the CL analysis, while the Recovery Management constitutes part of the CL decision function. During the service runtime, the closed loop allows to autonomously resolve failures or degradations, limiting service interruptions. However, the recovery procedure may exceptionally fail, e.g., if an alternative path cannot be found. In this case the Recovery Management module generates a CL failure notification which is processed by the CL Governance that, in turn, invokes the CL Coordinator to request a CL escalation procedure. This will be handled through an upper layer CL, working at the end-to-end service level.

3.8.2.3 Penalty-based management of concurrent service CLs

This section presents a solution to handle conflicts between concurrent service closed loops in an intent-based system.

Conflict resolution in an autonomous network is defined as reaching an agreement that satisfies all the conflicting intents and KPIs. However, sometimes this agreement does not fulfil all the intents involved, when this scenario happens, the conflicts does not have a solution to fulfil all intents and a sub optimal solution for this conflict is proposed. This is a design problem in which multiple conflicting objectives must be handled within a minimum (computation) time and without the presence of well-defined objective functions. In literature, there are different approach to solve such problem including multi-objective optimization with Pareto front [RMV+20], [MGG+18], [BJD+14], game theory and ML methods [MG15], [BMC20], [BMC21], [BSL18].

Multiple intents with differing goals operate across different levels and domains in the network. As with human engineers, without appropriate coordination the set of several CLs will inevitably lead to cross-purpose and conflict in the pursuit of their goals. Hence, one of the main tasks of an autonomous network is to efficiently manage multiple intents with a number of closed loops at operation spanning over different networking domains. 3GPP provides guidelines on deploying automatic functions to optimize several objectives in the RAN with Self Organizing Networks (SONs) to resolve a conflict. Specifically, this approach is based on static assignment, where pre-defined system parameters are mapped to pre-defined metrics and objectives. However, even with few parameters, metrics, and objectives, this static mapping generates a complex graph that requires a high level of expertise to understand the interactions and if there is any conflict. Indeed, autonomous networks are increasingly expected to exhibit the various self-* properties (e.g., self-healing, self-adaptation, etc.). Therefore, to be considered genuinely autonomous, the system should learn from its own experience to adapt to unknown situations without reliance on pre-defined rules. Additionally, such static interaction mapping is not robust to the changes found in a dynamic network, meaning that new services, which require new metrics or objectives, cannot be automatically deployed and require urgent human involvement to do so.

In the proposed solution, an **Intent Management Function**, or intent manager, works as an independent entity within a specified domain of intents. Through its northbound interface, it receives intents from either a human operator or another intent manager and is responsible for fulfilling these received intents. It stores intents as knowledge objects and autonomously makes decisions based on observations of the managed environment. On its southbound interface, it sends intents or actions.

Intent managers can be structured hierarchically, with a top-level intent manager overseeing end-to-end service management, while lower-level intent managers handle specific domains (such as RAN, Transport, Core, and Cloud). Within this hierarchy, the end-to-end intent manager breaks down intents to domain-level intent managers.

The prototype of an intent manager framework comprises two main components: a knowledge base and a reasoning engine. The knowledge base stores various knowledge objects, including information on the managed environment, its current state, intents, expectations, domain models, and expert knowledge. The reasoning engine utilizes this knowledge to make inferences and generate new knowledge. The framework, combined with agents, forms a complete intent manager. The reasoning engine autonomously decides which agents to invoke, based on stored intents, data (both current and historical), and supports forward and backward chaining through inference rules and relations.

In a network driven by intents, the intent manager, consisting of the intent manager framework and purpose-defined software components (agents), plays a crucial role in instantiating closed loops to meet the expressed expectations. Agents assume various roles such as data grounding, action proposers, predictors, evaluators, and actuators. These agents collaborate with the intent manager to fulfil intents through the instantiation of CLs.

Upon receiving new intents, CLs are initiated, with each expectation corresponding to a dedicated CL responsible for its fulfilment. These CLs consist of multiple agents, each assigned specific roles determined by the intent manager framework based on the knowledge available in the knowledge base.

As illustrated in Figure 3-106, Data grounding agents play a crucial role in this process. Their responsibility involves querying the network, processing data, and storing the acquired information in the knowledge base. These agents can gather raw data describing the state of the managed environment. For instance, a data grounding agent utilizing a user plane probe might routinely collect latency per UE (user equipment) as raw data. Subsequently, it processes this data, calculating the average latency per UE over a specific time period. The resulting average latency per UE is then stored as a UE property in the knowledge base. This stored information is then compared to the expectations of the intent to assess the alignment with the current state of the managed environment.

The implementation consists in an Intent Manager system, represented in Figure 3-106, where the solid lines represent exchange of data between components and the dashed lines represent the logical communication flow between agents. The system is composed of agents of different types, as follows:

- **Data grounding agent:** responsible for querying the network, processing and storing the collected data into the knowledge base. The Data grounding agent also collects information about the current status of the monitored KPIs. When these KPIs are not met, an object called *issue* is created in the knowledge base.
- **Proposal agent:** propose actions to achieve the target KPI.
- **Prediction agent:** estimate the effects of *proposal* on the properties before the proposed actions are potentially executed. For every proposed action, a fitness value called *penalty* is targeted. Each intent in the system has an associated penalty formula. The intent penalty expresses the cost of not satisfying or partially satisfying the intent.
- **Evaluation agent:** during the conflict resolution, the evaluation agent computes how the change in KPIs will be reflected in the penalty. Then, the evaluation agent selects those actions that minimizes the predicted intent manager penalty.
- **Actuator agent (s):** responsible for implementing the desired action.

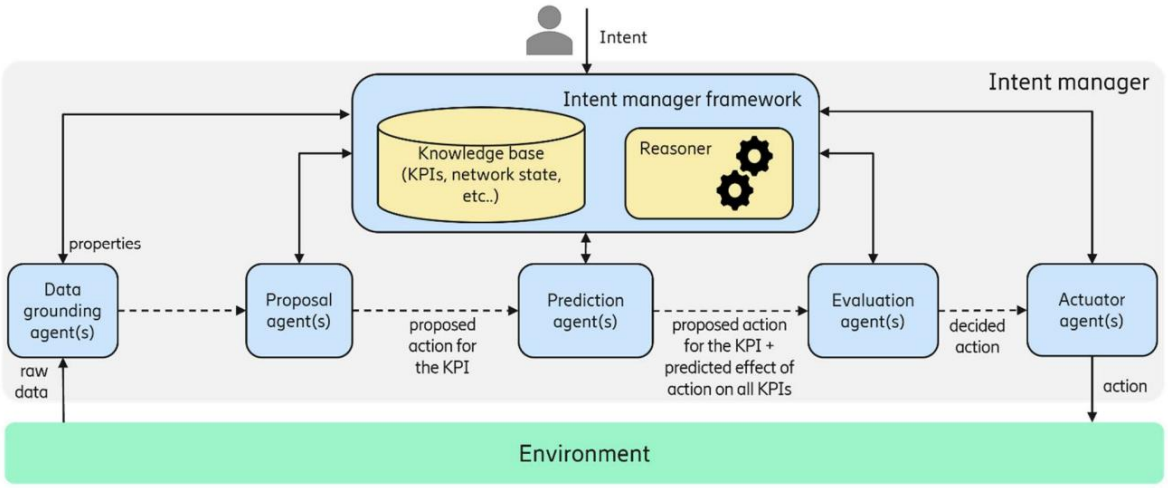


Figure 3-106: Intent manager system.

The mapping between the functional components of the CL architecture defined in section 3.8.1.1 and the software components represented in Figure 3-106 is highlighted in Table 3-9.

Table 3-9: Mapping between software components and CL functions

Software component	CL function	Description
Data grounding agent(s)	CL Monitoring function	Collects data from the network, process and store the information in the knowledge base.
Proposal agents(s)	CL Analysis and Decision functions	Elaborates the collected data defining proposed actions (i.e., decisions generated by a CL operating in controlled mode), which need to be validated through CL coordination.

Knowledge base (within Intent Manager Framework)	CL Knowledge function	Stores information shared among the other CL functions (including CL coordination functions).
Reasoner	CL Coordination function	Coordinates multiple CLs, elaborating proposed actions and interacting with the agents to select the most suitable actions.
Prediction agent(s)	Global Impact Assessment function for CL Coordination	Identify the impact of the proposed actions computing the associated penalties.
Evaluation agent(s)	Conflict Mitigation/ Resolution function for CL Coordination	Evaluates the predicted penalties and takes the final decision about the CL actions to be executed.
Actuator agent(s)	CL Execution function	Actuate the decided action.

Proposal and prediction agents can utilize more advance ML model that can incooperate causal learning and reasoning [MCA+24]. The proposed solution has been implemented and tested with a network emulator (see Figure 3-107), where end-to-end network connectivity is established through entities such as UEs, gNBs, and UPFs. Additionally, a representation of 5G core control plane functions is modelled, including SMF (Session Management Function), PCF (Policy Control Function), and UDM (User Data Management). User-Plane Probes are incorporated to furnish data regarding UE traffic. Furthermore, application sites are implemented and operate on separate virtual machines (VMs). As depicted in Figure 3-107, a dedicated server hosts a conversational video service, while one central and two edge sites run URLLC (Ultra-Reliable Low Latency Communication) services on Kubernetes (K8s) clusters.

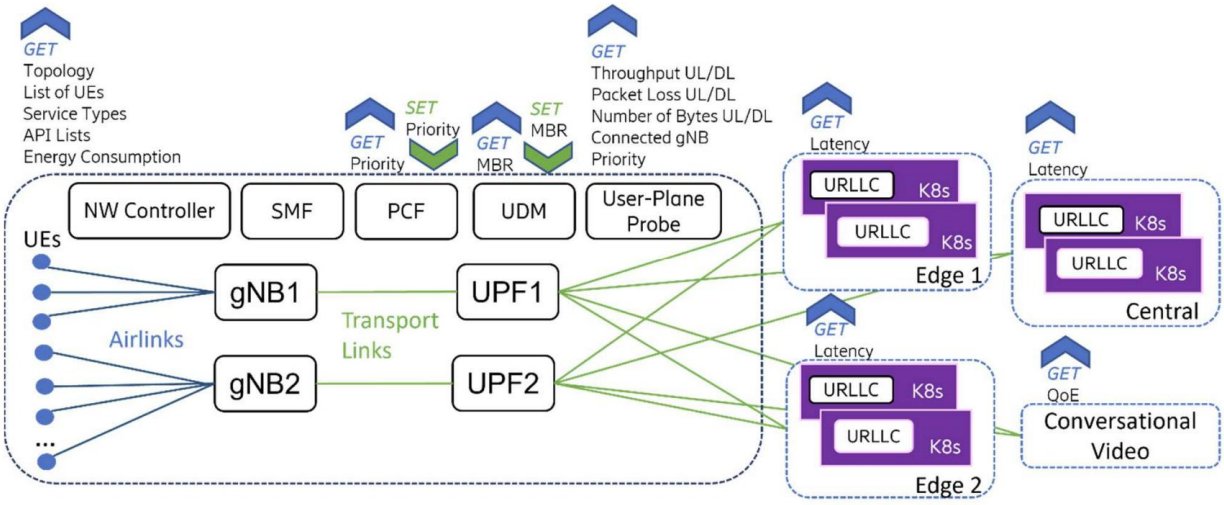


Figure 3-107: Network emulator used in the experiments.

A network controller oversees the network emulator through APIs. This controller not only manages the emulator but also functions as a data source for grounding agents, providing information on the topology, list of UEs, service types, energy consumptions, and API endpoints. The emulator entities are structured as Dockerized web servers utilizing Flask and defined API endpoints, enabling a range of get/set functions. Configuration of both air links and transport links is achieved through the Linux traffic control (tc) tool, responsible for implementing queuing and routing rules.

The validation scenario involves two service instances: a conversational video service instance and a URLLC service instance. The network emulator is configured with a 10 Mbps air link bandwidth and a 20 Mbps transport link bandwidth, with one gNB and UPF each. Figure 3-108 illustrates the observations from this experiment.

At t=0, we introduce the intent for the conversational video service. Since no instance of this service is running, the intent manager initiates the deployment. Once completed, three UEs begin utilizing the service, receiving

real video traffic. The conversational video has a priority of 50, and the observed QoE values surpass the target expressed in the intent (>4.4).

Around t=50, we introduce the intent for the URLLC service instance. The intent manager deploys the application to the central site, and three UEs start using the service with the same priority (50) as the conversational video. However, as traffic increases, the QoE for conversational video users decreases due to resource sharing with URLLC. As the measured QoE falls below the target, the intent manager proposes actions to address the issue: first, to increase the priority for conversational video to 40, where a lower number indicates higher priority. This proposal is accepted, but as the issue persists, the intent manager suggests further increases in priority (to 30 and then to 20). After proposing a priority change to 20, the intent manager predicts that the QoE will likely exceed the target, but the packet loss KPI for URLLC will not be met. Despite this, the evaluation accepts the proposal, considering that the intent manager's penalty with this action is expected to change from 10 to 1. Even though the conversational video intent has been satisfied, the intent manager detects an unfulfilled URLLC packet loss expectation. To address this issue, it suggests potential actions. However, upon evaluating the predicted KPIs, it becomes evident that implementing the proposed actions would lead to an increase in the intent manager's penalty. This is due to the fact that fulfilling the URLLC packet loss expectation would result in the QoE expectation becoming unmet. Consequently, the evaluation rejects the proposed actions, and the system retains one unfulfilled expectation.

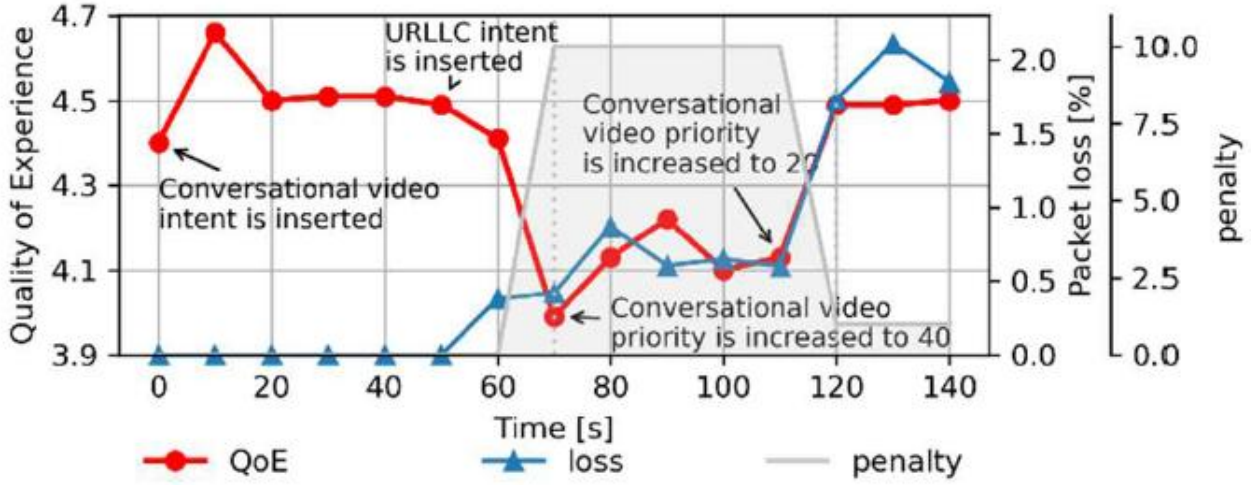


Figure 3-108: Experiment observations – conversational video QoE, URLLC packet loss and penalty.

3.8.3 Conceptual solutions

3.8.3.1 Specialized CLs: TSN/DetNet control

Closed control loops are vital in deterministic networks to ensure the continuous reliability of the network. The goal of the closed loop in this context is to autonomously update schedules, routes and monitoring configuration to maintain the end-to-end guarantees. The closed loop manages a single network segment containing TSN switches or DetNet routers, and the monitoring configuration.

In a single segment, the closed control loop must be able to analyse data in real-time and provide a suitable fix in case a problem is detected. In dynamic networks, closed control loops also provide a way to mitigate suboptimal routing decisions made in the past. By determining and rerouting inefficiently routed flows, the control loop optimizes the number of flows that will be admitted in the future. The analysis phase of the closed control loop is vital, and it must be able to process various metrics, such as time synchronization accuracy, experienced delays, and throughputs. Based on this information, the analysis step might deduce that some flows are at risk due to e.g., a broken link. The decision step will decide if routes and/or schedules need to be updated in certain devices. Furthermore, it must be able to handle incomplete data. In an ideal situation, all network equipment and flows are monitored: every link, every flow, and every router and switch. However, due to the overhead of monitoring, it is typically not possible to obtain all data at every time instance, and as such some data will be missing. The analysis must be able to deal with this by making reasonable assumptions or by inferring data based on what is available. Additionally, the closed loop can decide to update the

monitoring configuration if it concludes that additional monitoring is needed for extra analysis, i.e., it has detected 'blind spots' in the network. The execution step will simply install new routes or schedules on the devices or update the configuration of the monitoring algorithm (cf. Section 3.1.2.3).

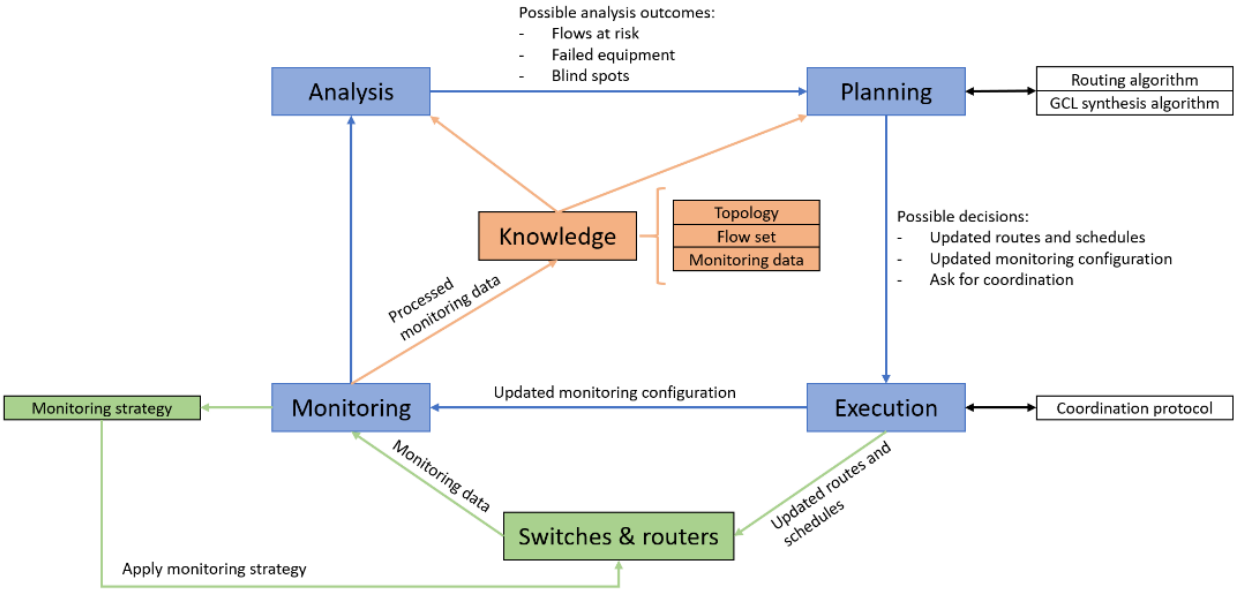


Figure 3-109: Specialized Closed Loop for TSN/DetNet Control.

In a multi-segment TSN/DetNet context (cf. Section 3.1.2.3), an additional communication protocol provides coordination between segments. This is necessary because a single closed control loop will not always be able to provide a fix, therefore requiring coordination with components outside of its own segment. This situation can be expected under high traffic load, when resources are scarce. The resulting closed loop architecture is shown in Figure 3-109.

3.8.3.2 Specialized CLs: Dependencies in AI/ML models and NDTs

5G and beyond networks increasingly rely on AI/ML methods to optimize and automate network and service management. In particular, different AI/ML-based solutions have been proposed to perform dynamic resource allocation and management while optimizing network and service KPIs. Furthermore, the use of NDT has been introduced to provide a realistic representation of network behaviour, and enable safer and more efficient model training. To satisfy the SLA requirements of the network and its services, the optimal performance of the AI/ML models and DTs employed for resource management must be maintained. Hence, those AI/ML should be managed by CLs that continuously monitor model performance and trigger updates as needed.

However, given that the AI/ML models for resource management are trained based on data generated by the NDT, a dependency relationship is established between the model and NDT, which should be taken into account by their respective CLs. Indeed, a performance degradation for the NDT might be an indicator of a future degradation for its dependent models and vice versa. This dependency can also be between two AI/ML models in case of Transfer Learning where a base model is used to train additional models.

To this end, a workflow has been established to perform AI/ML model and NDT lifecycle management while taking into account the dependency relationship between the CLs. Figure 3-110 illustrates an example where the AI/ML models and the NDT are inter-dependent. Namely, Model 1 has been trained with Transfer Learning using the base Model 2, and Models 2, 3 and 4 have been trained based on input from the NDT with different proportions (100, 15 and 50% respectively), where the models and NDT have been trained based on a given network state. Then, a significant change in the state of environment occurs, that was not captured by the NDT or the models, which causes a performance degradation in Model 2. This performance degradation can then affect additional related models at a later stage.

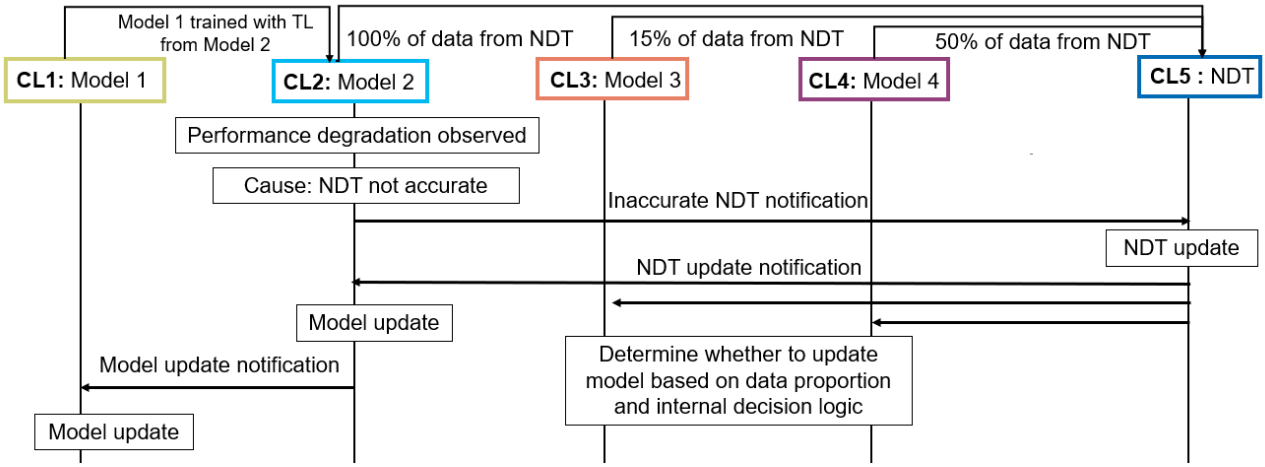


Figure 3-110: Inter-Control Loop dependencies between AI/ML models and NDTs.

