ETSI GR ZSM 009-3 V1.1.1 (2023-08)

Zero-touch network and Service Management (ZSM); Closed-Loop Automation; Part 3: Advanced topics

Disclaimer

The present document has been produced and approved by the Zero-touch network and Service Management (ZSM) ETSI Industry Specification Group (ISG) and represents the views of those members who participated in this ISG.
It does not necessarily represent the views of the entire ETSI membership.

Reference

DGR/ZSM-009-3_Cla_AdvTop

Keywords

automation, closed control loop, management, service

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° w061004871

Contents

Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (https://ipr.etsi.org/).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

DECT™, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™** and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

Foreword

This Group Report (GR) has been produced by ETSI Industry Specification Group (ISG) Zero-touch network and Service Management (ZSM).

The present document is part 3 of a multi-part deliverable. Full details of the entire series can be found in part 1 [i.2].

Modal verbs terminology

In the present document "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the ETSI Drafting Rules (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

1 Scope

The present document investigates advanced topics related to closed-loop operations such as learning and cognitive capabilities (e.g. based on different degrees of use and integration of artificial intelligence technologies), ways to set and evaluate levels of oversight, autonomy, and operational confidence on the behaviour of the closed loops. The present document will document problem statements and technical challenges, derive potential requirements, capture, and evaluate potential solution options, and provide recommendations for further standardization activities.

2 References

2.1 Normative references

Normative references are not applicable in the present document.

2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non-specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document, but they assist the user with regard to a particular subject area.

[i.1] S. Ayoubi, N. Limam, M.A. Salahuddin, N. Shahriar, R. Boutaba: "Machine Learning for Cognitive Network Management". IEEE Communications Magazine. Vol. 56(1), pp. 158-165, January 2018.

[i.2] ETSI GS ZSM 009-1: "Zero-touch network and Service Management (ZSM); Closed-Loop Automation; Part 1: Enablers".

[i.3] ETSI GS ZSM 009-2: "Zero-touch network and Service Management (ZSM); Closed-Loop Automation; Part 2: Solutions for automation of E2E service and network management use cases".

[i.4] ETSI GR ZSM 011 (V1.1.1): "Zero-touch network and Service Management (ZSM); Intent-driven autonomous networks; Generic aspects".

[i.5] TM Forum IG1253 (V1.2.0): "Intent in Autonomous Networks".

[i.6] Russell, Stuart J.; Norvig, Peter (2003) (2nd ed.): "Artificial Intelligence: A Modern Approach". Upper Saddle River, New Jersey: Prentice Hall. Chapter 2. ISBN 0-13-790395-2.

[i.7] ETSI GS ZSM 001: "Zero-touch network and Service Management (ZSM); Requirements based on documented scenarios".

[i.8] Stephen S. Mwanje, Christian Mannweiler: "Towards Cognitive Autonomous Networks" Publisher: WILEY.

[i.9] ETSI GS ZSM 002 (V1.1.1): "Zero-touch network and Service Management (ZSM); Reference Architecture".

[i.10] ETSI TS 128 312 (V17.1.1) (2022-10): "LTE; 5G; Management and orchestration; Intent driven management services for mobile networks (3GPP TS 28.312 version 17.1.1 Release 17)".

3 Definition of terms, symbols and abbreviations

3.1 Terms

Void.

3.2 Symbols

Void.

3.3 Abbreviations

For the purposes of the present document, the following abbreviations apply:

5G	Fifth Generation of mobile networks
AI	Artificial Intelligence
C-MAPE	Cognitive Control Loop
CL	Closed Loop
CLC	Closed Loop Controller
CPU	Central Processing Unit
IME	Intent Management Entity
IoT	Internet of Things
KPI	Key Performance Indicator
LTE	Long Term Evolution
M2O-CL	Made-to-Order Closed Loop
MD	Management Domain
ML	Machine Learning
OWL	Web Ontology Language
PM	Performance Management
QoE	Quality of Experience
RDF	Resource Description Framework
SLA	Service Level Agreement
SON	Self Organizing Networks
TOSCA	Topology and Orchestration Specification for Cloud Applications
TS	Traffic Steering
UE	User Equipment
VNF	Virtual Network Function

