**3GPP TSG-SA5 Meeting #162 *S5-253332***

Gothenburg, Sweden, 25 – 29 August 2025

**Source: Nokia, Samsung**

**Title: Pseudo-CR on Rapporteur cleanup and corrections**

**Document for: Approval**

**Agenda item: 6.19.4.1**

**Spec: 3GPP TS28.567**

**Version: 0.2.0**

**Work Item: Closed Control Loop Management**

**Comments**

This pCR makes correction and cleanup of the draft TS.

**Proposed Changes**

\* \* \* First Change \* \* \* \*

# 1 Scope

The present document describes, concepts and background, and specifies use cases and requirements, information models and procedures for use, control, conflict management and coordination of Closed control loops in network management.

# 2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication, edition number, version number, etc.) or non‑specific.

- For a specific reference, subsequent revisions do not apply.

- For a non-specific reference, the latest version applies. In the case of a reference to a 3GPP document (including a GSM document), a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

[1] 3GPP TR 21.905: "Vocabulary for 3GPP Specifications".

[2] 3GPP TS 28.535: "Management and orchestration; Management services for communication service assurance; Requirements".

[3] 3GPP TS 28.536: "Management and orchestration; Management services for communication service assurance; Stage 2 and stage 3".

[4] 3GPP TS 28.111: "Technical Specification Group Services and System Aspects; Management and orchestration; Fault Management (FM)".

[5] 3GPP TS 28.622: "Technical Specification Group Services and System Aspects; Telecommunication management; Generic Network Resource Model (NRM) Integration Reference Point (IRP); Information Service (IS)".

[6] 3GPP TS 28.572: "Management and orchestration; Management of planned configurations".

# 3 Definitions of terms, symbols and abbreviations

## 3.1 Terms

For the purposes of the present document, the terms given in TR 21.905 [1] and the following apply. A term defined in the present document takes precedence over the definition of the same term, if any, in TR 21.905 [1].

**Closed Control Loop**: A management function that monitors and controls a set of managed entities, and operates without any intervention from a human operator or any other management entity other than possibly the initial configuration.

## 3.2 Symbols

Void.

## 3.3 Abbreviations

For the purposes of the present document, the abbreviations given in TR 21.905 [1] and the following apply. An abbreviation defined in the present document takes precedence over the definition of the same abbreviation, if any, in TR 21.905 [1].

CCL Closed Control Loop

# 4 Concepts and overview

## 4.1 Closed Control Loops

A Closed Control Loop (CCL) is a type of control mechanism that monitors and regulates a set of managed entities with the objective of achieving a specific goal. A CCL can be logically decomposed into several stages or steps, each providing a specific functionality and where the steps work together to achieve the stated goal. Any two CCLs may have the same functionality but supported in different count of steps, i.e. the steps implement differing functional capabilities although together, each set achieves the same functionality. Similarly, any two CCLs with the same functionality and same count of steps, the respective steps may not have the same functionality.

A control loop is a building block for management of networks and services. The basic principle of any control loop is to adjust the value of an observed variable (expressed as for example an attribute) to control/influence the value of a desired goal (expressed as for example an attribute) for a controlled entity, such as a managed entity or managed function. The producers of the measurements or observations services, analysis services and control service, are all required to fully realize and use a control loop.

A Closed Control Loop (CCL) is a control loop which operates without any intervention from a human operator or any other management entity other than possibly the initial configuration of the measurement producer and configuration of the control loop. In a closed control loop the input to the control loop provided by human operator or other management entity may include the goal or policies. Besides the provisioning needed to realize the goal, the output of the closed control loop may also include closed control loop status to a human operator or other management entity.

Examples of well-known Closed Loop types are OODA loop, composed of 4 stages/steps (Observe, Orient, Decide, Act) and MAPE, also composed of 4 stages/steps (Monitor, Analyse, Plan, Execute).