In this case, the model dependency relationships can be leveraged to prevent the subsequent performance degradation by triggering updates in the NDT and the dependent models in a proactive manner: when performance degradation is observed for the Model 2, the CL coordination determines that the model needs to be updated, and that the degradation for the model is due to the NDT in CL5 not being accurate anymore. Thus, before updating the Model 2, CL5 triggers an update operation for its NDT to match the new environment state, and a notification is sent to the CLs of all dependent models, namely, CLs 2, 3 and 4. Upon receiving the notification, CL2 triggers a model update based on the newly generated NDT. Additionally, due to the Transfer Learning dependency, this update in CL2 triggers an action in CL1 to also update its model. Finally, based on the proportion of the input from the NDT and in accordance with pre-configured policies or optimized decision logic, CLs 3 and 4 may trigger an update of their respective models based on input from the updated NDT.

3.8.4 Relation with other enablers

3.8.4.1 Specialized CLs for the decentralized orchestration system

The real-time zero-touch control loops automation and coordination mechanisms provided by this enabler 8 could be well integrated as part of the proposed Decentralised Orchestration System (sub-enabler 5.2) described in Section 3.5.2. Beyond the general fact that both enablers rely on a cloud-native microservices based approach, from the functional perspective there can be also a good alignment in which the control loops technology associated to enabler 8 can be incorporated as a value-added complement associated to the M&O mechanisms defined for the decentralized orchestration approach.

Figure 3-111, based in Figure 3-97, illustrates the alignment in what regards the CLs deployment and governance. On one hand, the text blocks in green explain the alignment for the common infrastructure management and service provisioning services introduced in Section 3.5.2.1.1 (i.e., DS, IRS, ISPS, and SRS). On the other hand, the text blocks in red explain how the alignment with the tailor-made per-service MANO mechanisms.

As it can be appreciated (items 1 and 2 in the figure), all the CL functions defined in Table 3-7 would be deployed as part of the network service, as part of the MANO framework attached to the network service itself, that would be designed tailor-made according to each service specific needs. Following the general approach outlined for the sub-enabler 5.2 consisting in deploying the service assurance mechanisms as part of the network services themselves, the complexity of the CL functions to be deployed for each service could vary; e.g., certain complex services could require the CL functions to be implemented with a high level of complexity, while others could adopt a more lightweight approach. I.e., instead of relying on a common and complex CL functions framework part of an MNO-centric orchestration system, the alignment approach in this case would be based on deploying the CL functions with the networks services themselves (as part of their MANO block), and with the functionality adapted to the specific service requirements in each case. Besides, in what regards the communication between the CL functions and other components (e.g., AI Framework, AFs or NFs) it could be done using the components exposed interfaces, in a cloud-native approach, as described in

sub-section 3.3.2, being the Integration Fabric concept proposed from this enabler 8 a possible implementing way.

On the other hand, in what regards the alignment with the decentralised common infrastructure management and service provisioning services, the CL deployment operations would be integrated as part of the DN functionality (item 3 in the figure), while the IRN and the ISPN would also integrate their specific CL instances for implementing the infrastructure discovery mechanisms associated to the Resource Orchestrator (item 4). Since these common nodes (IRN and ISPN) would be deployed in a distributed manner through the network continuum, each of their instances would have its own specific CLs.

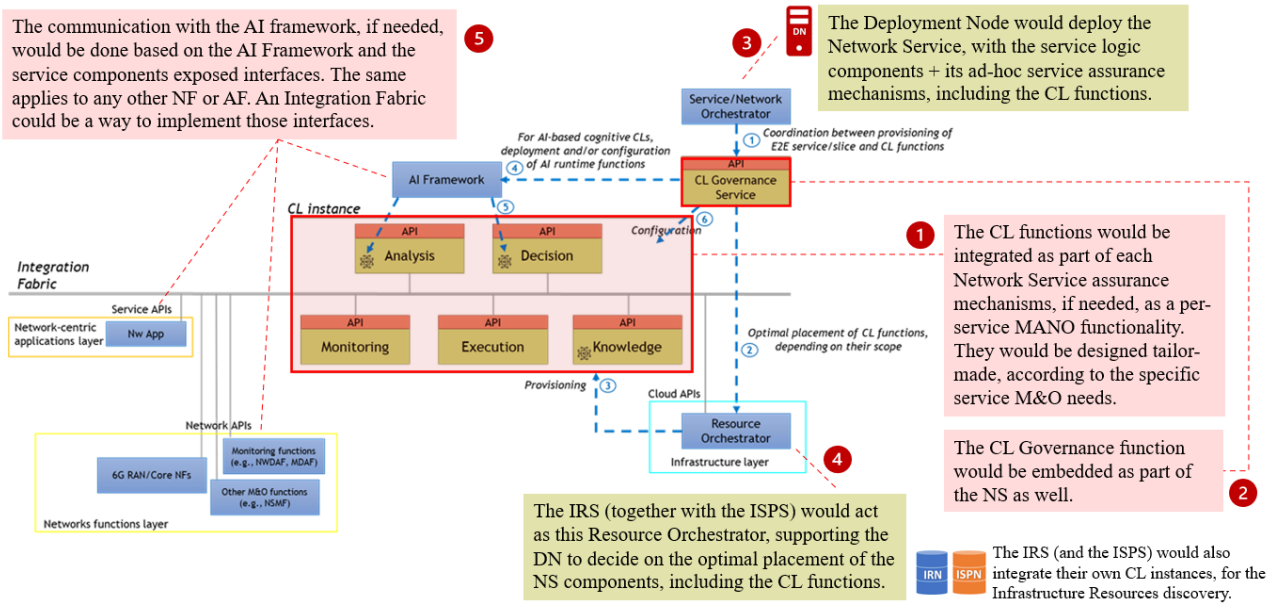


Figure 3-111: CLs deployment and governance. Alignment with sub-enabler 5.2.

In what regards the CLs coordination mechanisms, these would be also implemented as part of the tailor-made per-service MANO mechanisms, but of course, only if multiple CL instances were deployed per service, and the coordination among them were necessary (see Figure 3-112). A special case would be when coordination was required among CLs executing in different domains; however, that could be also targeted relying on the CL Coordination Service exposed interfaces.

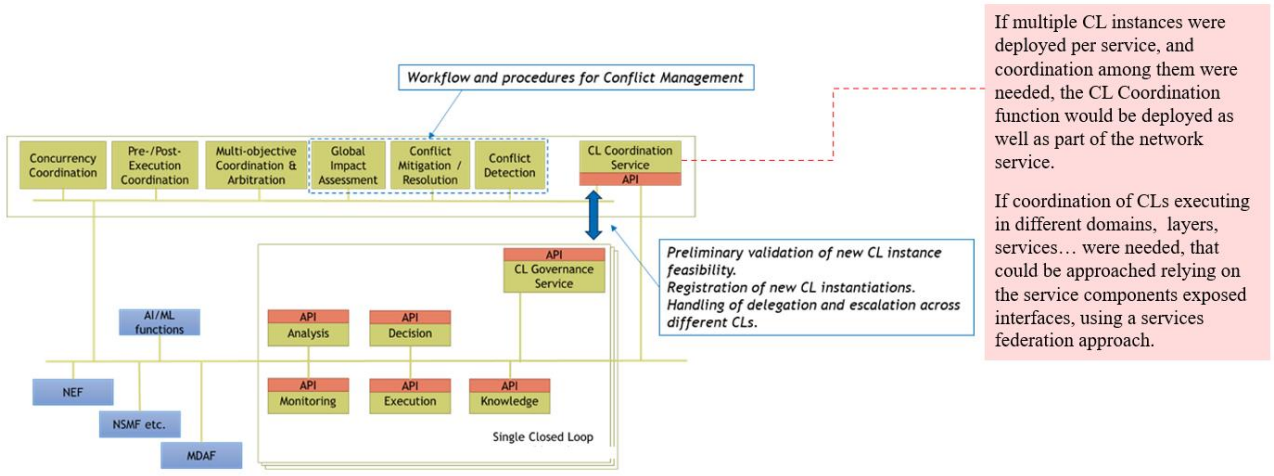


Figure 3-112: CLs coordination. Alignment with sub-enabler 5.2.

However, even though the coordination of different CLs could be approached one way or another, it is considered that, perhaps, the decentralized orchestration approach could be beneficial to reduce those coordination needs to just certain specific cases, since having the service assurance CLs delimited to each single service scope, the coordination complexity would be probably less than delegating on MNO-centric frameworks that would have to coordinate multiple CLs for multiple services in a centralized way.

3.8.4.2 Specialized CLs: Service autoscaling in the continuum

For implementing the autoscaling mechanisms proposed by sub-enabler 5.1, a series of software components are designed, defining the different stages of the closed loop process. These include the following:

- Application registration (CL Governance): An interface for the user to submit the application graph and its requirements in terms of intent or SLOs. The format for this process will follow the application model as described in the previous sections.
- Runtime registry (CL Knowledge): A component accessed in a centralized manner by all other components aiming at providing a synchronized view of the system at any time. It also provides the means to support, implement and validate the use of the defined data model.
- Multi-cluster orchestration interface (CL Execution): A REST API providing access to all components that need to interact with the computing infrastructure for deploying, scaling, migration, or other orchestration purposes.
- Observability server (CL Monitoring): The observability data fusion tool responsible for collecting, fusing and providing orchestration feedback signals to the deployed agents that request it.
- Agent (CL Analysis+CL Decision): An instance of a component that takes up to maintain specific SLOs related to a specific application. Multiple such agents can be deployed for a single application considering different requirements or different parts of the application.
- Agent management mechanism (CL Governance): A component responsible to deploy and manage agents that maintain specific SLOs.
- Agent message-passing mechanism (CL Governance): A mechanism allowing inter-agent communication through a series of channels for support multi-agent orchestration mechanisms.

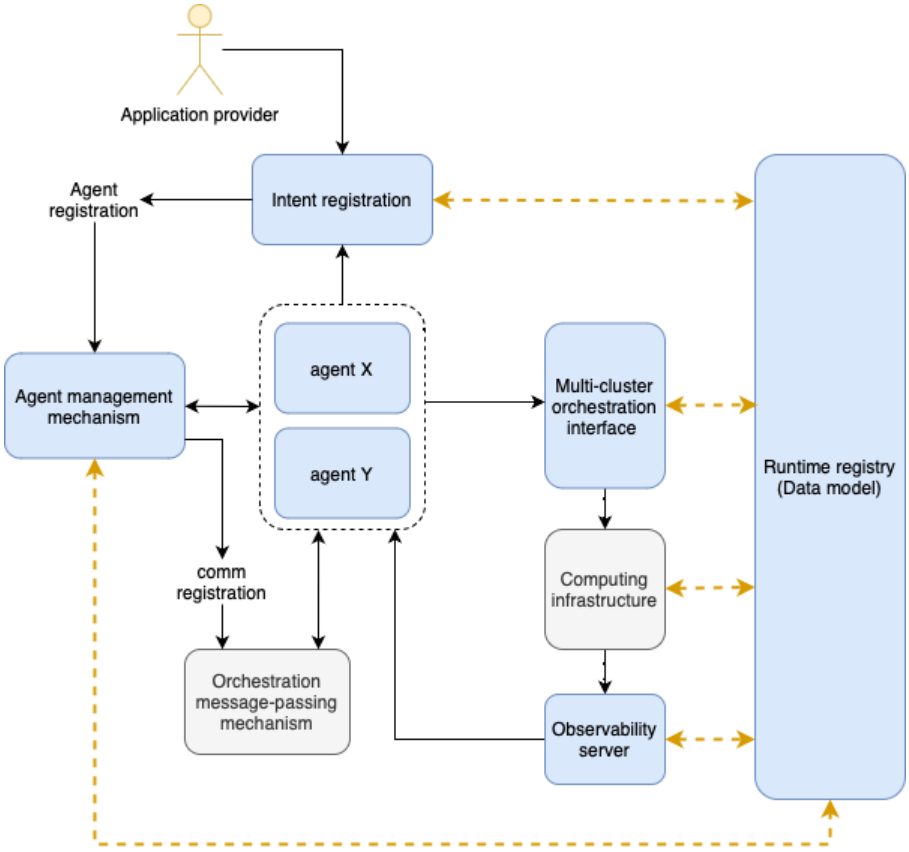


Figure 3-113: Service autoscaling in the continuum.

As shown in Figure 3-113, the closed loop is implemented based on the interaction of the agents managing the different entities in the continuum and the computing infrastructure. The process is initialised with the user intent registration which incorporates the identification of the SLOs and the relevant metrics described in the Runtime registry implementing the data model, followed by the deployment of the corresponding agents using the agent management mechanism. The agents deployed interact with the computing infrastructure using the

multi-cluster orchestration interface for applying the scaling actions indicated by the underlying mechanisms. The applied actions are evaluated based on the metrics collected by the observability server and the loop continues on by allowing the agents to decide again based on the metrics and the messages collected by other interacting agents.

3.8.5 Impacted KPIs and KVIs

The main KPIs considered to be impacted by the adoption of this enabler 8 are the following:

- **User Experience Data Rate.** The closed loops can automatically migrate service components on edge nodes closer to the users and adjust the allocation of network and computing resources according to the performance and the load of the network. This approach allows to guarantee the data rate perceived by the end user.
- **End-to-end latency.** The closed loops can move application components according to the mobility patterns of the users and can adjust the allocation of computing resources in the continuum, through edge and extreme-edge nodes. Service migration and resource reallocation can be driven by predictions related to users' position, processing and traffic load. This approach allows to reduce the end-to-end latency perceived by the user. In detail, the service latency can be improved elaborating the optimal placement of the application components specialized for extreme edge environments, i.e., (i) deciding which components can be offloaded to the edge, (ii) computing jointly network and edge/computing resource allocation, and (iii) selecting the target edge nodes based on computing resource capabilities (e.g., GPU) and current load.
- **Reliability.** The overall reliability of the mobile connectivity is improved through the increased level of network automation, with closed loops that can operate not only in reactive modes (e.g., to solve network failures after their detection), but also in proactive and predictive mode re-optimizing the resource allocation to prevent congestions. This is particularly relevant to guarantee network connectivity in case of volatile nodes that may be deployed at the extreme edge.
- **Service-level reliability.** This is implemented through the same mechanisms of reliability, but with closed loops operating at the service level. In this case, strategies can be based on preventive scaling or replication of service components. Closed loops coordination mechanisms enable the concurrent work of closed loops operating at the network and service layers, guaranteeing their consistency and avoiding conflicting decisions.
- **Availability**. Closed loops can be specialized to guarantee the continuity of the end-to-end service and mobile connectivity, guaranteeing the required level of QoS.
- **AI/ML-related capability.** This is associated with cognitive closed loops, where analysis and decision functions are implemented through AI/ML agents possibly deployed in a distributed environment. The CL Governance function, through its interaction with the AI framework, supports the provisioning and configuration of AI functions to be used in the context of closed loops.
- **OPEX** reduction for MNOs, through automation of M&O operations limiting the human intervention and guaranteeing an optimized usage of the available resources. These mechanisms enable self-configuration, self-adaptation, and self-optimization capabilities in the network logic. This reduces the manual human intervention of the network management and operation, with complex control and orchestration tasks that become fully autonomous in highly scalable, dynamic and multi-technology environments.

Regarding the KVIs, the following are impacted:

- **Trustworthiness**, through improved service and network reliability, as well as the usage of trustworthy AI techniques in cognitive closed loops for network automation.
- **Sustainability**. Energy efficiency can be improved by considering energy constraints or energy-related objective criteria in CL placement and re-optimization algorithms. Concrete examples are the prioritization of target edge nodes powered by renewable energy sources and the selection of robots based on current battery level and AI-based predictions on their battery consumption profile (the latter example is implemented through a PoC).

4 Alignment of M&O enablers with the E2E system blueprint

Table 4-1 summarises the alignment between WP6 enablers and the E2E system blueprint provided from WP2:

Table 4-1: Alignment of the enablers with E2E system blueprint.

WP6 Enablers	Aligned	Update Required
1: Network programmability framework	◆	
2: Monitoring and telemetry framework	◆	
3: Management capabilities exposure framework		◆
4: Security and trustworthiness framework		
4.1: Third-party resource control separation enabler		◆
4.2: User-centric service provisioning system		◆
4.3: Trust management system		◆
5: Synergetic orchestration mechanisms for the computing continuum		
5.1: Multi-agent systems for multi-cluster orchestration	◆	
5.2: Decentralised orchestration system		◆
5.3: Federated orchestration system		◆
6: AI/ML algorithms		
6.1: AI/ML-based control algorithms for sustainability	◆	
6.2: Trustworthy AI/ML-based control algorithms	◆	
7: Network digital twins creation mechanisms		◆
8: Real-time zero-touch control loops automation and coordination system		◆

The rationale for this alignment for each WP6 enabler is set out below. The analysis has been performed following a common methodology for all M&O enablers, identifying (i) the mapping of enablers' system components on existing elements in the blueprint, (ii) possible gaps in features offered by existing components or interfaces and (iii) missing functions or interfaces. The provided feedback has been discussed with WP2 through collaborative activities and it is being used in WP2 to refine the blueprint following an iterative approach.

4.1 Enabler 1: Network programmability framework

This enabler is well aligned with the proposed architecture, being implemented as a specific functionality within the M&O block and affecting the Transport Network Functions block in the Network Functions layer (Figure 4-1).

As a whole, the SDN controller provided in Enabler 1 would be part of that overall M&O block, allowing the network programmability through the Transport NFs in the NFs layer. For example, **Transport NF can be dedicated SDN controllers** deployed on demand or related to Transport Network Slice Subnet Management Function (TN-NSSMF).

According to this, the SDN controller North-bound Interface (NBI) would be provided through the Management capabilities exposure framework (provided by the Enabler 3) and Consumed by the M&O. On the other hand, the SDN controller South-bound interface (SBI) would be provided through the network infrastructure API, depending on network equipment. This will relate **Enabler 1 with infrastructure layer on X-haul**, as depicted.

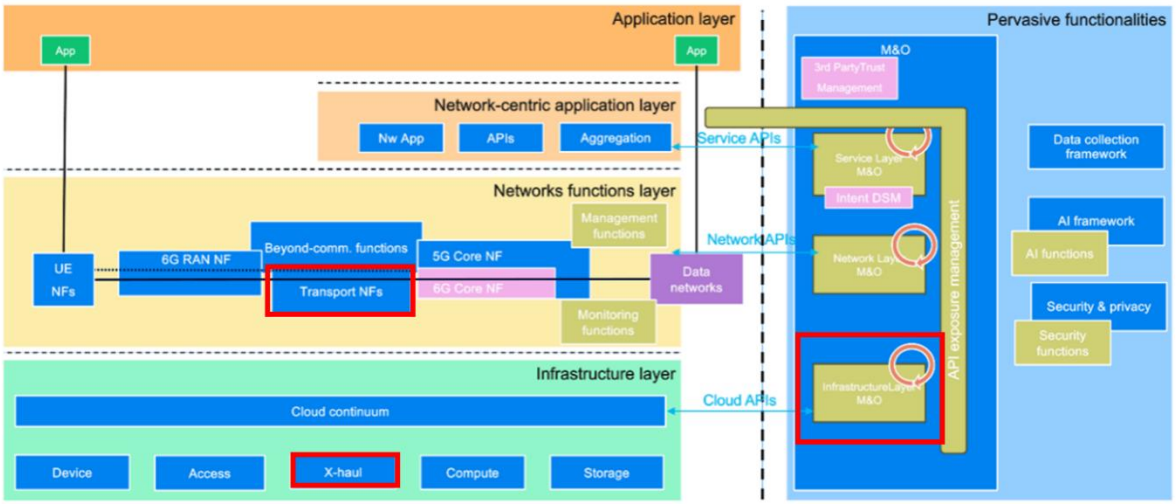


Figure 4-1: Overall alignment between Enabler 1 and the E2E system blueprint.

4.2 Enabler 2: Monitoring and telemetry framework

This enabler is considered well aligned with the proposed architecture, being implemented as a functionality associated with the Data Collection Framework block (in the Pervasive Functionalities layer), the Beyond Communication Functions block (in the Network Functions layer, see Figure 4-2), and the Monitoring Functions (which shall include active monitoring functionalities, such as probes).

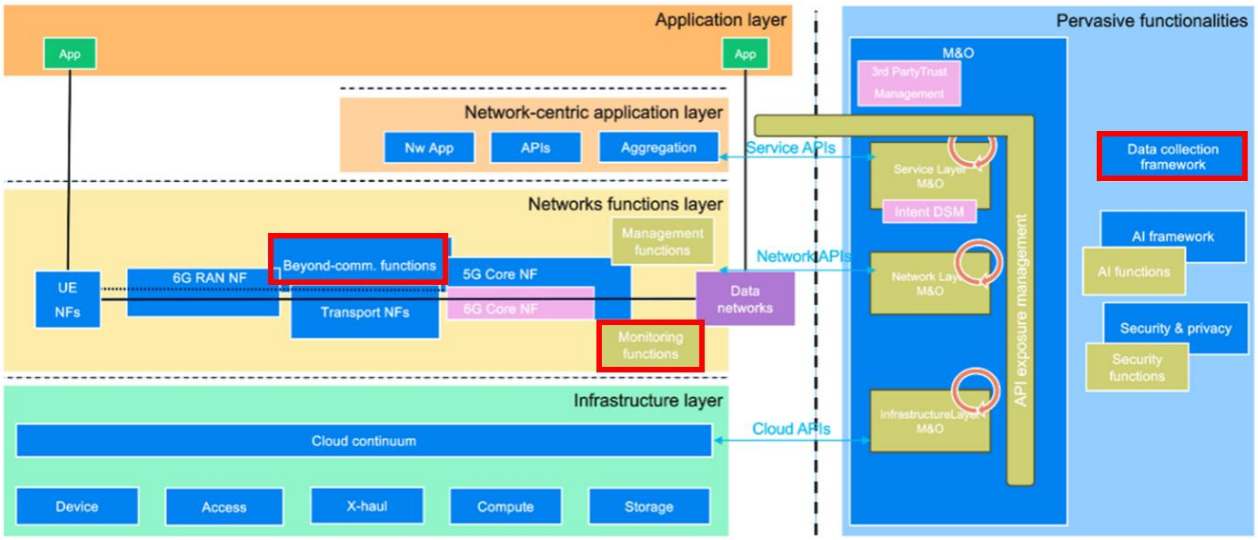


Figure 4-2: Overall alignment between Enabler 2 and the E2E system blueprint.