4 Next-generation closed-loop operations

It is expected that the future network management systems should be able to handle the increased complexity in the evolving network scenarios arising from diverse network deployments options and massive scalability requirements. As part of the evolution, the management of the network should be automated as much as possible. Accordingly, the management system, specifically the next generation of closed-loop operations, should be able to handle the additional requirements through machine-learning-driven or cognitive automation. Cognition built from learning based on different network and environment events will be used to derive insights and decisions on network and service management scenarios involving single or multiple network events. As described in [i.1] closed loops may benefit from applying machine learning in any of the stages, to make the closed loops more adaptive and self-learning.

As described in [i.8], the evolution towards autonomous networks empowered with cognitive capabilities will happen in stages starting with today's automated network employing SON as closed loops involving a set of automation capabilities configured and managed by the operator. The intermediate stage is the automated network, which adds cognitive capabilities to the management layer thereby eliminating the need for the operator to handle the configuration and management of the network management automation capabilities that constitute the closed loops. The configuration and management for the automation capabilities may be selected based on prevailing contexts in the network environment. Eventually, the operations will evolve further into a fully autonomous network in which cognitive capabilities are introduced into the closed loops of managed and management layers. These learning driven closed loop are then capable of learning the right configuration parameters for different network contexts and update the learning in the context model in a continuous manner. Autonomous networks enable the service user and the network operator as management service consumers to provide their expectations as "intents". The intent-based system converts the intent to objectives which are further processed by different closed loops to satisfy the expectations expressed by the intent. The system accepts input data from different sources to gather different perspective on the network and also the impacts from configuration and environmental changes. This will enable the system to contextualize the information with multi-dimensional perspective of the network behaviour (network performance and user experience) based on different scenarios (configuration, environment, user specific handling, etc.).

This study investigates various challenges in network automation and potential advanced closed loop solutions such as cognitive closed loops, intent based closed loops, dynamic closed loop composition and coordination, inter-dependent closed loop, etc. to support cognitive autonomous network.

5 Problem Statements

5.1 Cognitive closed loops

Closed loops have multiple stages which could benefit from applying Machine Learning (ML) technology. According to the C-MAPE principle [i.1] a cognitive control loop may incorporate ML into any of its stages: ML-based monitoring, analysis, decision making and optimized actions. Such approach makes a CL more adaptive and self-driven/self-learning, as opposed to traditional CLs that are usually self-regulating (implement a pre-defined control logic like in LTE/5G SON) but not self-learning (i.e. they do not change the control logic regardless of the outcome of their actions).

Applying ML within a CL requires the integration of one or more trained ML models in the CL implementation. With state-of-the-art ML methods, an ML model architecture (e.g. neural network layers, activation functions) defines the syntax and semantics of both the input data and the model output. That is, each ML model takes a set of well-defined input data and produces inference results of a specific kind. Due to such focus of a single model, a CL may benefit from having multiple models, each operating on different sets or types of input data and producing different outputs. The reasons for multiple models may include:

- Using multiple models in combination to implement a more complex analytics and decision pipeline (e.g. by considering insights generated by different analytics algorithms in parallel and/or in series). In this case, the involved models are simultaneously active and should be supplied with their respective input data.

- Some models are alternatives to others with different level of functionality or capability. For instance, different models may be created to analyse time series data coming in at different measurement intervals (5 minutes, 15 minutes, 1 hour). Depending on the resolution of the input data available in the deployment of the CL, the corresponding model is selected. Additionally, if the availability of the input data changes (e.g. a data source stops sending data for a period of time), the CL may autonomously switch between models.

- Some models implement special analysis cases that are only needed under specific circumstances. For example, in a hierarchical analysis, high level analysis and decision may be driven by a first model, whereas special non-anticipated cases, such as suspicion of an anomaly, triggers further analysis using other models. As the various models require different input data, the CL may need to activate specific data sources so that they start to produce data required by a newly activated model.

The above considerations forecast a more dynamic interplay between stages of a CL compared to a mostly linear collection-analysis-decision-actuation cycle. Such dynamism also contributes to improved level of autonomy for the CL, such as:

- Proactively re-program data sources to autonomously collect the right type, quality, and quantity of data. The data quality and quantity of data may be a pre-requisite for versatile and high-quality analytics (or to be able to perform analytics at all).