1. b)

Figure 4.1-1: Open control loop entities versus Closed control loop entities (see TS28.535[2])

## 4.2 Functional steps of a closed control loop

A closed control loop may manage any managed entity, e.g. a network resource or a communication service as described in TS 28.535 [2] and TS 28.536 [3]. Generally, the control loop consists of the steps Monitoring/data collection, Analysis, Decision and Execution. The adjustment of the resources of the managed entity used is completed by the continuous iteration of the steps in a management control loop.



Figure 4.2-1: Steps of a Control Loop  
a) the four functional steps; and  
b) 2 steps combined into a single management function

- The "Monitoring/data collection" step is responsible for collecting and pre-processing data from managed entities or from external sources.

- The "Analysis" step derives insights from the available data obtained in the monitoring/data collection step. The insights provide answers to the questions, for example What is likely to happen, what has happened and why.

- The "Decision" step is responsible for deriving workflows from insights provided by the analysis step. It decides which reactive, proactive or predictive actions should be taken in consideration of insights obtained in the analysis step.

- The execution step manages the activation of commands on the controlled resources or entities. The decision step should decide which actions are required, but not necessarily how they should be taken in the managed entities. So, the translation from actions to commands is a responsibility of the execution step.

## 4.3 Characteristic information of a CCL

### 4.3.1 Overview

A CCL is associated with a set of Characteristic information that describes its properties, behaviours and impact. This information includes CCL Goals, CCL Triggers, CCL actions and action plans as well as CCL scopes.

### 4.3.2 CCL Goals and targets

The CCL is responsible for a set of outcomes that need to be achieved or realized by the CCL. Each expected outcome is called a CCL target. A CCL may have one more CCL goals each containing a set of CCL targets.

### 4.3.3 CCL actions and actions plans

A CCL action is a configuration change that a CCL can perform over a managed entity such as configuring an attribute of managed object. A CCL may decide to perform several actions which can be combined in a single CCL action plan.

### 4.3.4 CCL scopes

The scope is the set of managed objects, their properties and network outcomes that are associated with the CCL for measurement, configuration and impact. The scopes for the different CCLs can be managed by the MnS consumer, i.e. they can be defined on to the CCL or revised by the MnS consumer. A CCL may have four scopes: the measurement scope, target (impact) scope, control scope and impact scope, defined as follows:

- measurement scope: the measurement scope is where related measurements are collected- control scope: control scope is the scope to which the CCL's actions are desired to be applied, e.g., the set of network functions and attributes that are the planned candidates to be modified by the CCL. The control scope is also called the action-space as it describes the set of candidate actions that the CCL can (is configured to be able to) execute.

- target scope: which relates to purpose of the CCL

- desired impact scope: the scope to which the CCL's actions are desired to have influence, e.g., it is both the network functions and attributes as well network outcomes like coverage areas that are planned to be influenced by the configuration’s actions of the CCL.

- impact scope: impact scope is the scope to which the CCL's actions have influence, e.g., it is both the network functions and attributes as well network outcomes like coverage areas that are influenced by the configuration actions of the CCL. This is different from the measurement scope, i.e. the scope where the CCLs measure and control scope, i.e. the scope where they act.

The impact scope may be known and bounded or unbounded and thus unknown - see figure 4.3.4-1. The bounded scope indicates that the area known by the CCL is the scope where its actions will impact. The unbounded impact-scope is the full network scope where the CCL’s action will have impact, but the CCL does have information that its action will have that impact to that scope.



Figure 4.3.4-1: Exemplification of known/bounded vs. unknown/unbounded impact scope: CCL A takes action in cell A expecting impact in cells A, B, C and D. if the impact is strictly in cells A, B, C and D, then the impact scope is known and bounded. However, if the impact scope includes cells E and F, then for the CCL, the true impact scope is unknown and thus unbounded.

## 4.4 Closed control loops purposes and uses

CCLs automate the management of network resources thereby taking control away from operators. TS 28.535 [2], TS 28.536 [3] describe the use of CCLs for service assurance. Additionally, CCLs may be used for several other use cases including among others use of CCLs for problem analysis and for fault management.