Based on this, the general approach would be that the Monitoring and Telemetry NBI would be provided through the Management capabilities exposure framework (enabler 3). It will rely on Enabler 3 capabilities to reach end devices, as well as any other location of cloud continuum. On the other hand, an SBI would provide multiple protocols (e.g., SNMP, gNMI, or others).

However, even though this enabler is well aligned, we proposed some recommendation regarding blueprint are:

- The Data Collection Framework should feed the AI/ML training somehow, but this is not represented in the figure.
- The 5G stack already includes the NWDAF (and its related management functions). Evolution of this enabler has been described previously and shall be considered in the scope of blueprint definition.
- Data collection framework naming might be misleading as Enabler 2 provides full support for monitoring and telemetry, thus allowing not only data collection, but also processing and exporting (including storage).

- Cloud-scale monitoring and telemetry data is expected. This shall be represented somehow in the E2E system blueprint.

4.3 Enabler 3: Management capabilities exposure framework

This enabler is currently only partially aligned with the end-to-end (E2E) system blueprint and requires specific adjustments for integration in the blueprint in future iterations. Notably, the Management Capabilities Exposure Framework developed in Hexa-X-II draws inspiration from the Integration Fabric concept [ZSM-002]. The primary goal is to encompass the functionality within its scope.

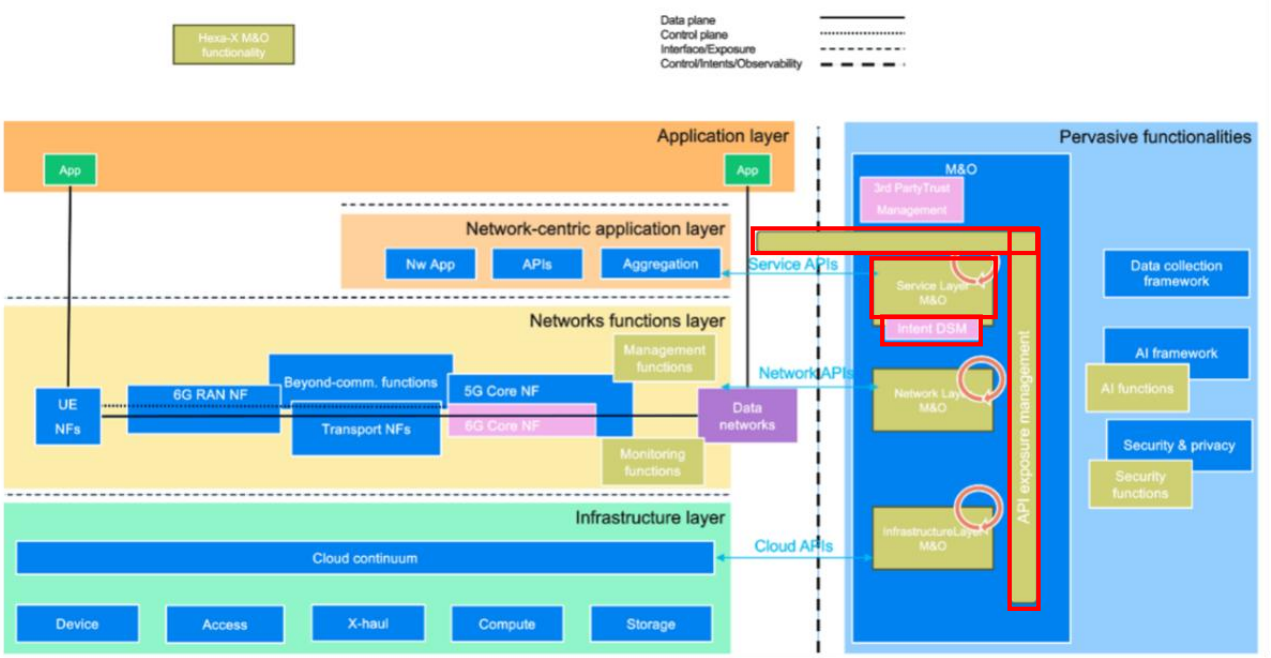


Figure 4-3: Overall alignment between Enabler 3 and the E2E system blueprint.

The Integration Fabric serves two main purposes:

1. It provides connectivity and reachability within the M&O system, particularly within the WP6 enablers. This is visually represented by the vertical segment of the "API Exposure Management" rectangle in the blueprint (see Figure 4-3).
2. It acts as a single-entry point for external communications and interactions with the WP6 enablers, depicted by the horizontal segment of the "API Exposure Management" block in the blueprint (see Figure 4-3).

To clarify the mapping and relationships within the framework, as depicted in Figure 4 4, the Management Capabilities Exposure Framework is directly connected to the Intent DSM, represented by the pink block labelled “Intent-based Digital Service Manager” in Figure 4 3. Furthermore, the placement of the horizontal segment of this block, positioned above the Service Layer M&O (shown as a light green upper block in Figure 4 3) indicates the specific APIs that the Management Capabilities Exposure Framework manages, namely the Service APIs.

This arrangement underscores the enabler’s role in interfacing with crucial components within the E2E system architecture. It facilitates streamlined communication and management across different layers, ensuring that external communications and system functionalities are efficiently handled through a centralized entry point provided by the Integration Fabric. This configuration not only enhances connectivity but also ensures that all components within the administrative domain are cohesively integrated, promoting robust interaction within and with the M&O functionalities defined by WP6 contribution.

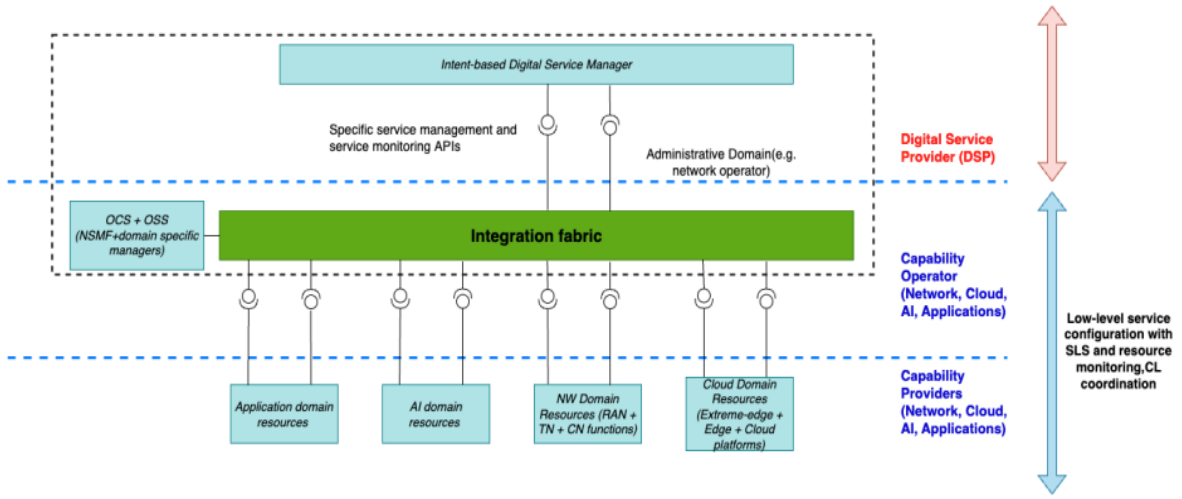


Figure 4-4: Integration Fabric – high-level architecture.

Finally, it is important to note that the Management Capabilities Exposure Framework is restricted to a single administrative domain. This means there will be one Integration Fabric instance per Mobile Network Operator (MNO). Based on this structure, it is anticipated that the management of extreme-edge resources will be handled by resource orchestrators within the cloud continuum, specifically enabler 5. Devices outside this domain, particularly those at the extreme edge (shown as the Device blue block in the bottom left of Figure 4-3), will be beyond the reach of Management Capabilities Exposure Framework. The fabric's role, in this specific scenario, is focused solely on ensuring that the enabler 5 orchestrator is accessible to other system components within WP6.

4.4 Enabler 4: Security and trustworthiness framework

This enabler is not considered well aligned with the proposed blueprint, requiring the removal of one functional block (the "3rd party trust management block) and the addition of two new functional blocks (the "3rd party service provisioning" and the "Trust Management" blocks, highlighted in Figure 4-5).

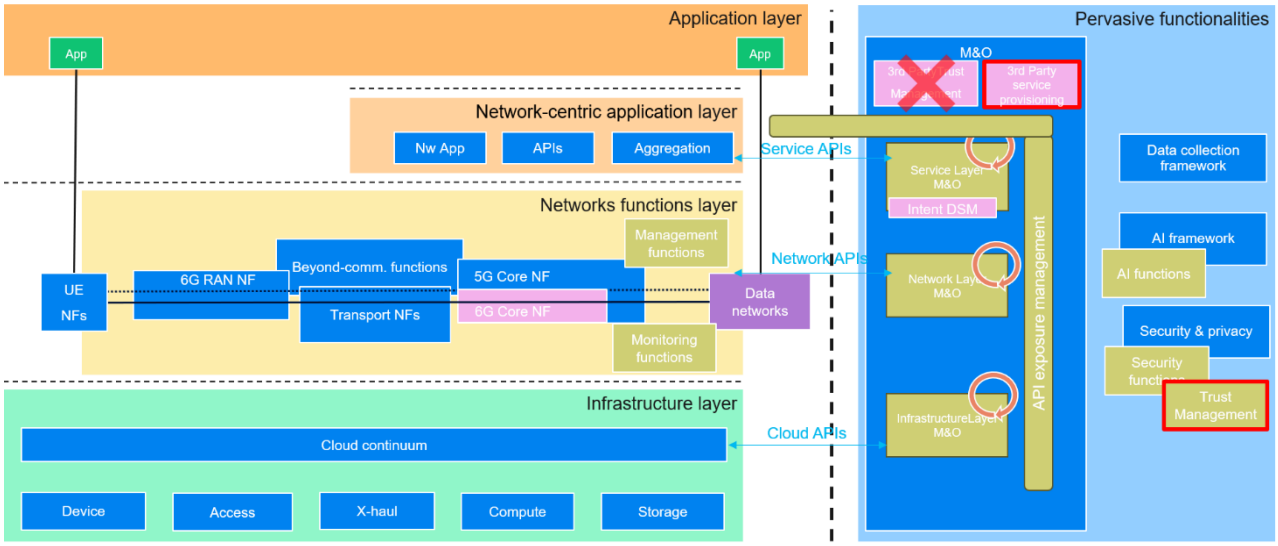


Figure 4-5: Overall alignment between sub-enabler 4.3 and the E2E system blueprint.

The "3rd Party service provisioning" block would cover the functionalities implemented by sub-enablers 4.1 (Third-party resource control separation enabler) and 4.2 (user-centric service provisioning system). The specific update request would be to replace "3rd party trust management" block in the blueprint with these new blocks.

On the other hand, the proposed new "Trust Management" block would implement the sub-enabler 4.3 functionality (i.e., the Trust management system). It is considered that this block should be understood as "pervasive", and alongside with the "Security functions" block. This enabler would be associated with the data collection probes intended to monitor the three M&O blocks (infrastructure, network, service), being its outputs (consisting of trustworthiness indexes) going to all these M&O layers.

4.5 Enabler 5: Synergetic orchestration mechanisms for the computing continuum

The applied M&O mechanisms support the lifecycle management for the deployment and operation of services, whether they may regard pure network services (e.g., VNF forwarding graphs composing a network service) or application-oriented services with specific requirements that have to be satisfied by the networking infrastructure. The latter refer to the applications provided through the "Network-centric application layer" and the "Application layer". Depending on the type of the service, different stakeholders and platforms may be involved to support the M&O functionalities. For instance, in case of a pure network service, M&O is provided by NFV Orchestrators managed by Communication Service Providers (CSPs) that reside at the "Network functions layer" of the Blueprint. In the case of application-oriented services, M&O is provided by frameworks that are compatible with edge/cloud computing orchestration technologies, where a set of open APIs can be consumed for interaction with the "Network functions layer".
Different techniques can be supported for multi-agent synergetic orchestration mechanisms that mostly follow a hierarchical approach, decentralized orchestration mechanisms that build upon decentralized intelligence, and federated orchestration mechanisms that consider the interaction between multiple orchestrators (e.g., by considering operation under different administrative domains). Peculiarities based on the need to manage resources that span across the computing continuum (IoT devices, extreme edge devices, edge/cloud infrastructure) are considered. This enabler is composed of 3 sub-enablers, considering the aforementioned techniques. Following is the analysis of the alignment for each of them.

4.5.1 Sub-enabler 5.1: Multi-agent systems for multi-cluster orchestration

This sub-enabler is considered well aligned with the proposed architecture, being implemented as a set of mechanisms within the M&O block in the Pervasive functionalities layer (Figure 4-6).

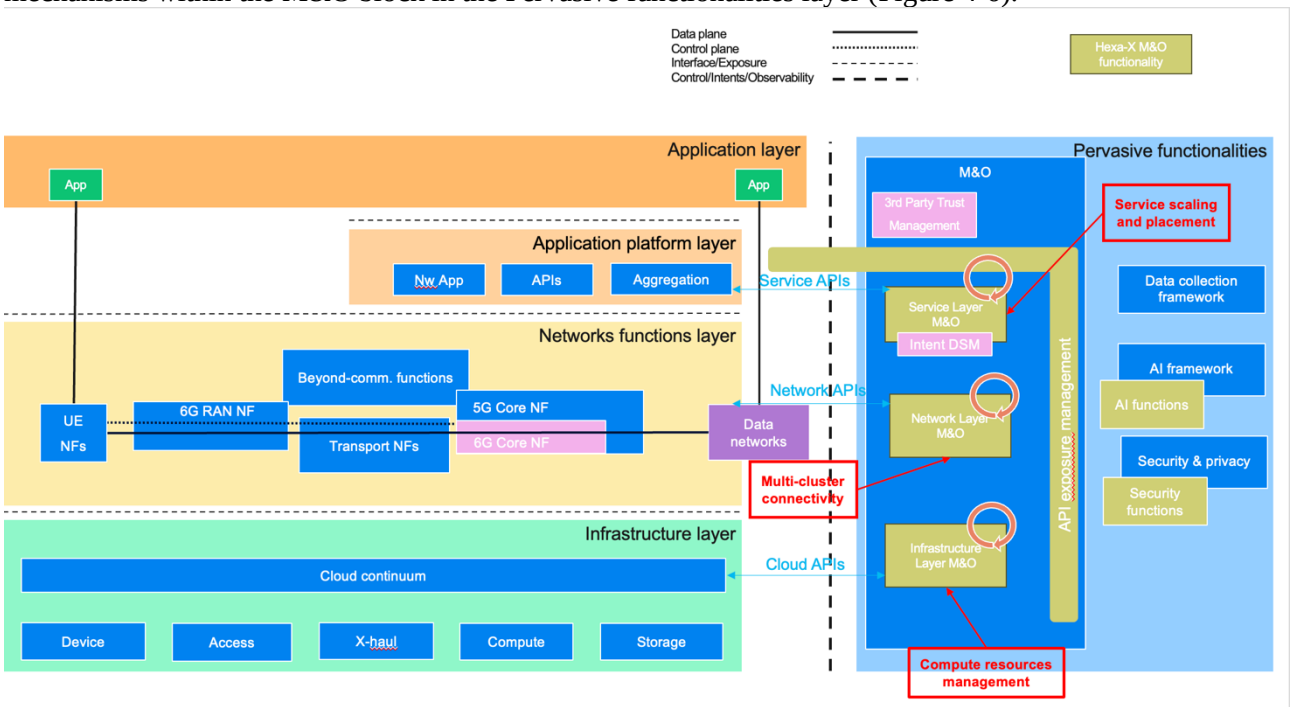


Figure 4-6. Overall alignment between Sub-enabler 5.1 and the E2E system blueprint.

In the context of this sub-enabler, we consider the existence of multiple agents that collaborate to provide orchestration of services across a multi-cluster infrastructure. A multi-layer hierarchical approach is considered, where higher level entities have the responsibility for the management of the service, however the control is distributed across multiple agents in the infrastructure. Collaboration schemes among agents by taking advantage of AI techniques are consider to inject automation and intelligence in the orchestration mechanisms. Multi-cluster connectivity aspects are examined to support connectivity and satisfy QoS requirements of the deployed services. To support the collaboration among the multiple agents, open APIs have to be specified for the interaction between the "Infrastructure Layer", the "Network Layer", the "Network-centric Application Layer" and the "Application layer" to enable management of compute and network resources in multi-cluster environments and synergies among the different stakeholders of a 6G ecosystem. For instance, such an API can provide an offering for the lifecycle management of a network slice over infrastructure in the computing continuum, aiming to serve the QoS needs for a specific vertical industry. It is thought that this should be considered in the design of the final architecture.

4.5.2 Sub-enabler 5.2: Decentralised Orchestration System

This sub-enabler is considered not to be supported by the E2E system blueprint provided from WP2. The main reason for the misalignment is that the blueprint represents an MNO-centric architecture, i.e., an architectural design focused on a single stakeholder (the MNO) and the functional blocks that would be enabled within its own domain. However, the Decentralised Orchestration System approach proposes basically the opposite, i.e., a decentralized orchestration architecture in which, although the MNO is still a relevant actor, it is seen as another stakeholder part of the network continuum, operating together with the other actors envisaged in the future 6G system. I.e., the proposed blueprint seems to assume that the M&O functionality can just fit into a single functional block within the MNO own domain (the M&O blue block represented in the blueprint diagram), assuming a centralised approach in what regards that M&O functionality, which is not aligned with the Decentralised Orchestration System approach.

Based on the previous, it is recommended to update the blueprint in the next design iterations to support the features provided with this enabler. Below the list of the most significant suggested changes:

a) The scope of the architectural design should be explicitly defined, including the whole network continuum with its inherent multiplicity of stakeholders. It is considered that the way the network continuum has been included in the blueprint (the long blue rectangle on top of the Infrastructure Layer) could be misleading: on the one hand it seems to express that the network continuum is "within" the MNO scope, which is not the case (the network continuum expands beyond the MNO own boundaries, including the extreme-edge domain); on the another hand, it should be explicitly highlighted that the MNO itself is part of such network continuum, i.e., part of a diverse ecosystem with multiple stakeholders and network domains, on which network services can be deployed and orchestrated in a decentralized way. This approach would make the architectural design multi-domain by design[7].

b) It should be highlighted also that the architectural design is based on the SBA approach, with all its main implications, e.g., that it is micro-services-based design, that it is based also on the usage of exposed interfaces to communicate those micro-services, that relies on DevOps mechanisms (with its associated functional blocks) for the network services development and operation, and that the deployment of network services may include distributed components on a widespread global scale.

c) The blueprint should communicate also that virtualized (mostly containerized according to the previous) NFs could be used for both: implementing common network devices (e.g., routers, switches, firewalls…) but also service logic components as well.

[7] Even if the Distributed Orchestration approach were not adopted, the scope of the architectural design should be explicitly defined in the blueprint, e.g., highligting that the architectual design would apply to a single MNO. As it is represented in its current version, the blueprint seems to take it for granted that the design applies to a specific MNO. If that is the case, this should be explicitly expressed in the architectural design.

d) Derived from (a), (b) and (c), the architectural design should illustrate that micro-service chaining could expand through the multiple network continuum domains, by relying on the service components exposed interfaces.
e) The blueprint should also represent that, instead of a single monolithic MNO-centric M&O block, there could be a diversity of distributed M&O systems (as described in Section 3.5.2) for implementing both: services provisioning and assurance. In any case, the MNO-centric M&O block would be one of these distributed systems.

4.5.3 Sub-enabler 5.3: Federated Orchestration System

This sub-enabler is also not considered to be aligned with the blueprint provided from WP2. The main reason is that for implementing this sub-enabler it should be necessary to consider the various stakeholders in a 6G ecosystem and their orchestration needs, in accordance with the defined layers (network functions layer, network-centric application layer, application layer).

Speaking about federated service orchestration, the orchestration platforms being used differ from legacy single provider systems. In parallel, separation of concerns has to be considered regarding what kind of services can be managed per stakeholder, while the interaction among stakeholders has to be based on (open) APIs. These APIs are currently not yet standardized. What is proposed from this WP6 are custom East/West bound APIs that connect to an overlay blockchain network that handles the smart contracts used to establish dynamic SLAs and service provisioning among participating providers.

Figure 4-7 shows how the blueprint should be updated to be aligned with the functionality provided by this enabler, in terms of new interfaces and new components, respectively.

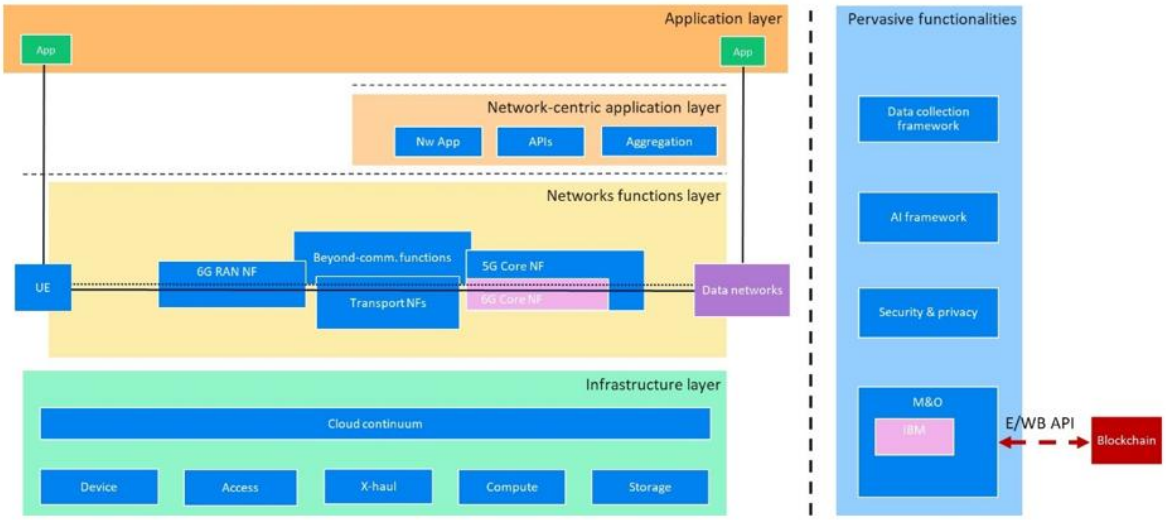


Figure 4-7: Overall alignment between Sub-enabler 5.3 and the E2E system blueprint – new interfaces.

4.6 Enabler 6: AI/ML Algorithms

The enabler consists of two sub-enablers, namely: sub-enabler 6.1 (AI/ML-based control algorithms for sustainability) and sub-enabler 6.2 (Trustworthy AI/ML-based control algorithms), both considered well aligned with the proposed blueprint. The alignment for each of them is described in the following section. The AI Framework block in these figures represents a set of pervasive functionalities that facilitate the training and deployment of AI/ML functions in the M&O framework. These AI/ML functions can be deployed in a distributed or centralized manner, or as part of an M&O function (i.e. orchestrator) depending on the specific M&O process they are part of.