- With enhanced analytics, CLs may implement self-learning capability by analysing the outcome of the actions and learning from a feedback based on how well the actions have reached the goal for which the actions have been initiated.

5.2 Composition and coordination of interdependent closed loops

Multiple closed loops may belong to the same use case with a common goal or related goals. These interdependent CLs may act directly or indirectly on the same managed entity. Hence, the predicted actions of these interdependent CLs may conflict with each other resulting in an undesired operation on the managed entity.

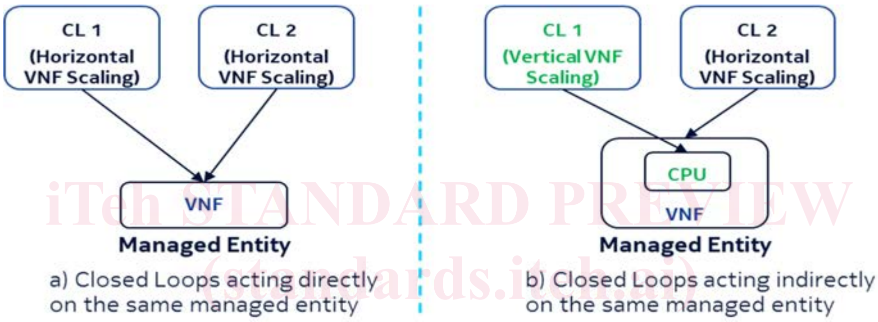

Figure 5.2-1: Example for Interdepended Closed loops

- As shown in figure 5.2-1a), two or more instantiated CLs may act *directly* on the same managed entity, e.g. two CLs acting on the same Virtual Network Function (VNF) instance for horizontal VNF autoscaling.

- As shown in figure 5.2-1b), two or more instantiated CLs may act *indirectly* on the same managed entity, e.g. when one CL acts on the CPU resource of the VNF instance for vertical VNF autoscaling and another CL acts on that same VNF instance for horizontal VNF autoscaling.

Challenges:

Currently, external coordination mechanism (i.e. via CL Coordinator as shown in figure 5.2-1) is employed to detect and mitigate CL conflicts either before (i.e. Pre-action coordination) or immediately after they occur (i.e. Post-action coordination).

In Pre-action coordination, the CL coordinator detects and resolves conflicts between CLs before triggering the Execution stage of the CLs.

In Post-action coordination, the CL coordinator identifies CL actions after the Execution stage that lead to undesired outcomes and determines how to avoid such conflicts in the future.

However, external coordination mechanism assumes that all CLs are independent and can thus be coordinated as independent entities. Consequently, the CLs are currently always composed independently of each other, even though the CLs are interdependent belonging to the same use case with common goal or related goals.

With hundreds or thousands of CLs instantiated in the network, some of these CLs may be nested or hierarchical. Scaling the functionality of the CL Coordinator becomes a challenge.

The potential solutions to address these challenges are specified in clause 6.3. **Related Requirement** are:

- ETSI GS ZSM 001 [i.7] clause 5.2 Req #70 "*ZSM framework shall support the capability of nested closed loops*".

- ETSI GS ZSM 009-2 [i.3] clause 5.2 [CL-general-2] "*The ZSM framework shall allow establishing different types of relationships between Closed Loops. NOTE 2: Examples of relationships are peer Closed Loops, hierarchical or nested Closed Loops.*"

The potential solutions to address these challenges are specified in clause 6.2.

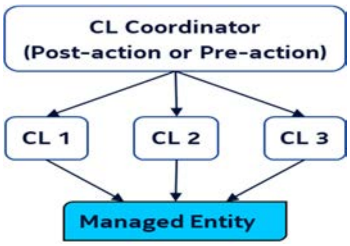

Figure 5.2-2: External Closed Loop Coordinator-1

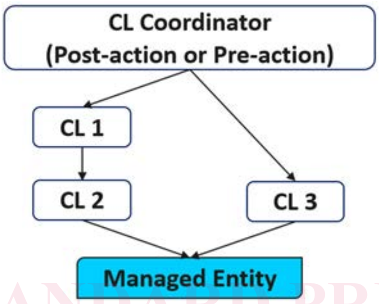

Figure 5.2-3: External Closed Loop Coordinator-2

5.3 Dynamic composition of closed loops

Closed loops may be, in general, within a single management function that combines all the closed loop stages or functionalities. In such cases, multi-vendor interactions need interfaces to the network resources while control and exposure interfaces focus on the capabilities of the combined function.