In each case, the CCL can be viewed as an entity to be managed, management capabilities related to the CCL will be exposed by the MnS producer that is associated with the CCL to enable the MnS consumer to manage the CCL. E.g., the characteristics and behaviours of the CCLs need to be directed by operators as consumers of CCL-related management services. Moreover, the MnS also exposes capabilities for coordinating the CCL's activities. Example capabilities include capability for providing information on conflict resolution and feedback on monitoring the impact of the CCL's actions on other closed control loops or management functions.

The 3GPP management system should provide capabilities that enable a consumer to:

- Manage the execution of CCLs. E.g. to request for and be notified about the instantiation of CCLs. For instance, if the consumer wants to request for instantiation of an Energy saving CCL for 10,000 cells.

- Compose or request for and be notified about the composition of a CCL from a set of specific components (such as analytics services or SON functions).

- Manage a closed loop composed from multiple components.

## 4.5 Realisation of closed control loops

The CCL may be composed on demand into what is called an open-box CCL from discrete entities in the Management system that accomplish the functional steps, e.g., using capabilities offered by the Management system. The CCLs steps (or the components accomplishing them) as well as the interactions among them, are based on 3GPP management services. E.g., the CCL may apply one or more PM jobs for the Monitoring/data collection step and apply an MDA capability for an analysis stage.

However, the CCL (e.g., in any form in Figure 4.2-1) can be pre-integrated into what are called Closed-Box CCLs. where, the CCLs components and their interactions are assembled prior to instantiating the CCL in the Management system. They are not accessible to the MnS consumers, only the CCL’s external interactions and capabilities, e.g., the control interface to define the CCL scopes is accessible to the MnS consumers.

## 4.6 Closed Control Loops conflicts

Multiple CCLs could co-exist and concurrently act within the same environment. The CCLs can affect one another, in the worst cases leading to conflicts. The conflicts may occur among desired outcomes, scopes or actions of the CCLs. The possible conflict scenarios include:

- CCL outcomes conflicts

- CCL Scope conflicts

- CCL-Trigger-time:

- CCL actions conflicts, with 2 subtypes – concurrent and non-concurrent actions conflicts.

- CCL metric-value conflicts: with 2 subtypes – concurrent and non-concurrent metric-value conflicts.

# 5 Management capabilities

## 5.1 Dynamic control and composition of CCLs - DynCCL

### 5.1.1 Description

CCLs may be dynamically realized. There are two aspects to dynamically realization of CCLs - dynamic instantiation of a CCL from an existing template and dynamically composing the CCL from discrete components based e.g. on the provided goals.

### 5.1.2 Use cases

#### 5.1.2.1 General CCL Control – DynCCL\_01

A CCL contains a set of logic functionalities or steps, each providing a specific functionality and where the steps work together to achieve the stated desired outcomes over a given network scope. The MnS consumer should be able to configure and receive information about the desired outcomes of the CCL.

Generally, the four CCL steps of Monitoring, Analysis, Decision and Execution are expected with the expectation that each step is accomplished by a single management function or service. However, one management function or service may also accomplish the functionality of more than 1 step. The MnS consumer should be able to receive information about the management functions or services that form the CCL.

A CCL may have four scopes including a desired outcomes scope, measurement scope, a control scope and an impact scope. The scopes for the different CCLs can be managed by the MnS consumer. The MnS consumer should also be able to receive reports about these different aspects of the CCL, e.g., about the status of the CCLs execution as well as to configure the reporting.

#### 5.1.2.2 Composing a CCL from discrete components – DynCCL\_02

A CCL may be composed from steps provided by different management functions or management services. i.e. the CCLs is assembled on demand by MnS consumers, using capabilities offered by the Management system, e.g. from independent management functions. The CCLs components, as well as the communication and interoperation between components, are based the different 3GPP management services. Accordingly, the MnS consumer should be able to identify and indicate the MnFs or MnS producers that should be used to compose a CCL. Moreover, the MnS consumer may indicate towards the MnS producer the request to compose the CL of a particular type (e.g. for optimizing energy efficiency) without requiring to state the specific components that should be used.