4.6.1 Sub-enabler 6.1: AI/ML-based control algorithms for sustainability

Figure 4-8 illustrates the envisaged alignment between this sub-enabler and the blueprint. As it can be observed, the E2E system blueprint includes the AI framework which can be employed to train models for

intelligent M&O with sustainability or other cost related targets. The trained models are provided to the M&O layer for intelligent control of the different architecture layers. Additionally, the data collection framework can be used to collect sustainability and energy-efficiency related data to provide as input to the models. Thus, no modifications of the blueprint are necessary for supporting this sub-enabler. However, in any description, there need to be at least aspects of sustainability and cost-efficiency as a target for the intelligence and the training of AI/ML models in the M&O.

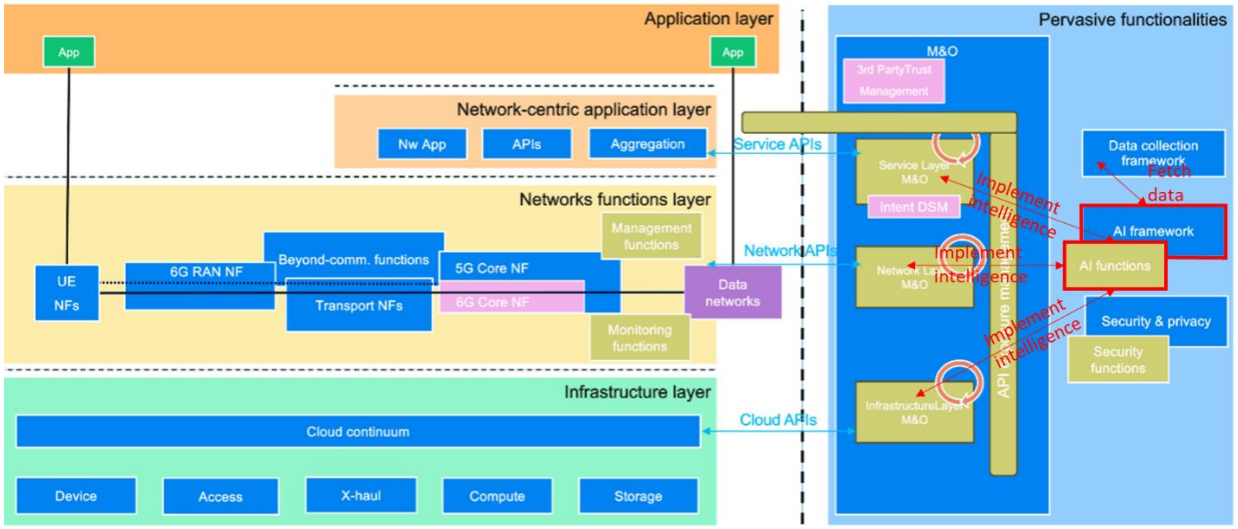


Figure 4-8. Overall alignment between Sub-enabler 6.1 and the E2E system blueprint.

4.6.2 Sub-enabler 6.2: Trustworthy AI/ML-based control algorithms

Figure 4-9 illustrates the envisaged alignment between this sub-enabler and the blueprint. As it can be appreciated the AI functions, and the Security functions (which are understood as part of the AI Framework and the Security & Privacy blocks in the blueprint) would host this Trustworthy AI/ML enabler, which is represented by the red block labelled as “Trustworthy AI”.

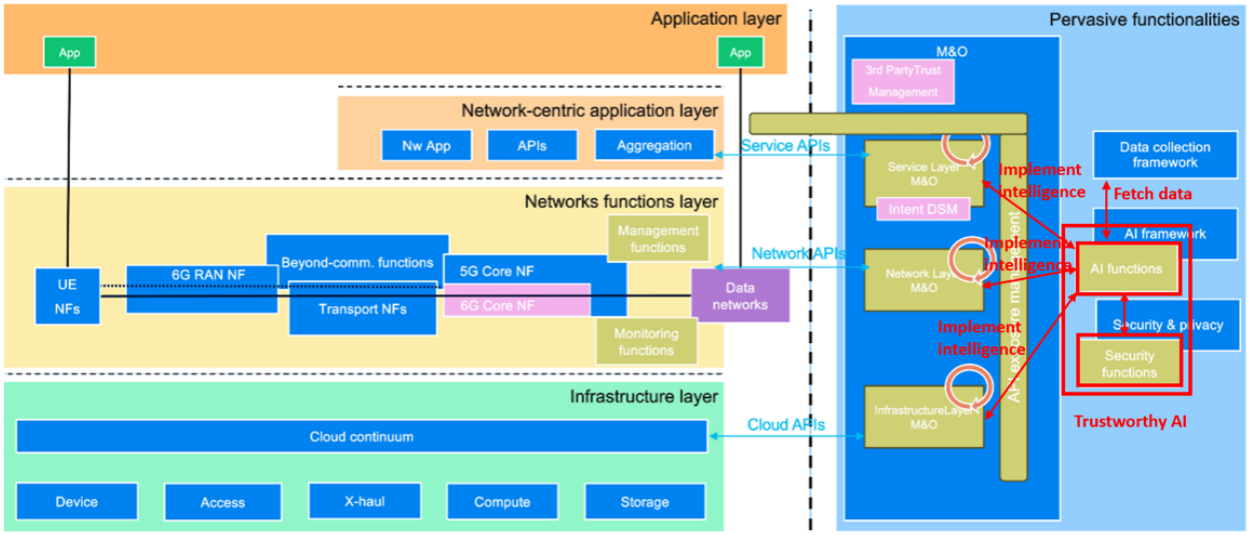


Figure 4-9. Overall alignment between Sub-enabler 6.2 and the E2E system blueprint.

Beside the functional blocks themselves, there are also envisaged interfaces between the different M&O layers and the “AI functions” block, to perform AI operations in the service layer, network layer and infrastructure layer M&O blocks. In addition, there would be a dedicated interface between the “AI function” block and the security function to improve security, privacy, and explainability of the AI/ML models, and to provide a robust and privacy-preserving (i.e., trustworthy) AI function. The AI functions would also need to implement interfaces with the Data collection framework, to fetch data to be used for model training and model prediction.

4.7 Enabler 7: Network digital twins creation mechanisms

This enabler is considered to not be supported by the E2E system blueprint currently provided from WP2. The main reason is that the blueprint, as presented, does not suggest that performance models (of any type, abstract or NDTs) are used in the M&O process. A suggestion to include this enabler in the blueprint is represented in Figure 4-10.

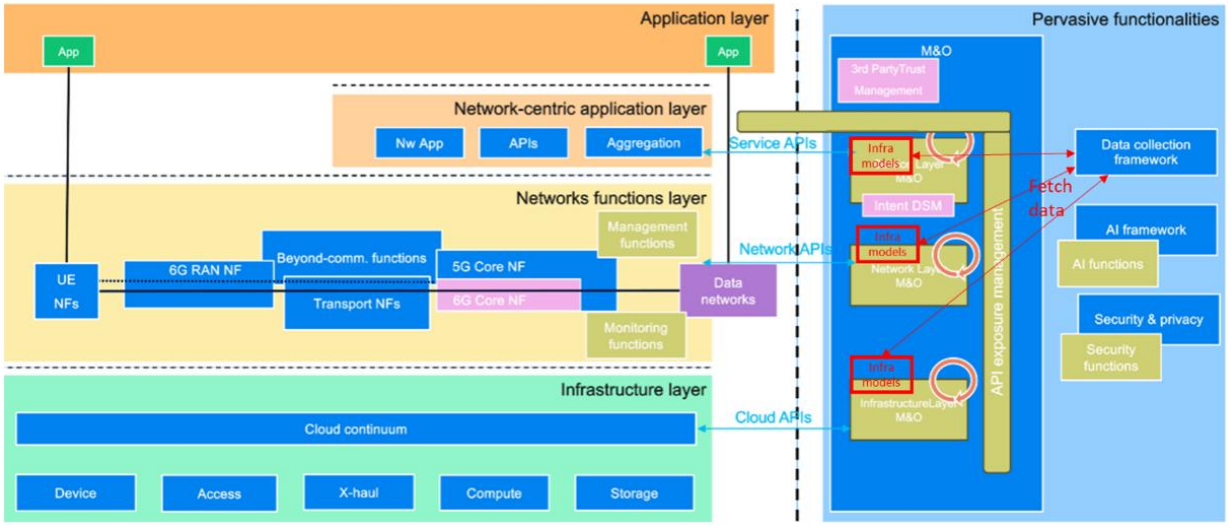


Figure 4-10. Suggested approach for aligning Enabler 7 with the E2E system blueprint.

In the Hexa-X-II M&O framework, the Network Digital Twins play an intermediary role between the M&O and the network infrastructure layers that are configured to provision a service. They are the models used by different M&O processes, generally in closed management loops, giving those processes an accurate virtual replica of the actual artifacts they represent. In order to incorporate these functionalities, the blueprint will need to include open APIs connecting the Digital Twin to the M&O block and the infrastructure and network functions layers as shown in Figure 4-10.

4.8 Enabler 8: Real-time zero-touch control loops automation and coordination system.

This enabler is considered not to be fully in line with the E2E system blueprint defined in WP2 for two main reasons:

- The CL coordination functionality is not yet explicitly indicated in the blueprint;
- The CL governance functionality can be implicitly considered as part of the Management Functions. However, the multi-domain nature of the CL instances is not entirely reflected in the picture, as it is instead for their multi-layer characteristic.

The following Figure 4-11 shows how the different abstractions and functionalities associated to this enabler would be mapped into the E2E system blueprint. The white boxes with red lines indicate how the current E2E blueprint components and APIs can be mapped in the CL automation and coordination system elements, while the red box indicates components which should be added.

The circles at the different layers of the M&O box (service, network and infrastructure layers) well represent the various CL instances that can be deployed in each layer with their own scope and objective. It should be considered that each CL includes a number of CL functions: Monitoring, Analysis, Decision, Execution and Knowledge. Their provisioning and orchestration are handled through the so-called CL Governance function, while the coordination of concurrent CLs is management through the CL Coordination Function (see section 3.8.1).

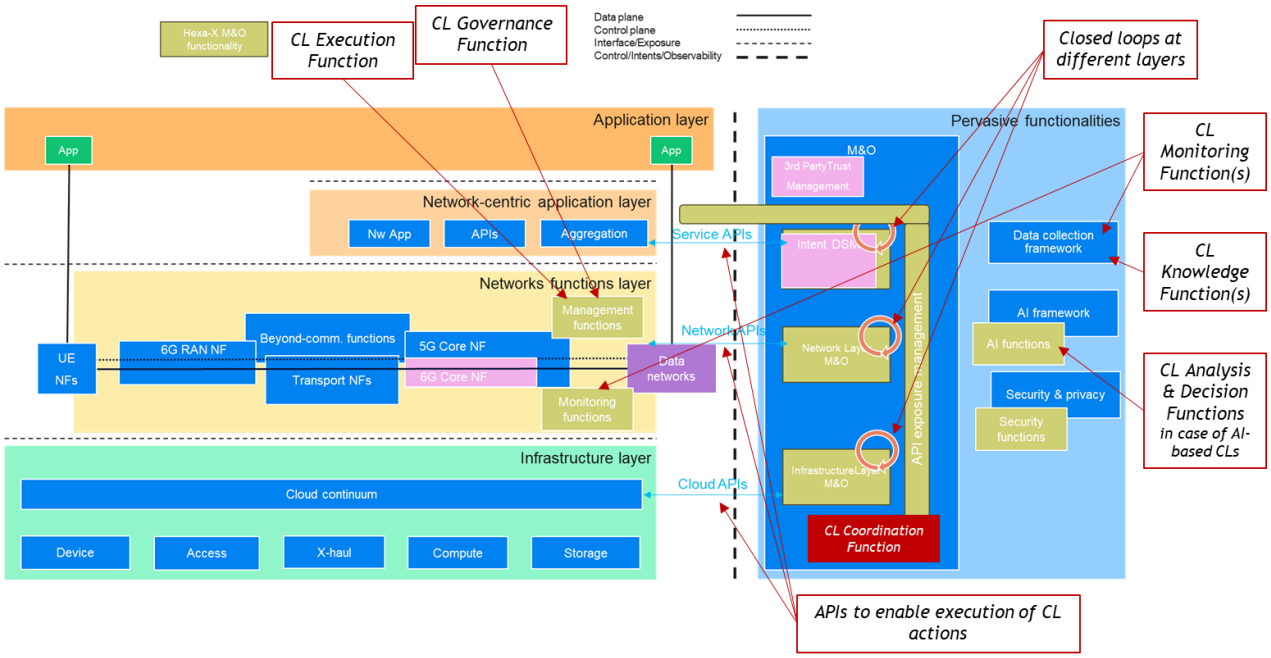


Figure 4-11: Suggested approach for aligning Enabler 8 with the E2E system blueprint.

As shown in Figure 4-11 it is possible to identify a clear mapping between the internal CL functions and the components already represented in the E2E blueprint, as follows:

- The **CL Monitoring** function collects the data required for the system automation from the system itself (e.g., through the NWDAF or the MDAF, or their equivalent in the future 6G stack) or from external sources. As such, it could be mapped to the "Monitoring functions" at the Network functions layer or the "Data collection framework" in the Pervasive functionalities, depending on the type of data to be monitored.
- The **CL Knowledge** function collects and stores the data to be shared among the CL functions and it can be mapped to the "Data collection framework".
- The **CL Analysis and CL Decision** functions, in case of cognitive closed loops, adopt AI/ML techniques and they can be mapped to the "AI functions". Their deployment could be mediated through the "AI framework".
- The **CL Execution** function depends on the particular type and scope of the CL. For example, it could be an SDN controller module if the CL is applied to the transport network, or a Network Slice Management Function (NSMF) if the CL is applied to a network slice, or a resource controller if the CL operates at the infrastructure layer. As such, the CL Execution function could be mapped to a generalized "Management function" represented at the Networks functions layer.

It is also worth to note that, depending on the CL scope, the decisions taken by the CL must be enforced through the CL Execution function at different layers and this requires suitable API to operate e.g., resources in the continuum, RAN or Core network functions, etc. These APIs are reflected in the E2E blueprint through the Cloud APIs, Network APIs, and Service APIs.

The **CL Governance** function is in charge of handling the provisioning, orchestration and life-cycle management of the CL services, e.g., deploying and interconnecting the CL functions, deciding their placement and dimension in the virtualized continuum environment, managing their scaling, migration, configuration, etc. As such, it can be considered a specific M&O "Management function".

The **CL Coordination** function, instead, provides the logic to coordinate groups of closed loops working in parallel in different layers or domains, but characterized by a given level of interdependency. For example, the CL coordination function may detect and mitigate CL conflicts, give an order and a timeline to execution actions by different CLs, manage delegation and escalation workflows between hierarchical CLs, etc. This function is not yet included in the E2E blueprint and it would need to be added as a new component of the pervasive M&O functionalities (see red box in Figure 4-11). The interaction between CL Coordination and CL Governance functions may be mediated through the Management capabilities exposure framework enabler.

5 Contributions to dissemination activities

This section reports the contributions to papers (Table 5-1) and demonstrations (Table 5-2) derived from the M&O concepts developed in Hexa-X-II.

Paper	Partners	Enablers
R. Pires, et al., “Closed-Loop Automation in 6G for Minimum Downtime Task Continuity in Surveillance Cobots”, EuCNC and 6G Summit, June 2024 [PBM+24]	VTT, NXW	#2, #5.1, #8
P. Alemany, et al., “Multi-Stakeholder Intent-based Service Management Automation for 6G Networks”, EuCNC and 6G Summit, June 2024 [AMO+24]	CTT, TID, NFR, VTT, NFI, EAB, EBY, ICC, NXW, OPT	#1, #3, #4.2, #8
J. Miserez, et al., An east-westbound control architecture for multi-segment deterministic networking, 2024 IFIP Networking Conference (IFIP Networking). IEEE, June 2024 [MCP+24]	IMEC	#1
R. Vilalta, et al., “Applying Digital Twins to Optical Networks with Cloud-native SDN Controllers”, IEEE ComMag, 2023. [VGX+23]	CTT, TID	#1, #2
R. Vilalta, et al., Providing Anomalous Behaviour Profiling by extending SmartNIC Transceiver support in Packet-Optical Networks [VVG+24]	CTT, TID, ATO	#1
R. Vilalta, et al., Demonstration of Intent-Based Networking for End-to-End Packet Optical Cloud-native SDN Orchestration, ECOC 2023 [VAM+23]	CTT	#1
C. Manso, et al., Introducing End-to-End Location Awareness in Packet-Optical Networks [MVG+23]	CTT	#1
M. Karaca, et al, “Utilizing Causal Learning for Cognitive Management of 6G Networks”, IEEE International Conference on Machine Learning for Communication Networks (ICMLCN), Sweden, May 2024.	EBY	#8
S. Kerboeuf, et al., "Design Methodology for 6G End-to-End System: Hexa-X-II Perspective," in IEEE Open Journal of the Communications Society, 2024, [KPJ+24]	All	All

Table 5-1: Contributions to papers.

Demonstration	Partners	Enablers
Decentralised orchestration in PoC#B.	ATO, ASA	#5.2
Closed-loop for automated service migration among cobots in PoC#B.	NXW, VTT, CTT	#2, #5.1, #8
M&O enablers with cobots on warehouse inventory management PoC#A/B	WIN	#4.3, #5.1, #6.1
TeraFlowSDN controller and data plane in-a-box	CTT	#1
Service autoscaling	ICCS	#5.1

ETSI MEC PoC 14: Network resource allocation for Application specific requests using MEC BandWidth Management service and TeraFlowSDN	CTT, TID	#1

Table 5-2: Contributions to demonstrations.

6 Conclusions

This deliverable provides a fundamental contribution towards the design of the Hexa-X-II E2E 6G System Blueprint, presenting the initial design of the 6G smart network management framework identifying a number of technological enablers and sub-enablers. The technical approach presented in the deliverable for each enabler, through the specification of enablers' system components and workflows, as well as proposals for internal execution logic, algorithms and applicable technologies, is intended to feed the design of the Hexa-X-II blueprint in its 2nd iteration for the aspects related to M&O. In this direction, a specific section has been dedicated the analysis of the proposed M&O enablers with respect to an intermediate version of this blueprint. This has identified some misalignments to be solved in the next iteration and given suggestions for the blueprint evolution in WP2, not only in terms of new components, procedures and interfaces, but also for what regards the representation of global concepts, e.g., related to resource continuum or multi-domain aspects.

The analysis of the impact on KPIs and KVIs for future 6G network at the enabler level is expected to drive the prioritization of the M&O enablers for their progressive introduction in the E2E blueprint under definition in WP2. In parallel, it will feed the work an KPIs and KVIs specification in WP1, providing a perspective oriented to the M&O aspects of the future 6G network. Here the focus has been on programmability, monitoring and automation, synergetic and distributed orchestration, scalability, and controlled but powerful and simplified exposure of network capabilities towards external actors. It is clear the deep attention to the sustainability of the future network architecture and the relevant contribution that the M&O system can provide in this direction. In this context, the work has been driven from the beginning with a critical analysis of opportunities and challenges related to the introduction of the most promising technologies for network management, like zero-touch network automation, user-centric and intent-based service management, AI/ML techniques and NDTs, towards a fully sustainable system. This has highlighted the need to design solutions that natively consider crucial aspects like trustworthiness, security, and energy efficiency following an integrated approach. This needs to spans architectural principles, dedicated system components, protocols, internal techniques and algorithms designed specifically to address energy efficiency and resource constraints. An important role here is expected to be covered by the multi-purpose security framework, able to guarantee secure and controlled interactions with third parties, but also extended to evaluate the level of trustworthiness of system elements, services and applications. In parallel, the algorithms at the core of the M&O system components (e.g., for resource allocation, function placement, or CL decisions for network automation) must be defined considering energy efficiency or security not only as their objectives, but also as key drivers for their own design and deployment. In this direction, the document has proposed AI/ML solutions based on secure, privacy-preserving or explainable techniques, as well as FL and sustainable MLOps, to address the two key issues of trustworthiness and energy efficiency in large-scale applicability and adoption of AI/ML for mobile network management.

Finally, it is worth to mention that preliminary implementation and initial validation results have been provided for most of the enablers. Some of them are directly feeding the project PoCs, at different integration stages. This provides a concrete contribution to the small-scale, experimental validation of the M&O concepts and supports the demonstration of project outcomes in conferences and public events. This contribution is also particularly relevant to provide realistic examples of how the system components developed for the M&O framework could be integrated together or with other elements of the E2E blueprint (developed in other WPs). Moreover, the use-case oriented approach allows to identify particular customizations or possible challenges that can feed WP1 work towards the consolidation and refinement of the Hexa-X-II use cases. Other M&O enablers have been implemented through simulations, to preliminary evaluate performance and potential benefits of various implementation alternatives in wider scenarios. However, at this stage, the development work is still in continuous progress. The collaboration with other WPs, especially in the context of the project PoCs, may lead to the introduction of new features, as well as the update of interfaces and workflows in the system components defined so far for M&O enablers. Their final version will be documented in the following D6.4 and D6.5.

7 References

[23.501] 3GPP TS 23.501, “System architecture for the 5G System (5GS)”, v18.5.0, March 2024.

[23.503] 3GPP TS 23.503, “Policy and charging control framework for the 5G System (5GS); Stage 2”, v18.5.0, March 2024.

[24.526] 3GPP TS 24.526, “User Equipment (UE) policies for 5G System (5GS); Stage 3”, v18.6.0, April 2024.

[28.533] 3GPP TS 28.533, “Management and orchestration; Architecture framework”, v18.1.0, April 2024.

[28.536] 3GPP TS 28.536, “Management and orchestration; Management Services for communication service assurance; Stage 2 and stage 3”, v18.0.0, March 2024.

[28.541] 3GPP TS 28.541, “Management and orchestration; 5G Network Resource Model (NRM); Stage 2 and stage 3”, v18.7.0, April 2024.

[28.622] 3GPP TS 28.622, “Telecommunication management; Generic Network Resource Model (NRM) Integration Reference Point (IRP); Information Service (IS)”, v18.6.0, April 2024.

[5GP21] 5GPPP, “Architecture Working Group. View on 5G Architecture”, Version 4.0, August 2021.

[ADB+17] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," in IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26-38, Nov. 2017, doi: 10.1109/MSP.2017.2743240.

[AIR24] Apache AirFlow, available at: https://airflow.apache.org/

[AMK+18] J. Aslan, K. Mayers, J.G. Koomey and C. France, “Electricity intensity of internet data transmission: Untangling the estimates”. 2018. Journal of industrial ecology, 22(4), 785-798.