Alternatively, for multi-vendor network automation environments, closed loops support specific modules for accomplishing specific tasks, e.g. to provide a specific analysis on some data. CLs may be composed from multi-vendors components using standardized services. As illustrated by figure 5.3-1, the different stages can support control and exposure services (marked with "C" and "E" in figure 5.3-1), through which an authorized entity (e.g. the ZSM framework consumer) may control the stages. Each stage also supports the run-time services (marked with D in figure 5.3-1) through which the stages exchange data to accomplish the network management tasks.

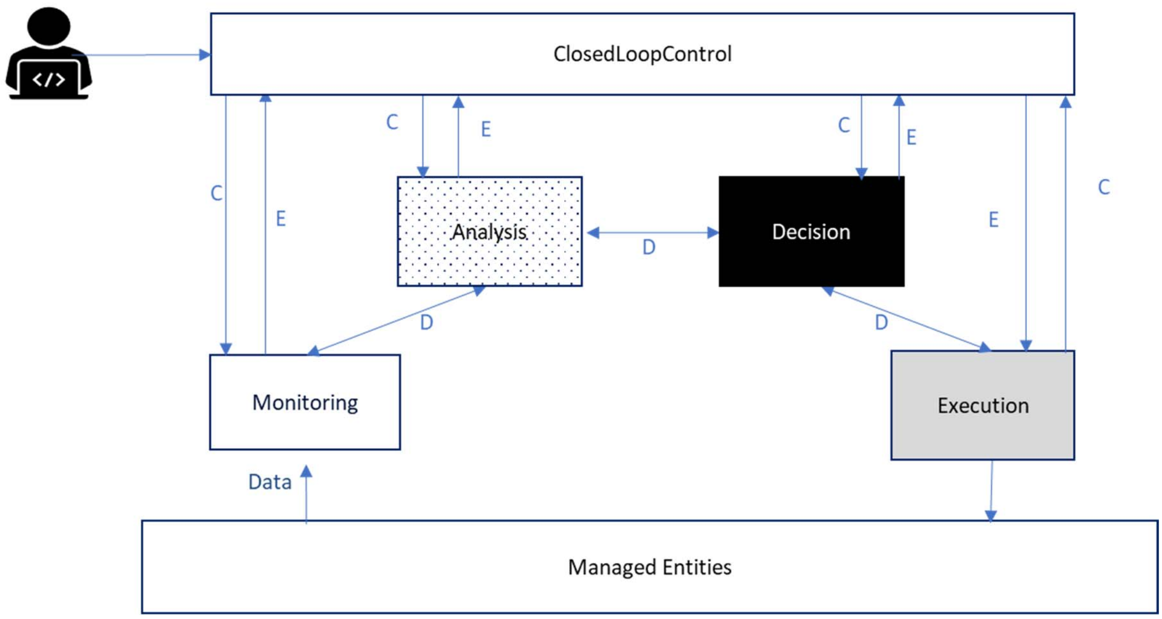

Figure 5.3-1: Automatic composition and management of multi-vendor closed loop

Given the likely high number of CLs and combinations thereof, it would be a heavy task for the human operators to manage. Hence It is necessary to provide automation means with which CL stages may be dynamically composed.

Challenges:

Made-to-Order Closed Loops (M2O-CL), as defined in clause 7.4 of ETSI GS ZSM 009-1 [i.2] and service capabilities further described in clause 5.4.6 of ETSI GS ZSM 009-2 [i.3], are assembled on demand by ZSM framework owner, or by other entities on behalf of the ZSM framework owner, using capabilities offered by the ZSM framework. Components of M2O-CLs may come from different ZSM framework vendors and can be associated with a CL instance based on demand. As described in ETSI GS ZSM 009-2 [i.3], it defines the capabilities needed to chain the different components together to prepare a M2O-CL for deployment.