[AMO+24] P. Alemany, R. Muñoz, J. Ordoñez-Lucena, R. Vilalta, D. Lopez, M. Boussard, H.Q. Tran, P. Porambage, R. Pires, H. Blue, J. Malinen, M. Uusitalo, M. Elbamby, M. Abdelaziz, P. Rugeland, J. Castaneda Cisneros, F. Brito, E. Dehghan Biyar, M. Karaka, A. Zafeiropoulos, I. Tzanettis, P.G. Giardina, G. Landi, X.R. Sousa, and S. Kerboueuf, “Multi-Stakeholder Intent-based Service Management Automation for 6G Networks”, EuCNC and 6G Summit, June 2024.

[BJD+14] R. Bjornson, E. A. Jorswieck, M. Debbah and B. Ottersten, "Multiobjective signal processing optimization: The way to balance conflicting," IEEE Signal Processing Magazine, pp. 14-23, 2014.

[BJZ+22] A. C. Baktir, A. D. N. Junior, A. Zahemszky, A. Likhyani, D. A. Temesgene, D. Roeland, E. D. Biyar, R. F. Ustok, M. Orli´c, and M. D’Angelo, “Intent-based cognitive closed-loop management with built-in conflict handling,” in 2022 IEEE 8th International Conference on Network Softwarization (NetSoft), pp. 73–78, 2022.

[BMC20] A. Banerjee, S. S. Mwanje and G. Carle, "Game theoretic conflict resolution mechanism for cognitive autonomous networks" in International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), 2020.

[BMC21] A. Banerjee, S. S. Mwanje and G. Carle, "Optimal configuration determination in cognitive autonomous networks," in 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), 2021.

[BSL18] E. Bradford, A. Schweidtmann and A. Lapkin, "Efficient multiobjective optimization employing gaussian processes, spectral sampling," Journal of Global Optimization, vol. 71, no. 407-438, 2018.

[CAM24] CAMARA project web page, available at: https://camaraproject.org/ (Accessed April 2024).

[CDT18] T. Cerny, M. J. Donahoo, and M. Trnka, “Contextual understanding of microservice architecture: current and future directions,” ACM SIGAPP Applied Computing Review, vol. 17, no. 4, pp. 29-45, 2018.

[CKS+21] I.P. Chochliouros, M.A. Kourtis, A.S. Spiliopoulou, P. Lazaridis, Z. Zaharis, C. Zarakovitis, and A. Kourtis, “Energy Efficiency Concerns and Trends in Future 5G Network Infrastructures”. Energies 2021, 14, 5392. https://doi.org/10.3390/en14175392.

[CON24] Containerlab web page, available at: https://containerlab.dev/ (Accessed April 2024)

[CVM+19] Casellas R, Vilalta R, Martínez R, Muñoz R. SDN control of disaggregated optical networks with OpenConfig and OpenROADM. InInternational IFIP Conference on Optical Network Design and Modeling 2019 May 13 (pp. 452-464). Cham: Springer International Publishing.

[DOC24] Docker. Available at: https://www.docker.com/

[ETH] go-ethereum documentation. Available at: https://geth.ethereum.org/docs. (Accessed : 2024, April 29).[Eve76] G. Everest, "Basic Data Structure Models Explained with a Common Example", in Computing Systems 1976, Proceedings Fifth Texas Conference on Computing Systems, Austin, TX, 1976 October 18–19, pages 39-46. (Long Beach, CA: IEEE Computer Society Publications Office).

[FAF+16] J. Foerster, I.A. Assael, N. de Freitas and S. Whiteson “Learning to Communicate with Deep Multi-Agent Reinforcement Learning”, Advances in Neural Information Processing Systems 29, 2016

[FKK+23] Famelis P, Katsikas GP, Katopodis V, Natalino C, Renom LG, Martinez R, Vilalta R, Klonidis D, Monti P, King D, Farrel A. P5: Event-driven Policy Framework for P4-based Traffic Engineering. In2023 IEEE 24th International Conference on High Performance Switching and Routing (HPSR) 2023 Jun 5 (pp. 1-3). IEEE.

[FLI24] Apache Flink. Available at: https://flink.apache.org/

[FPS+23] M. Ferriol-Galmés, J. Paillisse, J. Suarez-Varela, K. Rusek, S. Xiao, X. Shi, X. Cheng, P. Barlet-Ros, and A. Cabello, "RouteNet-Fermi: Network Modeling With Graph Neural Networks," in IEEE/ACM Transactions on Networking, vol. 31, no. 6, pp. 3080-3095, Dec. 2023, doi: 10.1109/TNET.2023.3269983

[FSP+20] F. Faticanti, M. Savi, F. D. Pellegrini, P. Kochovski, V. Stankovski and D. Siracusa, "Deployment of Application Microservices in Multi-Domain Federated Fog Environments," 2020 International Conference on Omni-layer Intelligent Systems (COINS), Barcelona, Spain, 2020, pp. 1-6, doi: 10.1109/COINS49042.2020.9191379.

[GKV+19] T. Goethals, S. Kerkhove, L. Van Hoye, M. Sebrechts, F. De Turck and B. Volckaert, “FUSE : a microservice approach to cross-domain federation using docker containers.” In V. M. Munoz, D. Ferguson, M. Helfert, & C. Pahl (Eds.), Closer: Proceedings of the 9th International Conference on Cloud Computing and Services Science, pp. 90–99, 2019. https://doi.org/10.5220/0007706000900099

[GLZ18] Gill, Manmeet Singh, Dale Lindskog, and Pavol Zavarsky. "Profiling network traffic behavior for the purpose of anomaly-based intrusion detection." 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE). IEEE, 2018.

[GNG+22] Gifre L, Natalino C, Gonzalez-Diaz S, Soldatos F, Barguil S, Aslanoglou C, Moreno-Muro FJ, Cornelio AQ, Cepeda L, Martínez R, Manso C. Demonstration of zero-touch device and l3-vpn service management using the teraflow cloud-native sdn controller. InOptical Fiber Communication Conference 2022 Mar 6 (pp. M3Z-15). Optica Publishing Group.

[GNMIC24] GNMI collector, available at: https://github.com/openconfig/gnmic (Accessed April 2024).

[GRA24] Grafana web page, available at: https://grafana.com/ (Accessed April 2024).

[HCD22] A. Heuillet, F. Couthouis, and N. Diaz-Rodriguez, “Collective eXplainable AI: Explaining Cooperative Strategies and Agent Contribution in Multiagent Reinforcement Learning With Shapley Values,” *IEEE Comput. Intell. Mag.*, vol. 17, no. 1, pp. 59–71, Feb. 2022, doi: 10.1109/MCI.2021.3129959.

[HEL24] HELM web page, https://helm.sh/ (Accessed May 2024).

[HEX24] Hexa-X project, https://hexa-x.eu/ (Accessed April 2024).

[HEX21-D71] Hexa-X, “Deliverable D7.1 Gap analysis and technical work plan for special purpose functionality”, June 2021.

[HEX22-D13] Hexa-X, “Deliverable D1.3 Targets and requirements for 6G – initial E2E architecture”, March 2022.

[HEX22-D62] Hexa-X, “Deliverable D6.2 Design of service management and orchestration functionalities”, April 2022.

[HEX23-D63] Hexa-X, “Deliverable D6.3 Final evaluation of service management and orchestration mechanisms”, April 2023.

[HEX23-D14] Hexa-X, “Deliverable D1.4 Hexa-X architecture for B5G/6G networks-final release”, August 2023.

[HEX223-D21] Hexa-X-II, “Deliverable D2.1 Draft foundation for 6G system design”, June 2023.

[HEX223-D22] Hexa-X-II, “Deliverable D2.2 Foundation of overall 6G system design and preliminary evaluation results”, December 2023.

[HEX223-D32] Hexa-X-II, “Deliverable D3.2 Initial Architectural enablers”, October 2023.

[HEX224-D33] Hexa-X-II, “Deliverable D3.3 Initial analysis of architectural enablers and framework”, April 2024.

[HEX223-D52] Hexa-X-II, “Deliverable D5.2 Characteristics and classification of 6G device classes”, October 2023.

[HEX223-D62] Hexa-X-II, “Deliverable D6.2 Foundations on 6G Smart Network Management Enablers”, October 2023.

[INF24] InfluxDB [Online]. Available at: https://www.influxdata.com/ (Accessed: Apr. 2024).

[IS21] K. Indrasiri, and S. Suhothayan, (2021) Design Patterns for Cloud Native Applications (Chapter 3). O'Reilly Media, Inc.

[ITU-X500] ITU-T Recommendation X.500, "Information technology - Open Systems Interconnection - The Directory: Overview of concepts, models and services (10/19)”, October 2019.

[Joh73] D. S. Johnson, “Near-optimal bin packing algorithms (Ph.D. thesis),” Massachusetts Institute of Technology, Cambridge, MA, United States, 1973.

[K3S] K3S Lightweight Kubernetes [Online]. Available at: https://k3s.io (Accessed: 9 Apr. 2024).

[K8S] Kubernetes [Online]. Available at: https://kubernetes.io (Accessed: 9 Apr. 2024).

[KAF24] Apache Kafka [Online]. Available at: https://kafka.apache.org/ (Accessed: Apr. 2024).

[KAFP24] Apache Kafka Protocol [Online]. Available at: https://kafka.apache.org/protocol.html (Accessed: Apr. 2024).

[KLM+22] H. Kokkonen, L. Lovén, N.H. Motlagh, A. Kumar, J. Partala, T. Nguyen, V. Casamayor Pujol, P. Kostakos, T. Leppänen, A. González-Gil, E. Sola, I. Angulo, M. Liyanage, M. Bennis, S. Tarkoma, S. Dustdar, S. Pirttikangas, and J. Riekki, “Autonomy and intelligence in the computing continuum: Challenges, enablers, and future directions for orchestration,” 2022. [Online]. Available: https://arxiv.org/abs/2205.01423.

[KPJ+24] S. Kerboeuf, P. Porambage, A. Jain, P. Rugeland, G. Wikström, M. Ericson, D.T. Bui, A. Outtagarts, H. Karvonen, P. Alemany, R. Muñoz, R. Vilalta, P. Botsinis, A. Ramos, J. Cisneros, M. Karaca, C. Karousatou, S. Barmpounakis, P. Demestichas, A. Zafeiropoulos, I. Tzanettis, S. Papavassiliou, P.G. Giardina, G. Landi, B. Han, A. Nimr, and M.A. Uusitalo, "Design Methodology for 6G End-to-End System: Hexa-X-II Perspective," in IEEE Open Journal of the Communications Society, 2024, doi: 10.1109/OJCOMS.2024.3398504.

[KSM24] kube-state-metrics web page, available at: https://github.com/kubernetes/kube-state-metrics (Accessed April 2024).

[LCA+22] I. Leyva-Pupo, C. Cervelló-Pastor, C. Anagnostopoulos, D. P. Pezaros, "Dynamic UPF placement and chaining reconfiguration in 5G networks", Computer Networks, Volume 215, 9 October 2022.

[LHP+16] T. P. Lillicrap, J.J. Hunt, A. Pritzel et al., “Continuous control with deep reinforcement learning,” in 4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings, Y. Bengio and Y. LeCun, Eds., 2016.

[LXD] LXD documentation. Available at: https://documentation.ubuntu.com/lxd/en/latest. (Accessed : 2024, April 11).

[LVU+16] Lopez V, Vilalta R, Uceda V, Mayoral A, Casellas R, Martinez R, Munoz R, Palacios JP. Transport API: A solution for SDN in carriers networks. InECOC 2016; 42nd European Conference on Optical Communication 2016 Sep 18 (pp. 1-3). VDE.

[MCA+24] M. Karaca, J. Sadasivan, A.C. Baktir, A. Palaios and A. Zahemszky “Utilizing Causal Learning for Cognitive 6G Networks”, IEEE International Conference on Machine Learning for Communication Networks, Sweden, May , 2024.

[MCP+24] J. Miserez, D. Colle, M. Pickavet, and W, Tavernier, “An east-westbound control architecture for multi-segment deterministic networking”, 2024 IFIP Networking Conference (IFIP Networking). IEEE, June 2024.

[MCT22] Mohammadi S, Colle D, Tavernier W. Latency-aware topology discovery in SDN-based time-sensitive networks. In2022 IEEE 8th international conference on network softwarization (NetSoft) 2022 Jun 27 (pp. 145-150). IEEE.

[MNC+17] Muñoz, R., Nadal, L., Casellas, R., Moreolo, M.S., Vilalta, R., Fàbrega, J.M., Martínez, R., Mayoral, A. and Vílchez, F.J., 2017, June. The ADRENALINE testbed: An SDN/NFV packet/optical transport network and edge/core cloud platform for end-to-end 5G and IoT services. In 2017 European Conference on Networks and Communications (EuCNC) (pp. 1-5). IEEE.

[MEC003] ETSI GS MEC 003, “Multi-access Edge Computing (MEC); Framework and Reference Architecture”. March 2023.

[MEC015] ETSI GS MEC 015, “Multi-Access Edge Computing (MEC); Traffic Management APIs”, 2020.

[MECP014] ETSI MEC PoC 014, “Network Resource Allocation”. Available at: https://mecwiki.etsi.org/index.php?title=PoC_14_Network_resource_allocation (Accessed May 2024).

[MG15] J. Moysen and L. Giupponi, "Self coordination among SON functions in LTE heterogeneous networks," in 2015 IEEE 81st Vehicular Technology Conference (VTC Spring), 2015.

[MGG+18] J. Moysen, M. Garcia-Lozano, L. Giupponi and S. Ruiz, "Conflict Resolution in Mobile Networks: A Self-Coordination Framework Based on Non-Dominated Solutions and Machine Learning for Data Analytics," IEEE Computational Intelligence Magazine, pp. 52-64, May 2018.

[MOO24] Moonlight Game Streaming. Available at: https://moonlight-stream.org/ (Accessed May 2024)

[MOR24] NVIDIA Morpheus. Available at: https://github.com/nv-morpheus/Morpheus (Accessed May 2024)

[MQT24] MQTT web page, available at: https://mqtt.org/ (Accessed April 2024)

[MVG+23] C. Manso, et al., Introducing End-to-End Location Awareness in Packet-Optical Networks, ECOC 2023.

[NAG24] Nagios.Available at: https://www.nagios.org/ (Accessed April 2024)

[NET24] Network Service Mesh web page, available at: https://networkservicemesh.io/ (Accessed April 2024)

[NET24a] Network Service Mesh documentation web page, available at: https://networkservicemesh.io/docs/concepts/enterprise_users/ (Accessed April 2024)

[NFL15] [NLF15] C. B. Nielsen, P. G. Larsen, J. Fitzgerald, J. Woodcock, J. Peleska. 2015. Systems of Systems Engineering: Basic Concepts, Model-Based Techniques, and Research Directions.

ACM Comput. Surv. 48, 2, Article 18 (November 2015), 41 pages. https://doi.org/10.1145/2794381

[NFV13] "Network Functions Virtualization (NFV) – Network Operator Perspectives on Industry Progress", SDN and OpenFlow World Congress, 2013.

[NG116] GSMA Generic Network Slice Template, Version 9.0, April 2023 [Online]. Link: https://www.gsma.com/newsroom/wp-content/uploads//NG.116-v9.0-1.pdf

[NG13] Nadeau, Thomas D., and Ken Gray. SDN: Software Defined Networks: An authoritative review of network programmability technologies. " O'Reilly Media, Inc.", 2013.

[NOMAD] Nomad [Online]. Available at: https://www.nomadproject.io (Accessed: 9 Apr. 2024).

[NTO24] Ntopng [Online]. Available at: https://www.ntop.org/products/traffic-analysis/ntop/ (Accessed: 9 Apr. 2024).

[OMN24] OMNET++, available at: https://omnetpp.org/

[OTL24] OpenTelemetry, available at: https://opentelemetry.io/

[PBM+24] R. Pires, H. Blue, J. Malinen, M. De Angelis, P.G. Giardina, G. Landi, M. Laukkanen, K. Aloha, and P. Porambage, "Closed-Loop Automation in 6G for Minimum Downtime Task Continuity in Surveillance Cobots", EuCNC and 6G Summit, June 2024.

[PLD+22] N. Piovesan, D. López-Pérez, A. De Domenico, X. Geng, H. Bao, M. Debbah, "Machine Learning and Analytical Power Consumption Models for 5G Base Stations", IEEE Communications Magazine, 60(10), 56-62, 2022.

[PRE24] PREDICT-6G project, https://predict-6g.eu/ (Accessed April 2024).

[PRO24] Prometheus web page, available at: https://prometheus.io/ (Accessed April 2024).

[PVH+20] E. Puiutta and E. M. S. P. Veith, "Explainable Reinforcement Learning: A Survey," in *Machine Learning and Knowledge Extraction*, vol. 12279, A. Holzinger, P. Kieseberg, A. M. Tjoa, and E. Weippl, Eds., in Lecture Notes in Computer Science, vol. 12279. , Cham: Springer International Publishing, 2020, pp. 77–95. doi: 10.1007/978-3-030-57321-8_5.

[RCV+22] D. Rosendo, A. Costan, P. Valduriez, G. Antoniu, "Distributed intelligence on the Edge-to-Cloud Continuum: A systematic literature review", Journal of Parallel and Distributed Computing, vol. 166, pp. 71-94, 2022, ISSN 0743-7315, https://doi.org/10.1016/j.jpdc.2022.04.004 (https://www.sciencedirect.com/science/article/pii/S0743731522000843).

[RFC6749] IETF RFC 6749: "The OAuth 2.0 Authorization Framework" [Online]. Link: https://datatracker.ietf.org/doc/html/rfc6749

[RFC8341] IETF RFC 8341: "Network Configuration Access Control Model" [Online]. Link: https://datatracker.ietf.org/doc/html/rfc8341

[RFC8466] IETF RFC 8466: "A YANG Data Model for Layer 2 Virtual Private Network (L2VPN) Service Delivery" [Online]. Link: https://datatracker.ietf.org/doc/html/rfc8466

[RFC8795] IETF RFC 8795: "YANG Data Model for Traffic Engineering (TE) Topologies" [Online]. Link: https://datatracker.ietf.org/doc/html/rfc8795

[RFC9543] IETF RFC 9543: "A Framework for Network Slices in Networks Built from IETF Technologies" [Online]. Link: https://datatracker.ietf.org/doc/html/rfc9543

[RMV+20] A. Rodríguez-Molina, E. Mezura-Montes, Villarreal-Cervantes, M. G, Aldape-Pérez and M, "Multi-objective meta-heuristic optimization" Applied Soft Computing, vol. 93, pp. 106342, 2020.

[RSS+18] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster and S. Whiteson, "QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning", International Conference on Machine Learning, PMLR vol. 80, pp. 4295-4304, 2018.

[RTC+20] Steven van Rossem, Wouter Tavernier, Didier Colle, Mario Pickavet, Piet Demeester, "VNF Performance modelling: From stand-alone to chained topologies", Computer Networks Volume 181, 9 November 2020.

[RTI+24] F. Ruggeri, A. Terra, R. Inam, and K. H. Johansson, “Evaluation of Intrinsic Explainable Reinforcement Learning in Remote Electrical Tilt Optimization,” in *Proceedings of Eighth International Congress on Information and Communication Technology*, vol. 696, X.-S. Yang, R. S. Sherratt, N. Dey, and A. Joshi, Eds., in Lecture Notes in Networks and Systems, vol. 696. , Singapore: Springer Nature Singapore, 2024, pp. 835–854. doi: 10.1007/978-981-99-3236-8_67.

[SCA24] Scaphandre web page, available at: https://github.com/hubblo-org/scaphandre (Accessed April 2024).

[SGT+09] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner and G. Monfardini, "The Graph Neural Network Model," in IEEE Transactions on Neural Networks, vol. 20, no. 1, pp. 61-80, Jan. 2009, doi: 10.1109/TNN.2008.2005605.

[SHR20] T. Subramanya, D. Harutyunyan, R. Riggio, "Machine learning-driven service function chain placement and scaling in MEC-enabled 5G networks", Computer Networks, Volume 166, 15 January 2020.

[SLY15] K. Sohn, H. Lee, and X. Yan, “Learning structured output representation using deep conditional generative models.”, Advances in Neural Information Processing Systems, NeurIPS, 28, 2015.

[SPA24] Apache Spark. Available at: https://spark.apache.org/

[S21] Schmager, Stefan. "TRUSTWORTHY AI." (2021).

[STL23] STL Partners, “From Telco to Techno: Siz tenets for success”, August 2023, available at: https://stlpartners.com/research/telco-to-techco-six-tenets-for-success (Accessed 1 February 2024).

[STS18] S. Siami-Namini, N. Tavakoli and A. Siami Namin, "A Comparison of ARIMA and LSTM in Forecasting Time Series," 2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA), Orlando, FL, USA, 2018, pp. 1394-1401, doi: 10.1109/ICMLA.2018.00227.

[SWARM] Docker Swarm [Online]. Available at: https://docs.docker.com/engine/swarm (Accessed: 9 Apr. 2024).

[TAZ+22] I. Tzanettis, C-M. Androna, A. Zafeiropoulos, E. Fotopoulou, S. Papavassiliou. Data Fusion of Observability Signals for Assisting Orchestration of Distributed Applications. *Sensors*. 2022; 22(5):2061. https://doi.org/10.3390/s22052061

[TEL24] Telegraf web page, https://www.influxdata.com/time-series-platform/telegraf/ (Accessed April 2024).

[TER24] TeraFlow project, https://www.teraflow-h2020.eu/ (Accessed April 2024).

[TF09] M. Tsugawa and J. A. B. Fortes, "Characterizing user-level network virtualization: performance, overheads and limits," in International Journal of Network Management, November 2009

[TFS24] ETSI TeraFlowSDN SDG, https://tfs.etsi.org/ (Accessed April 2024).

[THA24] Thanos web page, available at: https://thanos.io/ (Accessed April 2024).

[TMF-634] TMForum 634, “Resource Catalog Management v5.0.0”. September 2023.

[TMF-639] TMForum 639, “Resource Inventory Management API User Guide v4.0.1”, May 2020.

[TMF-640] TMForum 640, “Service Activation and Configuration API User Guide v4.0.1”, Jul 2020.

[TMF-664] TMForum 664, “Resource Function Activation and Configuration API User Guide v4.0.1”, April 2020.

[TMF-IG1230] TMForum IG1230, “Autonomous Networks Technical Architecture v1.1.1”. February 2023.

[TMF-IG1251] TMForum IG1251, “Autonomous Networks – Reference Architecture v1.0.1”. September 2022.

[TMF-IG1253] TMForum IG1253, “Intent in Autonomous Networks v1.3.0”. September 2022.

[TMF-IG1253A] TMForum IG1253A, “Intent Modelling v1.0.0”. July 2021.

[TMF-IG1253E] TMForum IG1253E, “Using Intent in AN – Use Cases, Scenarios and Examples v1.0.0”. September 2023.

[TMF-OA] https://www.tmforum.org/oda/open-apis/ (Accessed April 2024).

[TRE24] TRex. Available at: https://trex-tgn.cisco.com/ (Accessed April 2024).

[VAM+23] R. Vilalta, et al., Demonstration of Intent-Based Networking for End-to-End Packet Optical Cloud-native SDN Orchestration, ECOC 2023.

[VGX+23] Vilalta R, Gifre L, Casellas R, Muñoz R, Martínez R, Mozo A, Pastor A, López D, Fernández-Palacios JP. Applying Digital Twins to Optical Networks with Cloud-native SDN Controllers. IEEE Communications Magazine. 2023 Aug 28.

[VHI+21] K. Vandikas, H. Hallberg, S. Ickin, C. Nyström, E. Sanders, O. Gorbatov, and L. Eleftheriadis, “Ensuring energy-efficient networks with artificial intelligence”, Ericsson Technology Review, April 2022, available at: https://www.ericsson.com/en/reports-and-papers/ericsson-technology-review/articles/ensuring-energy-efficient-networks-with-ai (Accessed April 2024)

[VJP22] F. Vannella, J. Jeong, and A. Proutiere, “Off-Policy Learning in Contextual Bandits for Remote Electrical Tilt Optimization,” IEEE Trans. Veh. Technol., pp. 1–11, 2022, doi: 10.1109/TVT.2022.3202041.

[VMC+21] Vilalta R, Muñoz R, Casellas R, Martínez R, López V, de Dios OG, Pastor A, Katsikas GP, Klaedtke F, Monti P, Mozo A. Teraflow: Secured autonomic traffic management for a tera of sdn flows. In2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit) 2021 Jun 8 (pp. 377-382). IEEE.

[VVG+24] R. Vilalta, F. J. Vílchez, Ll. Gifre, C. Manso, J.L. Carcel-Cervera, R. Leira, J. Aracil-Rico, J.P. Fernández-Palacios, R. Martínez, R. Casellas, R. Muñoz, Providing Anomalous Behaviour Profiling by extending SmartNIC Transceiver support in Packet-Optical Networks, OFC, 2024.

[WAC08] S. Wang, G.S. Avrunin, and L.A. Clarke, (2008). Plug-and-Play Architectural Design and Verification. In: R. de Lemos, F. Di Giandomenico, C. Gacek, H. Muccini, M. Vieira, (eds) Architecting Dependable Systems V. Lecture Notes in Computer Science, vol 5135. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-85571-2_12

[WN10] G. Wang and T. S. E. Ng, "The Impact of Virtualization on Network Performance of Amazon EC2 Data Center," 2010 Proceedings IEEE INFOCOM, San Diego, CA, USA, 2010, pp. 1-9, doi: 10.1109/INFCOM.2010.5461931

[WST11] J. Whiteaker, F. Schneider, and R. Teixeira, "Explaining packet delays under virtualization" in ACM SIGCOMM Computer Communication Review 41(1), pp. 38-44, January 2011, doi: 10.1145/1925861.1925867

[Yan14] X.-S. Yang, Chapter 5- genetic algorithms, in: X.-S. Yang (Ed.), Nature-Inspired Optimization Algorithms, Elsevier, Oxford, 2014, pp. 77–87. doi:https://doi.org/10.1016/B978-0-12416743-8.00005-1

[YLC+21] L. Yang, Y. Lu, J. Cao, J. Huang and M. Zhang, "E-Tree Learning: A Novel Decentralized Model Learning Framework for Edge AI," in IEEE Internet of Things Journal, vol. 8, no. 14, pp. 11290-11304, July 2021, doi: 10.1109/JIOT.2021.3052195

[ZHA20] F. Zhang, L. J. O'Donnell, Chapter 7 - Support vector regression, Editor(s): Andrea Mechelli, Sandra Vieira, Machine Learning, Academic Press, 2020, Pages 123-140, ISBN 9780128157398

[ZSM-002] ETSI GS ZSM 002, “Zero-touch network and Service Management (ZSM); Reference Architecture”. August 2019.

[ZSM-005] ETSI GR ZSM 005, “Zero-touch network and Service Management (ZSM); Means of Automation”. May 2020.

[ZSM-009-1] ETSI GS ZSM 009-01, “Zero-touch network and Service Management (ZSM); Closed loop Automation; Part 1: Enablers”, v1.1.1. June 2021.

[ZSM-009-2] ETSI GS ZSM 009-02, “Zero-touch network and Service Management (ZSM); Closed loop Automation; Part 2: Solutions for automation of E2E service and network management use cases”, v1.1.1. June 2022.

[ZSM-009-3] ETSI GS ZSM 009-03, “Zero-touch network and Service Management (ZSM); Closed loop Automation; Part 3: Advanced topics”, v1.1.1. August 2023.

[ZSM-011] ETSI GR ZSM 011, “Zero-touch network and Service Management (ZSM); Intent-driven autonomous networks; Generic aspects”, v1.1.1 February 2023.

[ZYD+24] C. Zhou, H. Yang, X. Duan, D. Lopez, A. Pastor, Q. Wu, M. Boucadair, and C. Jacquenet, “Network Digital Twin: Concepts and Reference Architecture”, v0.5, IRTF Internet Draft, March 2024.

8 Appendix

8.1 Information models

This section describes the information models defined in the Hexa-X-II M&O system.

8.1.1 Information model for Trustworthy 3P

This section describes the Trustworthy 3P information model, with the attributes of the various information elements detailed in Table 8-1 - Table 8-4 below.

The “Identity” class includes the following attributes:

Table 8-1: “Identity” class attributes.

Attribute	Description	Type	Multiplicity	Support
identityName	Specifies a readable string to uniquely identify a role. *Allowed values: N/A*	String	1	M
identityType	Indicates the type of user identity. *Allowed values: user_name, email_address, phone_number, IPv4_address, Ipv6_address, FQDN.*	ENUM	1	M
credential	Specifies the credential of the user identity, used for the authentication. It can be secret or certificate-based assertion (NOTE 1) *Allowed values: N/A*	String	1	M
authnMethod	Specifies the method used by the authentication function for authentication purposes. *Allowed values: client_Credentials, mutual_TLS*	ENUM	1	CM
roleList	Specifies the collection of roles associated with an identity. It lists the distinguished name (DN) of those <<role>> class instances (NOTE 2).	DN	N	M
NOTE 1: Secrets work with REST/OpenAPI, while certificate-based works with Netconf/NACM. NOTE 2: A Distinguished Name (DN) is used to uniquely identify an instance of a class within a name space. A DN is built from a series of "name components", referred to as Relative Distinguished Names (RDNs). ITU-T Recommendation X.500 [ITU-X500] defines the concepts of DN and RDN in detail, using ASN.1.				

The “Role” class includes the following attributes:

Table 8-2: “Role” class attributes.

Attribute	Description	Type	Multiplicity	Support
roleName	Specifies a readable string to uniquely identify a role. *Allowed values: N/A*	String	1	M
accessRuleList	Specifies the collection of access rules allocated to a role. It lists the distinguished name (DN) of those <<accessRule>> class instances (NOTE 1).	DN	N	M

The “AccessRule” class includes the following attributes:

Table 8-3: "AccessRule" class attributes.

Attribute	Description	Type	Multiplicity	Support
ruleName	Specifies a readable string to uniquely identify an access rule. *Allowed values: N/A*	String	1	M
resourceType	Identifies the type of resource protected by this access rule. *Allowed values: network_node, cloud_native_function, network_slice_subnet, network_slice, sub_network.*	ENUM	N	M
node	Represents the management properties of the resource protected by this access rule. These properties are specified in a standardized information model (NOTE 1). This attribute points to the DN where the information model is stored. *Allowed values: N/A*	DN	1	M
nodeScope (NOTE 2)	Selection criteria for the target management properties of the resource. Only the properties that pass the criteria will be protected by access rules. *AllowedValues: <attributes>*	ENUM	N	O
nodeFilter (NOTE 3)	Filter criteria for the target resource instances. Only the instances that pass the criteria will be protected by the access rule. *AllowedValues: area_of_interest&=, admin_domain&, network_domain=&*	ENUM	N	O
permissionList	The list of permissions defined in the access rule.	DN	N	M
NOTE 1: These models can be found in standardization bodies like 3GPP (e.g. NRM fragments in TS 28.622 [28.622] and 28.541 [28.541]), IETF (RFCs for L2/L3NM and L2/L3SM e.g., RFC8466, RFC8299, RFC9291, RFC9182). NOTE 2: If absent, all properties of this resource will be protected by the access rule. NOTE 3: If absent, all instances of this resource will be protected by the access rule.				

The "permission" dataType includes the following attributes:

Table 8-4: "Permission" dataType attributes.

Attribute	Description	Type	Multiplicity	Support
permissionId	Specifies a UUID for the permission. *Allowed values: N/A*	String	1	M
permissionName	Name for the permission. *Allowed values: N/A*	String	1	O
action	Specifies a CRUD operation to gain access to the resource. *Allowed values: CREATE, READ, UPDATE, DELETE*	ENUM	N	M
policy	This allows enabling/disabling the action. *AllowedValues: ENABLED, DISABLED*	ENUM	N	O

8.1.2 Information models for Resource Orchestration in the continuum

Data models for resources and application graphs

For supporting orchestration in the compute continuum two data model aspects are introduced:

- a representation of the resources, i.e. a model of the computing and the network infrastructure,
- an application graph representation, modelling designed service communication patterns and data transfers.

Taking into consideration a multi-cluster infrastructure, the main entities constituting the computing representation should include the cluster and the corresponding physical or virtual computing nodes of each cluster. The network representation is described as a graph of network connections (links) between network nodes (e.g. routers/switches), while it is also crucial to provide support for virtual links created by network controllers (e.g. SDN) to make network performance guarantees. Runtime information of applications placed in the continuum is important to be registered close to the infrastructure for supporting orchestration actions. Specifically, a deployment identifying the placement cluster and replication factor of each service represents the application instance, while an individual runtime instance records a replica's resource utilization and state (see Figure 8-1).

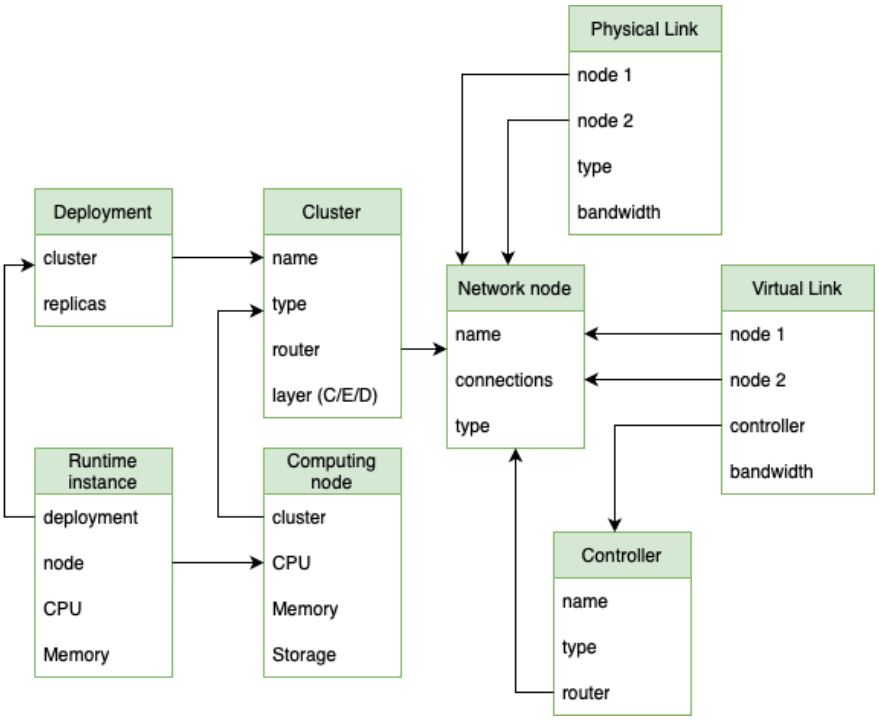


Figure 8-1: Data model for resources.

A detailed application description is important for identifying its underlying complexity and its connectivity characteristics that can be utilized to optimize its performance. An application graph is defined as a network of services interacting with each other through their endpoints. For one service to access another, a service call (link) is made having a specific payload size. A sequence of service calls makes up a directed acyclic graph (DAG), a workflow executing a certain functionality. The service's runtime information is recorded in the infrastructure model so that each service maps to a deployment (see Figure 8-2).

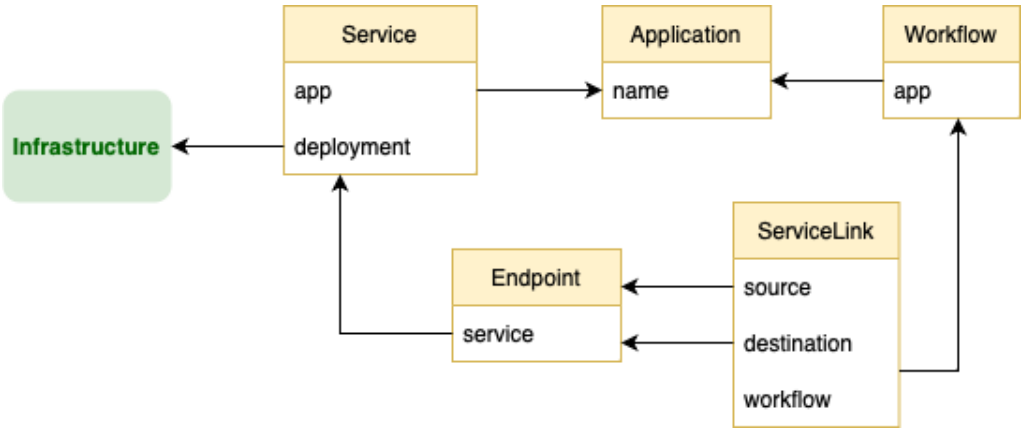


Figure 8-2: Data model for application graph.

TMF Data models for resource specification

Orchestration across the continuum, with the use of resources from different nature (cloud, edge, extreme edge), requires an important resource abstraction and service description work that provides a common language (agnostic of the technology) for managing these resources and digital services. This approach is followed by TMForum that defines a resource catalogue with specifications [TMF-634] related with logical resources, physical resources, and resource functions (an abstraction of ETSI NFV concepts like VNF, forwarding graphs, network services, etc), as can be seen in Figure 8-3.

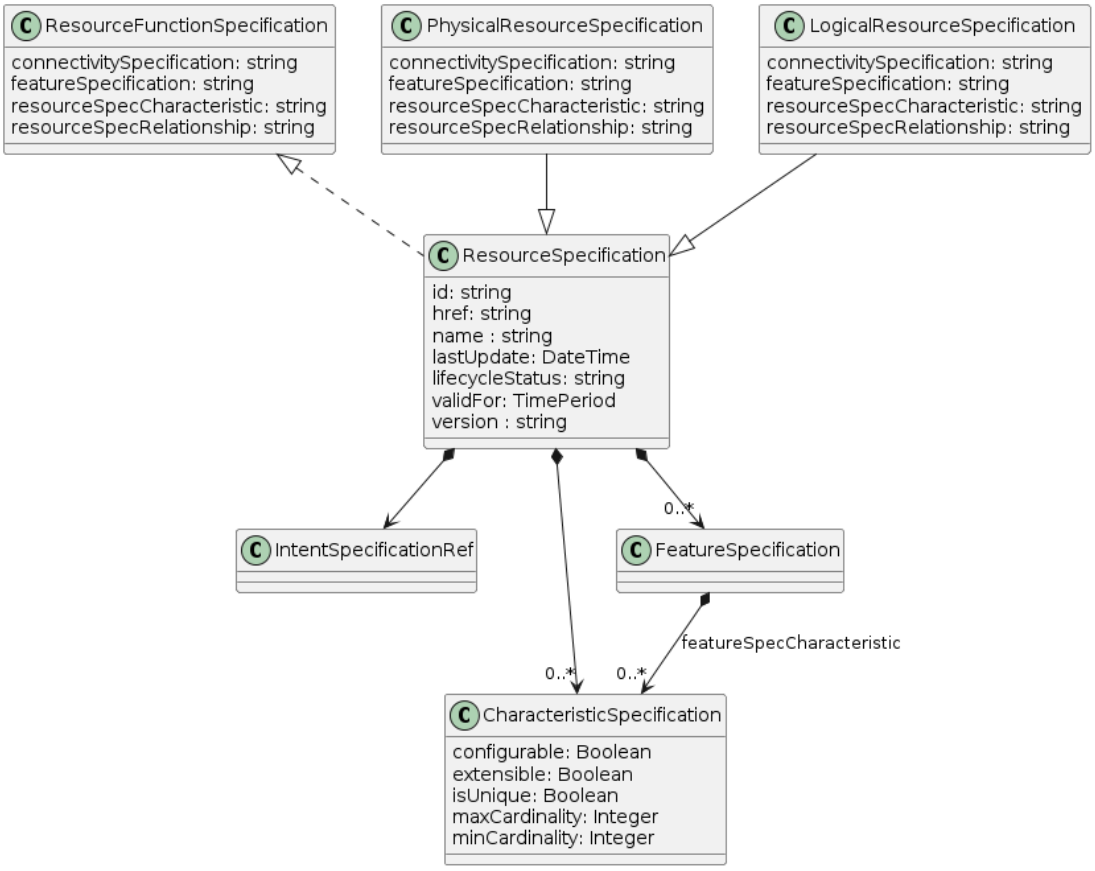


Figure 8-3: Resource Specification data model.

These definitions are complemented with a resource inventory for instantiated resources [TMF-639] and the activation and configuration of the resource functions (lifecycle management including tasks for scaling, healing, and migration [TMF-664]).

Resource level can be linked with service level through the well-known CFS/RFS TMForum view, using the service catalogue definitions (Figure 8-4 shows the service specifications definition). Finally, the business

level can be managed as well with the product catalogue, inventory and ordering definitions, covering different layers of the orchestration process.

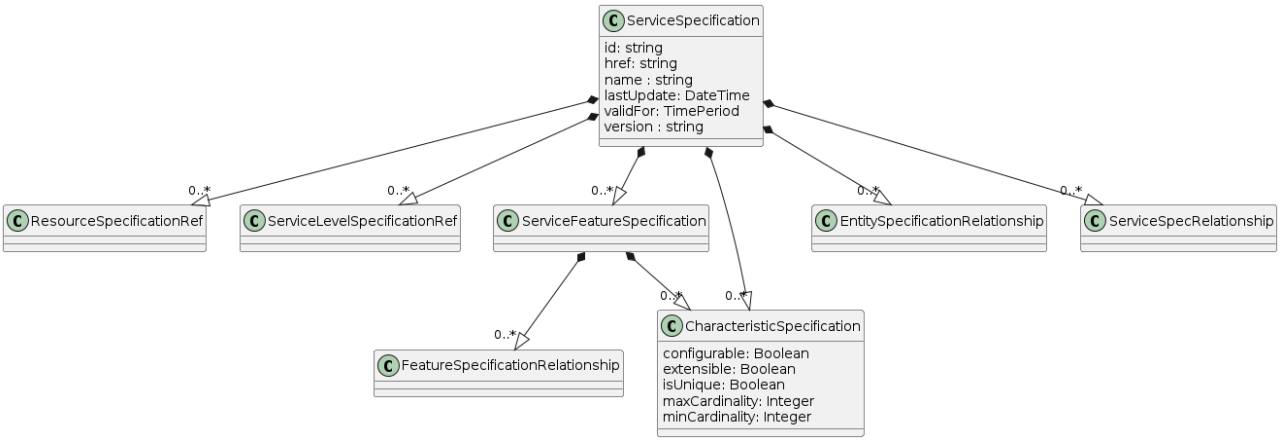


Figure 8-4: Service Specification data model.

In summary, the TMForum approach provides a robust framework for resource abstraction and service description across the service continuum. Their relevance is amplified by the growing complexity of networks that span cloud, edge, and extreme edge environments. The widespread adoption of these frameworks by the community underscores their importance in enabling efficient and scalable service delivery in an increasingly interconnected and decentralized digital landscape.

Data models for extreme edge nodes in the continuum

The solution proposed for the modelling of the extreme-edge, edge and cloud continuum resources in the REC-EXEC software aims at:

- support multiple and heterogeneous virtualization platforms and their computing capabilities (e.g., Kubernetes, K3s, Microk8s, OpenStack, etc.);
- seamlessly support various continuum segments (e.g., extreme-edge, edge or cloud);
- support different types of extreme-edge devices (e.g., IoT devices, sensors and actuators, robots and cobots, etc.), making use of ontologies to better characterize specific extreme-edge device categories.

Extensible and abstract information models, followed by specialized ones for relevant categories of extreme edge nodes, have been implemented and integrated in the *Resource Orchestrator for Continuum across Extreme-edge, Edge and Cloud* (REC-EXEC) software prototype. The REC-EXEC collects platform-level and device-level information about clusters nodes in the extreme-edge, edge and cloud continuum and then maps these data in specific information models.

Figure 8-5 depicts the graph of data models leveraged by REC-EXEC.

The *Node* information model describes generalised computing capabilities and virtualization platform information, deployment domain, and per-device information, possibly based on standard semantic models.

The *platformNodeInfo* field of the Node information model is an instance of a concrete class that extends the *PlatformNodeInfo* abstract class representing the computing and virtualization platform-specific information of the Continuum node. *PlatformNodeInfo* and its specific subclasses aim to represent CPU, RAM and storage capabilities, GPU capabilities, SR-IOV support, virtualization platform type and its specific information.

The *nodeDomain* field of the Node information model is an instance of a concrete class that extends the *NodeDomain* abstract class representing the location of a node in the extreme-edge, edge and cloud continuum.

The *deviceInfo* field of the Node information model is an instance of a concrete class that extends the DeviceInfo abstract class or a concrete class that extends one of the abstract classes that represent a specific device family; this field aim to represent device capabilities, characteristics and constraints, information for generic devices and attributes for each device category, additional parameters for context and environment description, common semantics modelling for device categories (e.g., based on standard ontologies like Dronetology for Drones and OCRA for collaborative robots, etc.), with extensions for each instance.

To support the deployment of service applications a platform-agnostic service description information model has been designed in the context of the development of the REC-EXEC software component.