The composition process - according to current solution - does not support means to request for the dynamic composition of the closed loop. Accordingly, such means do not reflect sufficient flexibility to automatically decide the number and type of needed stages and how to handle situations where one or more needed components are not available. Moreover, a flexible and abstract way to submit a M2O-CL composition request needs to be supported.

To provide a flexible/abstract way to submit a M2O-CLs composition it is important.

- To determine the minimum needed information to form M2O-CLs. The needed information may be:
  - Catalogue(s) of CL components
  - CL goal and description
  - Required capabilities of management functions that combine to form M2O-CLs
- To determine the required number of stages and capability to form M2O-CLs.
- To configure the M2O-CL stages, e.g. with their sources of data or where to deliver their reports.
- To map output of one stage to the input of subsequent stages e.g. data transformation might be required between the stages to facilitate mapping.

5.4 Intent-based closed loops

Intelligent networks need to have the ability to adapt to new situations and new services without the intervention of human operators or experts. One needed feature that is necessary for any intelligent network is to decouple the requirements that are given to the management systems from the solutions that are used to deploy and manage the services. As defined in ETSI GR ZSM 011 [i.4], intents are the formal definition of the requirements, goals, and constraints that a technical system receive and should be used as the main input information for autonomous management of the network.

An intent is a declarative information object that allows the system to understand utility and the value of its actions from a producer's perspective. This allows the system to become autonomous and evaluate situations and potential action strategies, rather than being just limited to follow instructions that operators and experts have specified in policies. This means that intelligent decisions that were previously made exclusively by policy developers can become automated.

On one hand, intents are necessary for providing information about requirements and utility, which is one of the foundations for autonomous systems; on the other hand, closed loops are the key enablers for automating the configuration of networks and steering their state towards the wanted state. The combination of these two concepts, i.e. intents and closed loops - which is needed to realize intelligent and autonomous networks - is still a challenging task.

As described in ETSI GR ZSM 011 [i.4], the management domains may contain Intent Management Entities (IME), which are responsible for managing the life cycle of intents. This implies that an IME is able to detect whether the intent has been fulfilled and, if not, find and execute corrective actions; therefore, IMEs have to leverage closed loops functionalities while dealing with the intent lifecycle management.

Figure 5.4-1 shows the main interactions between IMEs within different management domains (which may include the end-to-end management domain). The IME receives all intents directed toward its autonomous domain and it has to report back to the intent origin regarding its success in fulfilling the intent. The intent fulfilment has to be executed based on measurements that allow the IME to compare the state of the network to the expected state derived from the intent. If the system is not meeting the requirements, corrective actions are needed. The actions can be through conventional management interfaces or, if the targeted is intent-aware, by defining its requirements through another intent.