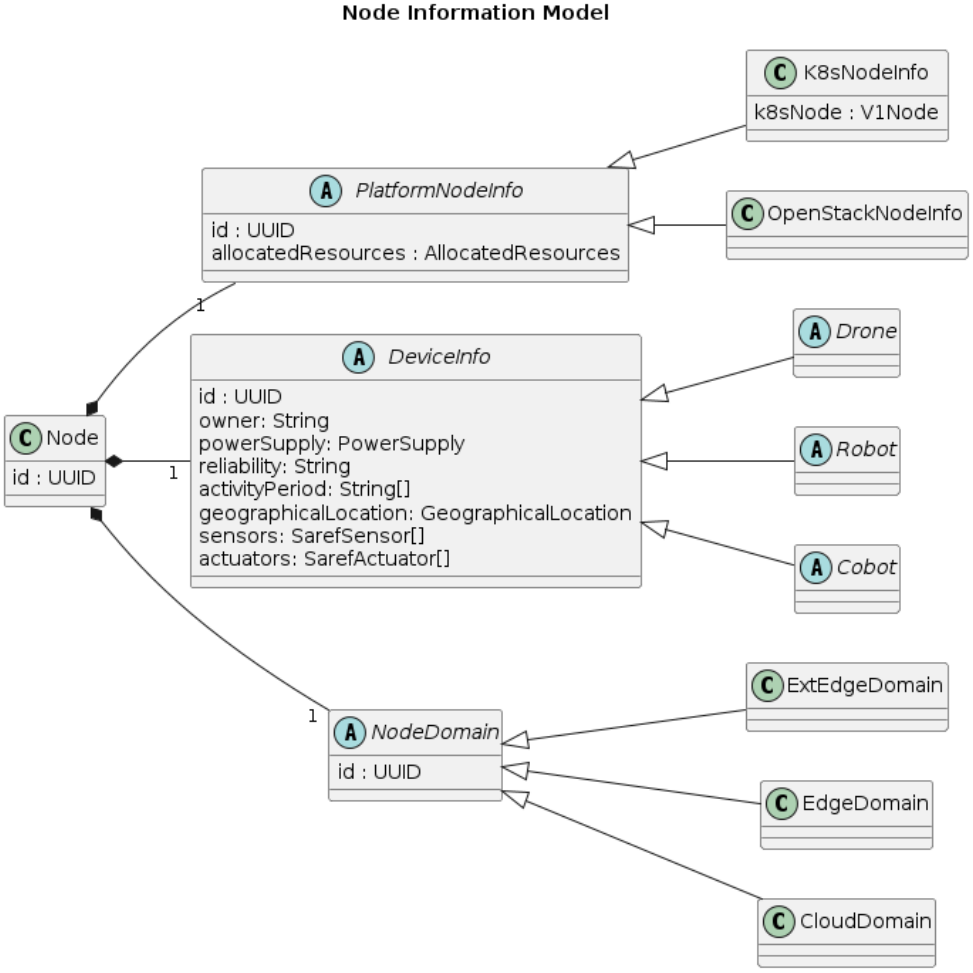


Figure 8-5: Node data model.

Figure 8-6 shows the graph of the data models involved in the definition of a service application.

The information model used to deploy service applications describes the target virtualization platform, the service components of the service application, the high-level platform agnostic compute requirements (e.g., CPU, memory, storage, etc.) of each service component, the characteristic and placement for each service component, the orchestration template to be used to orchestrate each service component (e.g., Helm Chart or plain descriptors for Kubernetes-based platforms, Heat Template for OpenStack, etc.).

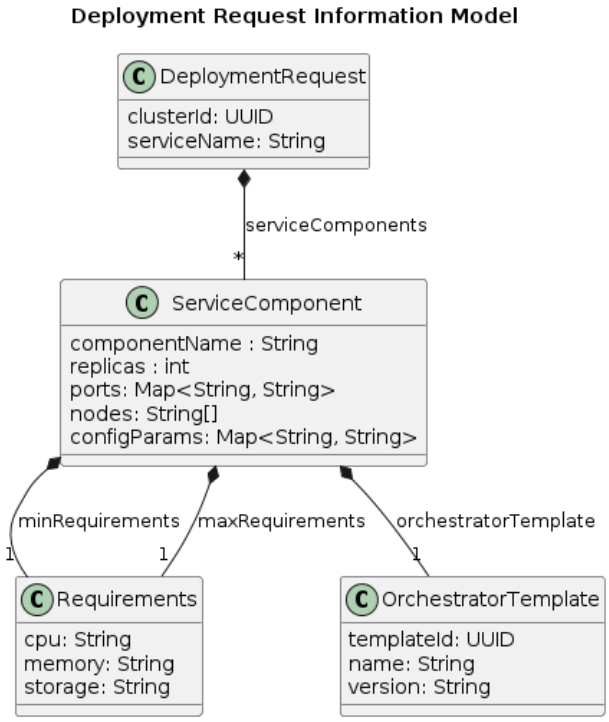


Figure 8-6: Data model for service application.

Data models for functionality allocation

The data model used internally by the functionality allocation component is reported in Figure 8-7.

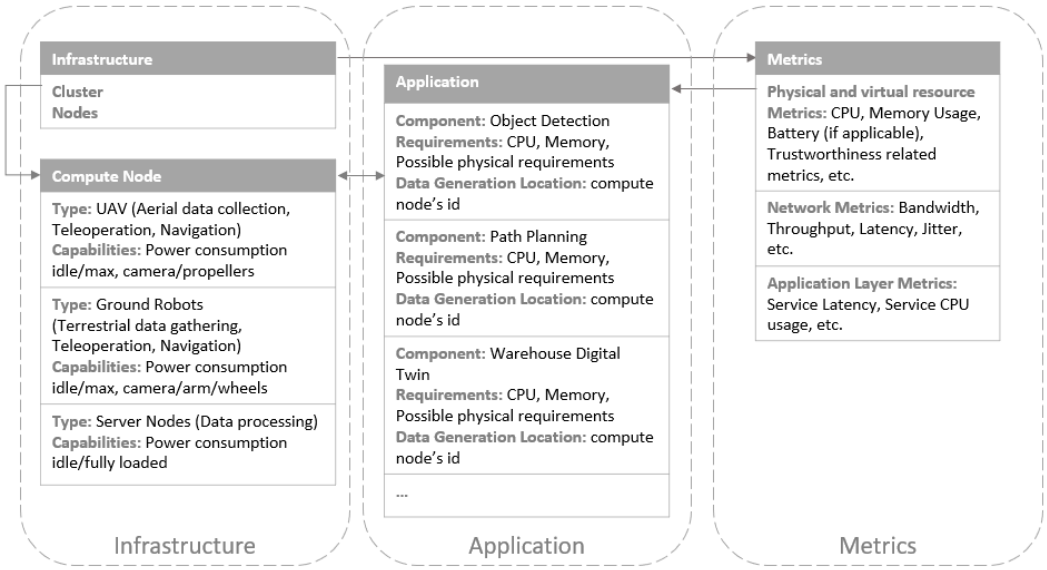


Figure 8-7: Data model used for functionality allocation system.

8.1.3 Information model for Closed Loop Descriptor

The information model of a generalized CL Descriptor is depicted in Figure 8-8, Figure 8-9, and Figure 8-10 with a UML representation.

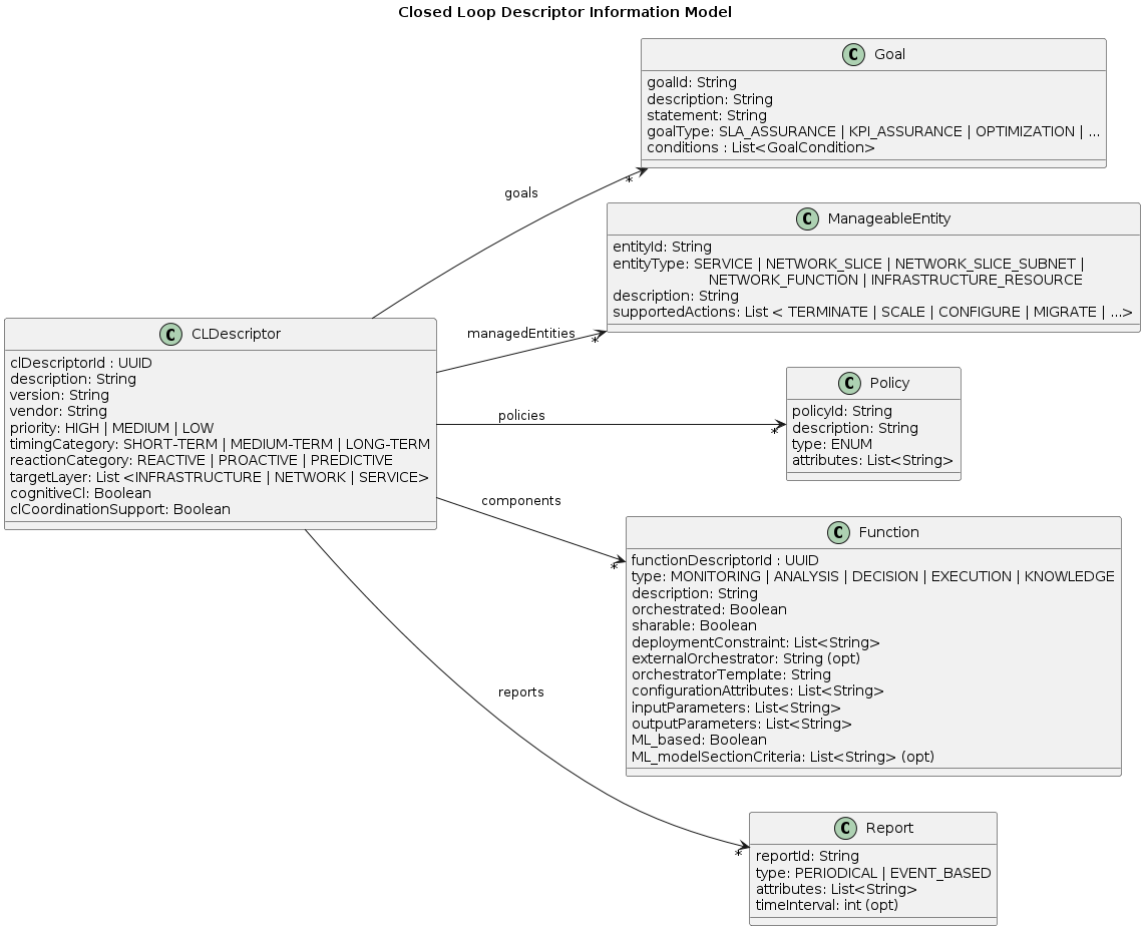


Figure 8-8: Information model for CL Descriptor (1).

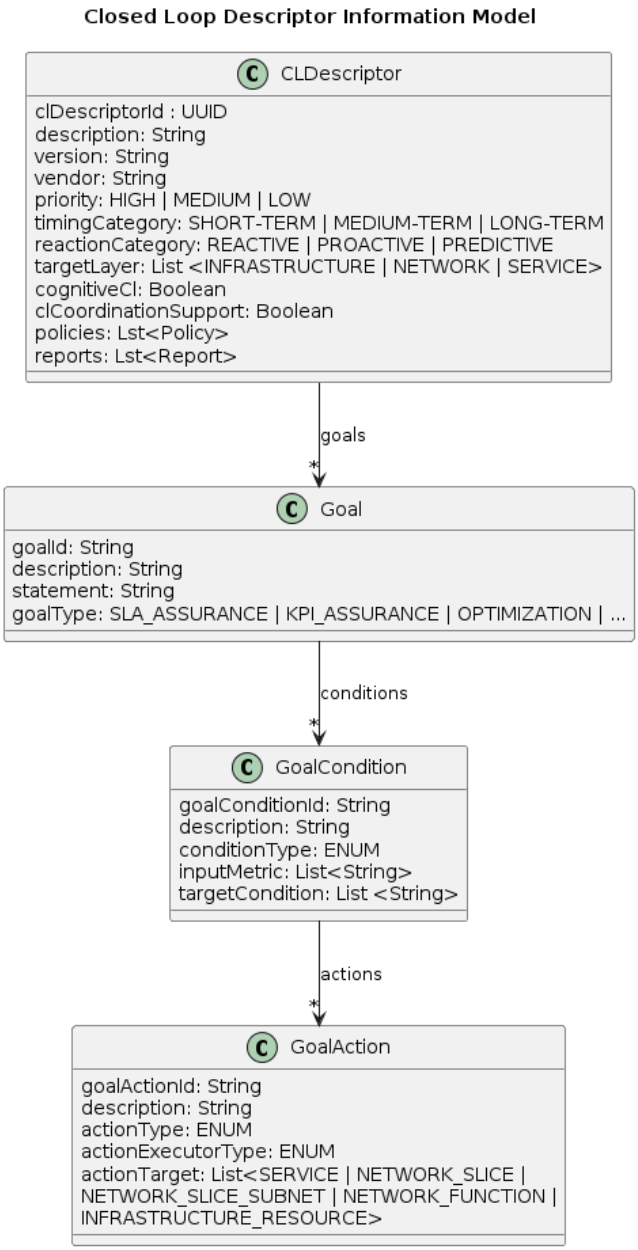


Figure 8-9: Information model for CL Descriptor (2).

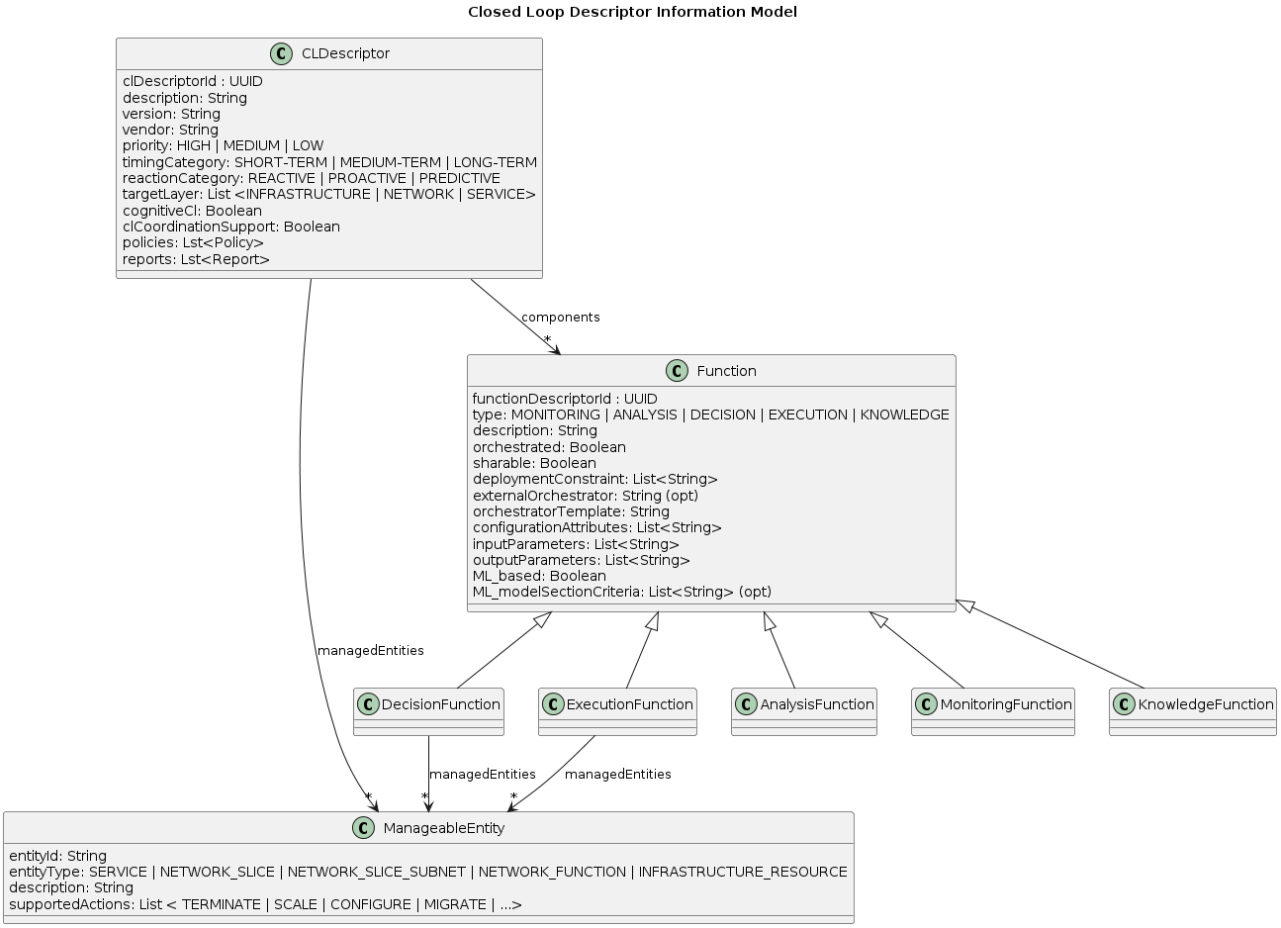


Figure 8-10: Information model for CL Descriptor (3).

8.2 Interfaces

This section describes the interfaces defined for the enablers presented in Section 3.

8.2.1 Interfaces of SDN Orchestrator and SDN controllers

This section describes the interfaces of the End-to-End SDN Orchestrator (section 8.2.1.1) and the Technological-Domain SDN controller (section 8.2.1.2), which were presented in section 3.1.

8.2.1.1 E2E SDN Orchestrator

The exposed interfaces are described in Table 8-5.

Table 8-5: E2E SDN Orchestrator exposed interfaces.

Interface name	Service descriptions	Potential consumers	Possible Standards
NBI E2E SDN Orchestrator	Support for Transport Network Slices	OSS/BSS	IETF draft-liu-teas-transport-network-slice-yang
SBI E2E SDN Orchestrator	Layer 2 Virtual Private Network (L2VPN) Service Delivery	Technological Domain SDN controller	IETF – RFC 8466
SBI E2E SDN Orchestrator	Connectivity Service, Topology Discovery, Telemetry	Technological Domain SDN controller	ONF Transport API

8.2.1.2 Technological-Domain SDN controller

The exposed interfaces are described in Table 8-6.

Table 8-6: Technological-Domain SDN controller exposed interfaces.

Interface name	Service descriptions	Potential consumers	Possible Standards
NBI (IP)	Layer 2 Virtual Private Network (L2VPN) Service Delivery	E2E SDN Orchestrator OSS/BSS	IETF – RFC 8466 [RFC8466]
SBI (IP)	Router configuration and monitoring	Technological Domain SDN controller	OpenConfig gNMI
NBI (Optical)	Connectivity Service, Topology Discovery, Telemetry	E2E SDN Orchestrator OSS/BSS	ONF Transport API
SBI (Optical)	ROADM configuration	Technological Domain SDN controller	OpenROADM

8.2.2 Interfaces of the Monitoring and Telemetry framework

Figure 8-11 shows the interfaces between the monitoring framework (enabler #2) and the closed loops framework (enabler #8), mediated through the Management Capabilities Exposure framework implemented with Kafka (enabler #3).

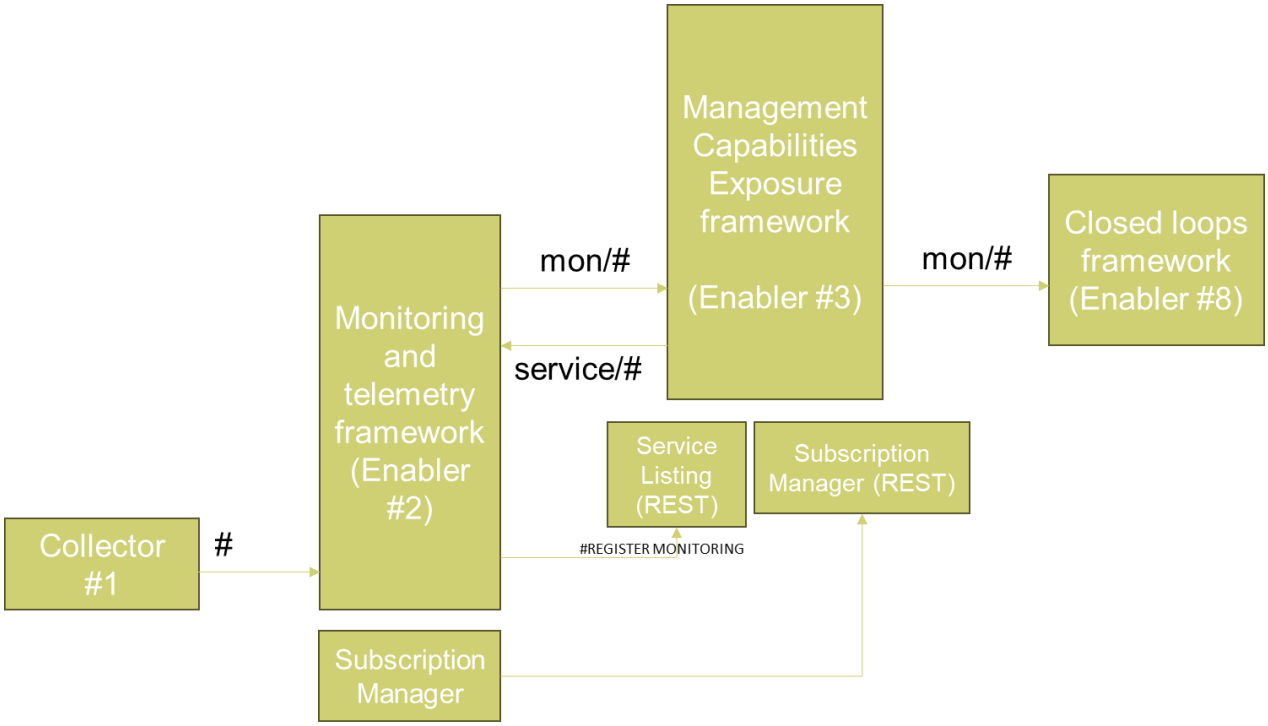


Figure 8-11: Integration between Monitoring and telemetry framework and other enablers.

Organizing topics for monitoring and telemetry in a data bus involves structuring them in a way that is logical, scalable, and facilitates efficient data distribution. The possible topics for exposing data in the data bus are described below.

- **Hierarchical Structure:** Organize topics hierarchically to represent the structure of the system. For example, you could have a top-level topic for each major component, and subtopics for specific functionalities or data types within each component. As an example for this structure, possible topics are listed.

  - /component1/functionality1
  - /component1/functionality2
  - /component2/dataType1

- **Functional Categories:** Group topics based on the functionality or purpose of the data. This makes it easier to navigate and understand the purpose of each topic. As an example for this structure, possible topics are listed.
  - /sensorData/temperature
  - /sensorData/humidity
  - /systemStatus/alerts

- **Wildcard Subscriptions:** It is possible to use wildcard subscriptions to subscribe to multiple topics with a single subscription. This can be useful for scenarios where a component needs to receive a broad range of data. As an example for this structure, possible topics are listed.
  - /component1/#

- **Time Series Data:** If dealing with time-series data, it is possible to consider including a time-related hierarchy to organize data by timestamps. As an example for this structure, possible topics are listed.
  - /component1/sensorData/temperature/2023/11/16
  - /location2/sensorData/temperature

- **Priority Levels:** Assign priority levels to topics based on the criticality of the data. This can help prioritize processing for critical information. As an example for this structure, possible topics are listed.
  - /highPriority/alerts
  - /lowPriority/status

- **System Components:** Use topics to represent different system components, making it easy to isolate and monitor specific parts of the system. As an example for this structure, possible topics are listed.
  - /frontend/logs
  - /backend/logs

- **Event Types:** Categorize topics based on event types to distinguish between different types of data. As an example for this structure, possible topics are listed.
  - /event/authentication
  - /event/error

- **Geographical or Logical Partitioning:** If your system has geographically distributed components, consider organizing topics based on location. As an example for this structure, possible topics are listed.
  - /location1/sensorData/temperature

8.2.3 Interfaces of the Management Capabilities Exposure framework

This section describes the interfaces of the Management capability exposure framework, for subscriptions (Table 8-7), service listing (Table 8-8) and security (Table 8-9) functionalities.

Table 8-7: Subscription API.

REST operation	Resource	Description
POST	/register	An enabler can use this endpoint to onboard in the system, by presenting a correct message body.
DELETE	/unregister	An enabler can use this endpoint to offboard from the system, by presenting a correct message body.
POST	/registerNewTopic	An enabler can use this endpoint to register a new communication channel in addition to the custom one related to the registration. This

		new communication channel can be fixed or volatile, i.e. will be cancelled after it is no more in use
POST	/registerSchema	An enabler can register a new Schema using this endpoint.
OpenAPI available at: https://app.swaggerhub.com/apis/telefonica-646/integration-fabric-subscription-api/1.0.0		

Table 8-8: Listing API.

REST operation	Resource	Description
GET	/listAllTopic	An enabler can use this endpoint to retrieve all the topics onboarded in the system.
GET	/topic/{common-name}/info	An enabler can use this endpoint to retrieve more detailed info about a specific topic.
GET	/listAllOnboardedEntities	This endpoint makes possible to list all the onboarded entities common names.
GET	/listOnboardedGroupNames	This endpoint makes possible to list all the onboarded entities consumer group names.
GET	/listSingleSchema/{schema}	An enabler can use this endpoint to retrieve more detailed info about a specific schema.
GET	/listAllSchemas	An enabler can use this endpoint to retrieve all the schemas stored in the registry.
OpenAPI available at: https://app.swaggerhub.com/apis/telefonica-646/integration-fabric-listing-api/1.0.0		

Table 8-9: Security API.

REST operation	Resource	Description
GET	/sign-certificate/{common-name}	The party using this endpoint can retrieve the keystore, fundamental to communicate with the Kafka Cluster
GET	/ca-cert	The party using this endpoint can retrieve the certification authority public certificate, fundamental to communicate with the Kafka Cluster
POST	/modifyPermission	The party presenting a correct message body can modify a specific ACL related to a topic.
OpenAPI available at: https://app.swaggerhub.com/apis/telefonica-646/integration-fabric-security-api/1.0.0		

8.2.4 Interfaces of CL Governance and CL Coordination functions

The interfaces of the CL Governance and Coordination services are reported in Table 8-10.

Table 8-10: CL Governance and Coordination interfaces.

Interface name	Service descriptions	Producer	Potential consumers	Possible Technologies
CL_LCM_Notifications	Allows to publish information on changes on the LC of CLs, including registration of new CL instances and their interdependent CLs (independent vs nested vs hierarchical).	CL Governance	CL Coordination Service	MQTT
CL_LCM_Operation	Allows to operate over LC of CLs (pause, unpause, remove…)	CL Governance	Service/Network Orchestrator CL Coordination Service	REST
CL_Configuration	Allows to configure objectives, targets, internal policies, and in general configuration attributes for clusters of CLs.	CL Governance	Service/Network Orchestrator CL Coordination Service	REST
CL_Decision	Allows to expose the required action plan to be executed by a CL to achieve their goals/kpis, requesting permission from CL Coordination Service (e.g escalation notification). It includes information of managed entities, predicted result, interdependent actions (nested and hierarchical CLs). Via queries or subscribe/notify mechanisms.	CL Governance	CL Coordination Service	REST MQTT
CL_Execution	Allows/Deny the execution of the requested action plan. The original action plan could be modified by CL Coordination Service to avoid conflict between CLs (via queries or	CL Coordination Service	CL Governance	REST MQTT

	subscribe/notify mechanisms).			
CL_Conflict	Provides information on managed conflicts.	CL Coordination Service	Service/Network Orchestrator	REST
CL_Impact	Allows to request the evaluation of the impact of stages from a set of CLs	CL Coordination Service	Service/Network Orchestrator	REST

8.3 Overview of TM Forum, IETF, and ETSI Framework Alignments

8.3.1 NBI TMForum

The major goal is to align the concepts defined by TM Forum through "Autonomous Networks Technical Architecture" [TMF- IG1230] with "ETSI GR ZSM 011 – Zero-touch network and Service Management" [ZSM-011] (see also enabler #8 in section 3.8) and correlated with IETF Network Slice Service Interface. This alignment can be discussed in terms of interoperability, technological innovation, and the potential for future research and development. The following image presents a diagram that highlights the integration and alignment of telecommunications standards related to autonomous network architectures, featuring documents and frameworks from TM Forum, ETSI, and IETF to illustrate the technical and strategic overlaps in network management.

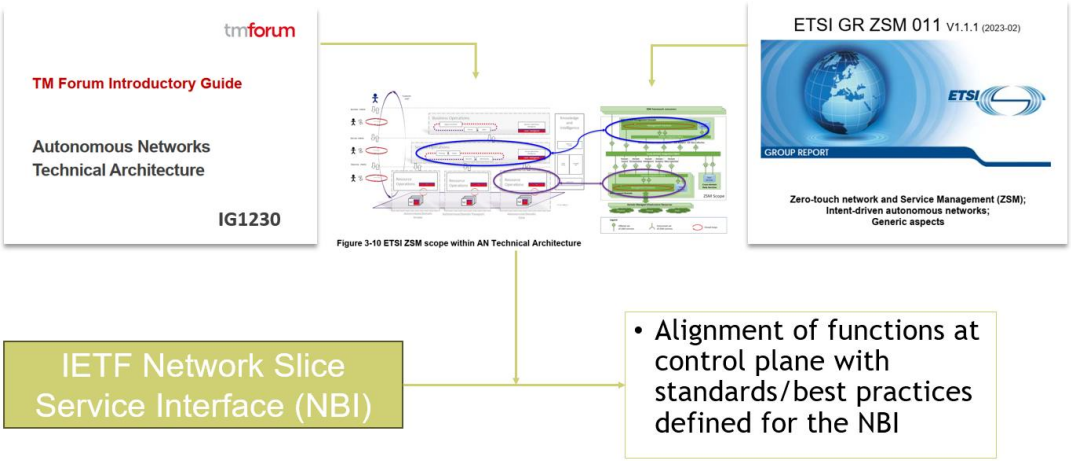


Figure 8-12: Overview of standards related to NBI for autonomous networks.

8.3.2 Perspective on TM Forum APIs and IETF Standards

8.3.2.1 TM Forum - Interoperability and Standardization

API Standardization for Operational Efficiency

TM Forum's Open APIs [TMF-OA] are used for operational automation in telecommunications networks. **Service Activation and Configuration API**, for example, explores its impact on reducing manual intervention and enhancing service reliability. These standardized APIs lead to consistency in service deployment and maintenance across diverse network infrastructures.

Integrating AI for Network Optimization

TM Forum's APIs like Analytics and AI Management are pivotal in integrating AI into network operations. This enables the analysis of the effectiveness of AI algorithms in network traffic prediction, resource allocation, and anomaly detection. This integration is crucial in evolving networks, especially with the

increasing complexity and data volume, where AI-driven insights can significantly optimize network performance.

Customer-Centric Services and AI

Customer Management and **Product Catalogue Management APIs** are focal points for research on customer-centric services. AI, enabled by these APIs, can be used for personalized service offerings based on customer behaviour analysis, thus enhancing user satisfaction and engagement.

AI-driven Personalization

Leverage AI to analyse customer data for personalizing services. This includes studying predictive models for customer preferences and usage patterns, which can inform dynamic service offerings and proactive customer support.

8.3.2.2 TM Forum alignment with IETF Standards

NFV and SDN Management

The alignment between TM Forum's network APIs and IETF's work on NFV and SDN is a critical area of research, and the project focuses on how TM Forum's Resource Function Activation and Configuration API manages virtual network functions within the frameworks provided by IETF standards, examining the efficacy and challenges in managing these next-generation network paradigms. Regarding the deployment and management of network services the focus is on the Service Activation and Configuration API.

The Resource Function Activation and Configuration API from TM Forum is a crucial link in aligning with IETF's proposal on network resource M&O, particularly in environments like NFV and SDN. This API plays a pivotal role in automating the activation and configuration of Virtual Network Functions (VNF) and other network resources. Its alignment with IETF is evident in how it facilitates the implementation of IETF-defined protocols and architectures for NFV and SDN. For instance, the API's capabilities in dynamically configuring network resources are crucial in a scenario where IETF standards define the underlying network behaviours and protocols. By enabling efficient and scalable management of network resources, this API contributes to the realization of more agile, responsive, and efficient network environments. It effectively bridges the gap between high-level service management, as advocated by TM Forum, and the granular control of network functionalities as defined by IETF standards.

The Service Activation and Configuration API from TM Forum aligns with IETF's efforts in streamlining the deployment and management of network services. This API is central to the operationalization of services over the network infrastructure, including those that utilize IETF's protocols and data models. By standardizing the processes involved in service activation, modification, and decommissioning, the API ensures that services are deployed with consistency and reliability, adhering to the specifications and parameters as defined by IETF standards. It plays a key role in translating customer-focused service requests into actionable configurations on the network. This alignment is particularly significant in the context of complex services, such as those in multi-layered networks or 5G environments, where IETF's protocols and network slices come into play. The API's functionality thus not only complements IETF's technical standards but also reinforces TM Forum's commitment to enhancing service delivery and customer experience in telecommunications networks.

8.3.2.3 Intent-Based Management and Autonomous Networks

Alignment in achieving network operational goals through automation and AI

In the next image we have a schematic representation of the interaction between various TM Forum APIs within a network management system, specifically highlighting, in a red square, the roles of TMF640 and TMF664 APIs in service and resource function activation. These APIs interface with a programmable service controller - an AI/ML-based system tasked with the start, configuration, and monitoring of services, which is part of the broader domain automation framework. We have as well TMF628 and TMF635 APIs, which manage performance and service usage, respectively, illustrating how they interact to monitor and report the status of services. It represents the data flow between initiating service requests, configuring service elements, and monitoring through advanced AI/ML functions, providing a holistic view of how network services are managed dynamically and intelligently.

Unified Network Management Approach

TM Forum's APIs (TMF640-Service Activation Configuration and TMF664 – Resource Function Activation) can manage network resources conforming to IETF's protocols, ensuring cohesive network management strategies that align technological advancement with operational requirements.

Network Slice Management and YANG Models

The YANG data model defined by IETF is foundational for implementing standardized network slicing capabilities, allowing for detailed and secure configuration, management, and operation of network slices across various network environments. It supports the establishment of network slices with specific service-level objectives and expectations, facilitating advanced network management and customization capabilities as envisioned in the broader context of autonomous networks and intent-driven networking.

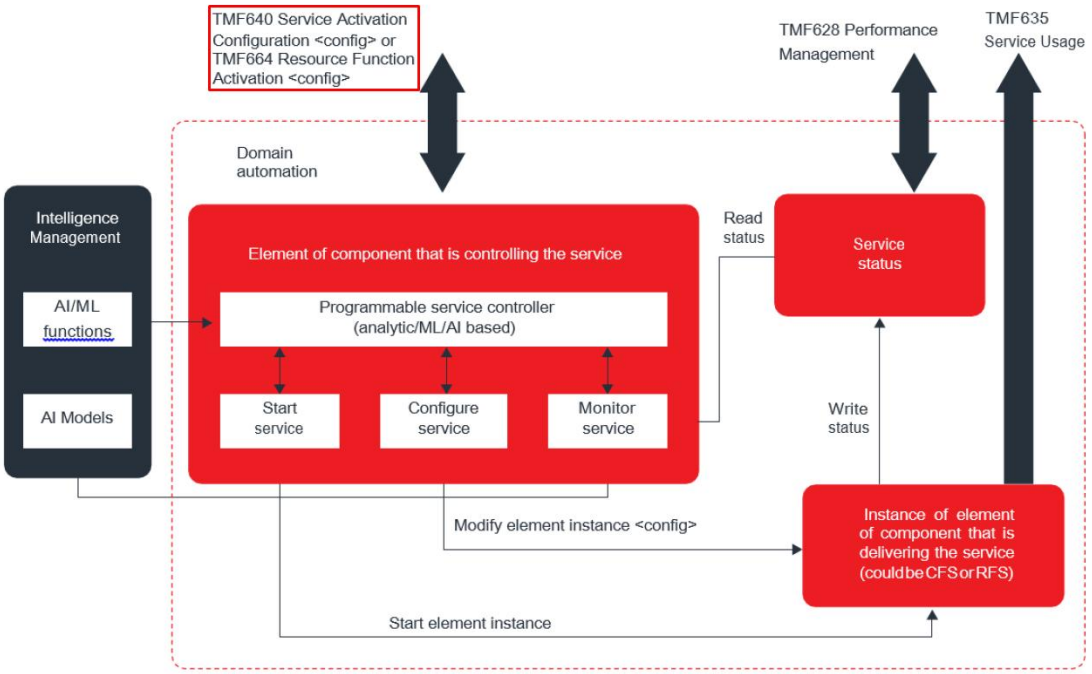


Figure 8-13: TM Forum's APIs for Domain Automation – an example from [TMF-IG1230].

TMF and IETF alignment

The following image illustrates the alignment between the IETF Network Slicing Frameworks and the TMF Service Management Approaches, highlighting the convergence of technologies and methodologies that are instrumental in modern network management.

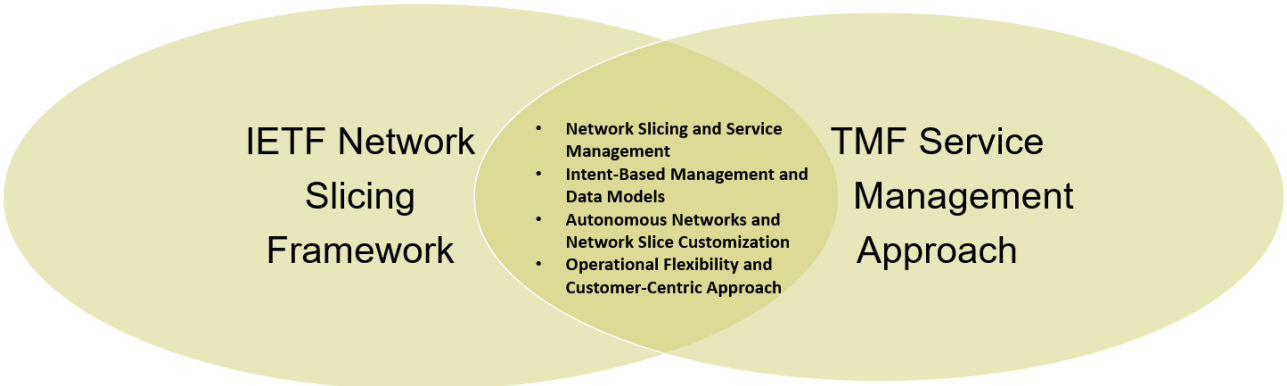


Figure 8-14: IETF Network Slicing Framework and TMF Service Management Approach intersection.

On one side, the IETF Network Slicing Frameworks are designed to provide a technical foundation for network slicing, which includes the creation, operation, and termination of network slices that cater to various service levels across a shared physical infrastructure. This encompasses intent-based management and data models,

which enable networks to understand and execute on high-level business objectives, and the capacity for network slice customization, which allows for the tailored creation of network slices according to specific service requirements. Autonomous networks are also a focal point, emphasizing the need for networks that can self-manage based on predefined policies and objectives, enhancing operational efficiency and agility.

On the other side of the diagram, TMF Service Management Approaches encapsulate a series of best practices and standardized APIs that enable service providers to manage customer services effectively. These approaches ensure operational flexibility, allowing service providers to quickly adapt to market changes and evolving customer needs. Moreover, they prioritize a customer-centric approach, ensuring that the services provided align closely with customer expectations and improving overall customer satisfaction.

At the intersection, the alignment of these two paradigms is clear: both IETF and TMF are converging on the need for network slicing and service management that are both responsive to the demands of next-generation networks and grounded in the customer's strategic business objectives. By combining the technical prowess of IETF network slicing with the operational and service-oriented focus of TMF, the industry can move towards more holistic, agile, and customer-focused network services. This synergistic approach is fundamental to the future of network operations, ensuring that networks are not only technically robust but also aligned with the business goals and service quality that customers expect.

TMF and ETSI alignments in IETF Network Slice Service Interface

The alignment between TM Forum and the ETSI within the context of the IETF Network Slice Service Interface represents a significant stride toward harmonizing network management and operational practices across different layers of network infrastructure. This collaboration is particularly pivotal as it merges TM Forum's expertise in service management frameworks and business process standardization with ETSI's technical standards, including those for NFV and multi-access edge computing (MEC). The integration of these standards within the IETF Network Slice Service Interface facilitates a more unified approach to network slicing. By aligning TM Forum's service management APIs with ETSI's network slicing specifications, network operators can leverage a comprehensive set of tools for creating, managing, and terminating network slices, ensuring that these slices meet the specific requirements of various services, from high-throughput video streaming to ultra-reliable low-latency communications in 5G and beyond networks.

The practical outcome of TM Forum and ETSI alignments in the IETF Network Slice Service Interface is the enablement of a seamless, end-to-end network slicing management framework that supports the dynamic and complex needs of modern telecommunications networks. This framework benefits from TM Forum’s rich set of management and operational APIs that offer standardized procedures for service orchestration, customer management, and resource optimization. Concurrently, it incorporates ETSI’s technical standards, which provide the necessary definitions and architectural foundations for network slicing across core, transport, and edge segments. This alignment not only streamlines the deployment and management of network slices but also fosters innovation by providing a clear, standardized interface for introducing new services and capabilities. Additionally, it enhances interoperability among service providers, allowing for the creation of global, cross-network services that can operate seamlessly over multiple administrative domains. Such advancements underscore the importance of TM Forum and ETSI's collaboration, marking a significant step forward in the evolution of network services and the broader goals of digital transformation in the telecom sector. The following diagram represents the interaction between TM Forum, ETSI and IETF NBI and the aspects of implementation that can be achieved.

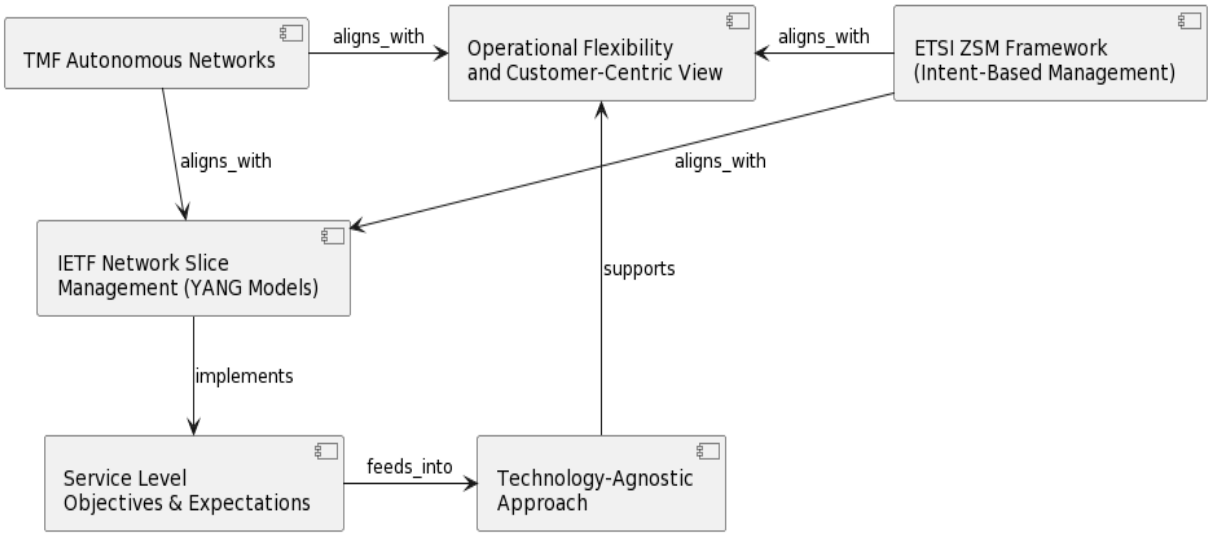


Figure 8-15: TM Forum, ETSI and IETF NBI alignment.

Intent Standards Classification

The following table presents a structured comparison between the TM Forum, ETSI's ZSM/ENI (Zero-touch network & Service Management / Experiential Networked Intelligence), and IETF (Internet Engineering Task Force), highlighting their respective contributions to the conceptualization, modelling, architectural definition, classification, and domain intent categorization within the framework of network operations and service management.

Table 8-11: Intent Standards Classification.

	Concept Definition	Model	Architecture	Classification	Domains	Intent Category
TM Forum	IG 1230	IG1253	IG1251 and IG1253	IG1253A	FFS in IG1253E	Business, Service Resource
ZSM/ENI	ZSM005/ ZSM011 (in 2021)	ENI PoC8	ENI005, ITANA			Service, Resource
IETF	RFC7575 draft-irtf-nmrg-ibn-concepts-definitions-03	RFC 8049 RFC 8466 RFC 8969 draft-ietf-opsawg- l3sm-l3nm- 09 draft-yang-nmrg- network measurement-intent-01	draft-ietf-opsawg-service-assurance-architecture-00	draft-irtf-nmrg-ibn-intent-classification-03		Network / Domain Specific

8.4 Related technologies

8.4.1 The compute continuum and the convergence with the edge

The orchestration of the computing continuum present specific management challenges directly related to the integration and use of the Edge Computing technology. Among all the challenges, it stands out above all, the discovery and the selection of edge resources that better fit with the user requirements of latency, CPU,

memory, energy consumption, prize, etc. In this context, the Linux Foundation's Camara project [CAM24] is working on the definition of APIs related with the edge capabilities exposure and the network management:

- Edge Capabilities exposure:
  - **Simple Edge Discovery**: get the name of the closest MEC to the network-attached device that hosts the application that made the API request (closest in terms of shortest network path)
  - **MEC Exposure and Experience Management**: a richer APIs that include Edge application lifecycle workflows and session management, that are based on GSMA Operator Platform Architecture. From the developer point of view, it supports provisioning operations (to retrieve the list of MECs and their status, discover the capabilities/resources available in the MEC in terms of CPU, memory, storage, GPU, to get details of all the onboarded applications,…) and runtime operations (discover the closest MEC platform to a particular terminal, to obtain the optimal MEC for an application and a particular terminal taking account connectivity, shortest network path, cost, network load, etc). On the same way, from the operator perspective, it support provisioning intents (to publish a list of MECs, their coverage, capabilities and status, to map application's requirements to the best MEC for hosting it, based on app demands for CPU, memory, storage, GPU, bandwidth, mobility…) and runtime intents (to publish events to developers about which changes which MEC is optimal for their application and connected terminals).
- Network management
  - **Traffic Influence**: provides the ability to ask for the minimal end to end latency in a specific geographical area to enhance the quality of experience leveraging on local instances of an application deployed on the Edge, activating the appropriate route. This operation links directly with the work that is being analysed in Enabler 1.