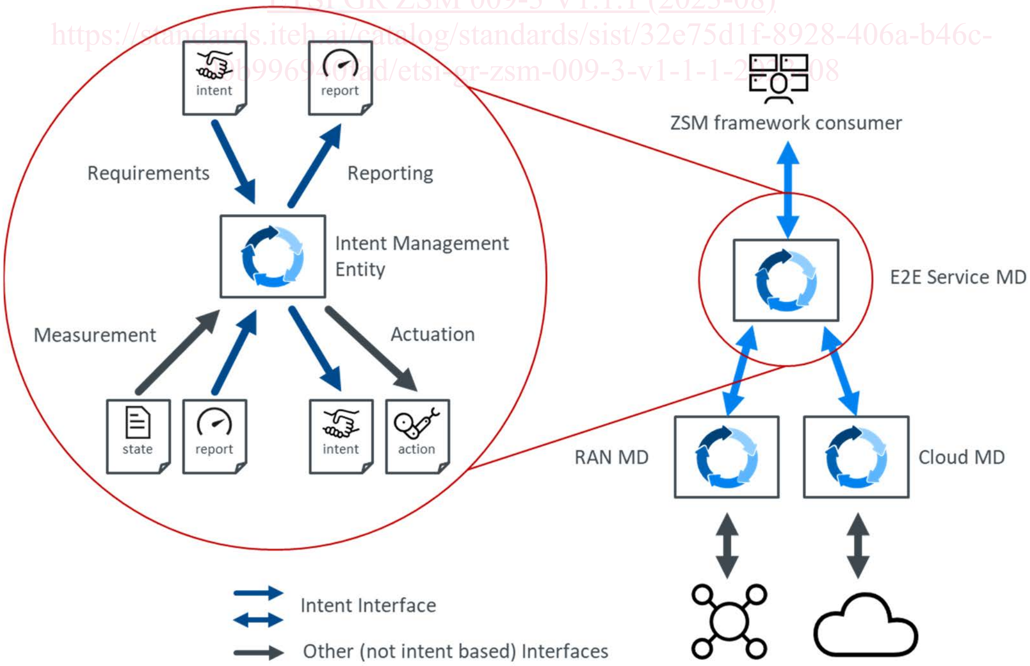

Figure 5.4-1: Automatic composition and management of multi-vendor closed loop

Challenges:

Intent-based operations are well explained in ETSI GR ZSM 011 [i.4] and in other standards ([i.5] and [i.10]).

As investigated in ETSI GR ZSM 011 [i.4], the IMEs operate different closed loops that are responsible to produce and maintain the outcomes expected from the intent received, e.g. a service with specified QoE.

The implementation of IMEs and the use of closed loops can be realized in different forms, but a general challenge is how the IMEs interact with the specific business logic within a management domain, so that they can together translate an intent that expresses business needs into detailed technical configurations.

Such translation can be done in generic way, such closed loop enablers can be used to create a generic closed loop structure that is able to deal with any type of intent, given they follow a common syntax and modelling. This would relieve the burden of implementers that would not need to design and code specific (and different) closed loops for each type of intent that is employed.

The challenge that should be tackled is to create intent-driven closed loops that have a generic structure and that:

i) are able to deal with any type of intent, and

ii) can be specialized to translate business logic into technical configurations in any management domain.

5.5 Resource locality and scarcity on closed loop automation

The implementation of a closed loop may require a certain number of resources. The number of resources may potentially grow with the complexity of the activities expected from the closed loops. Complex operations that are likely to be performed by a closed loop might include artificial intelligence (AI) model training, data analysis and other non-trivial decision-making processes. At the same time, some resources that are supposed to be managed by closed loops may be hosted in local nodes, quite far (and in some cases, isolated) from central nodes. This is the situation that industry foresees with the on-going trend of IoT-edge-cloud continuum, where managed resources do not only include only the ones hosted in centralized and regional data centers, but also on extreme-edge nodes (also known to as the far-edge).

As for the management of extreme-edge node resources, there exists different variants.

On the one hand, implementing the closed loop locally, on the extreme-edge node; this means the closed loop and the resources under its managed scope will both run on the same execution environment. The main issue of going for this approach is that it may be very costly (per compute unit), or ineffective, as it would deplete a large part of the scarce resources available in the extreme-edge node.

On the other hand, implementing the closed loop remotely, on a central node. However, this approach might pose serious issues related to the exchange of data and control flows between the extreme-edge and central nodes. Examples of these issues can include:

- disconnection risks between both nodes, which may result in non-acceptable service downtime;
- data sharing: moving local data from extreme-edge node to central node for their processing may raise privacy concerns, especially when these data hold sensitive (business-critical or regulation-subjected) data;
- performance degradation, in terms of latency (e.g. increased time to send the monitoring data and receive action plans) and backhaul capacity (more exchange of data and control flows between nodes mean higher bandwidth consumption, which might result in poor resource efficiency and congestion).

Challenges:

Currently, resource locality is not taken into account when considering closed loop instantiation and operation. While cognitive and self-learning closed loops are expected to bring better results than traditional non-self-learning ones, they also require larger amount of data and processing power to operate, which is not necessarily compatible with specific use cases where local resources are scarce, such as mobile network edge resource management.

Centralizing the closed loop in an area with more resource may result in increased risk of disconnection from the managed resources. In many cases, such as security, such disconnection may not be acceptable.

Similarly, centralization may induce additional delays in the loop, which may also be a problem if the action plan needs to be carried out in a timely manner.