Two approaches are possible:

- Composition from management Functions: Different management functions may be used to realize the different steps of a closed loop, for example, an MDA function may realize the analytics step of the CCL while another management function may realize the decision step of the CCL.



Figure 5.1.2.2-1: Management functions as steps of a closed control loop

- Composition from management services: Different management services may be used to realize the different steps of a closed loop, i.e. the management service provides the output expected from a specific step.

EXAMPLE: A capability of the MDA MnS realizes an analytics step of the CCL while another capability may realize a specific data collection step of the CCL.

 b)

Figure 5.1.2.2-2: Management services used as implementations of CCL steps:  
a) MDA MnS and PM job the respective implementations of the analysis and data collection steps; and b) MDA MnS as the implementation of the decision step

The MnS consumer should be enabled to control the composition of such a CCL. The MnS consumer could request for and be notified about the composition of a CCL from a set of specific components (i.e. specific management functions or management services). The request could indicate components with specific given capabilities (such as analytics services with specific analytics types) which should be combined to achieve the closed loop. Moreover, the request could be for composition of a CCL required to achieve a specific set of desired outcomes or goals.

### 5.1.3 Requirements

Table 5.1.3-1

|  |  |  |
| --- | --- | --- |
| Requirement label | Description | Related use case(s)/ Motivation |
| REQ-DynCCL\_01-01 | The CCL MnS Producer should support a capability to provide information to the MnS consumer about the management functions and services that make up the CCL and where applicable the functionality accomplished by the components. | UC-DynCCL\_01  Clause 5.1.2.1 |
| REQ-DynCCL\_01-02 | The CCL MnS Producer should support a capability enabling the MnS consumer to configure and receive information on status of execution of the CCL. | UC-DynCCL\_01  Clause 5.1.2.1 |
| REQ-DynCCL\_01-03 | The CCL MnS Producer should support a capability enabling the MnS consumer to configure and receive information on the scopes of the CCL | UC-DynCCL\_01  Clause 5.1.2.1 |
| REQ-DynCCL\_01-04 | The CCL MnS Producer should support a capability to report to the MnS consumer about the CCL | UC-DynCCL\_01  Clause 5.1.2.1 |
| REQ-DynCCL\_02-01 | The CCL MnS Producer should support a capability enabling the MnS consumer to request for a CCL (instance) to be composed from a set of management function types or instances or management services | UC-DynCCL\_02  Clause 5.1.2.2 |
| REQ-DynCCL\_02-01 | The MnS producer for CCL management should support a capability enabling the MnS consumer to request that a CCL of a specific type or fulfilling a stated goal should be composed from a set of management function types or instances or services | UC-DynCCL\_02  Clause 5.1.2.2 |
| REQ-DynCCL\_02-01 | The MnS producer for CCL management should support a capability enabling the MnS consumer to provide conditions under which a CCL can be dynamically composed or instantiated triggered to execute | UC-DynCCL\_02  Clause 5.1.2.2 |
| REQ-DynCCL\_02-01 | The MnS producer for CCL management should support a capability enabling the MnS consumer to be notified when a CCL is dynamically composed or instantiated or triggered to execute | UC-DynCCL\_02  Clause 5.1.2.2 |

## 5.2 Historical CCL Capability - HISCCL

### 5.2.1 Description

TBD

### 5.2.2 Use Cases

#### 5.2.2.1 CCL creation based on Historical CCL data capability – HISCCL-01

This use case describes the need of maintaining information about the CCLs that existed in the past. Those CCLs are called Historical CCLs.

In an automation environment, before a consumer request to create a CCL it would like to know the data related with Historical CCLs that were available with the producer. This information will enable consumer to request for an optimal CCL. The information about historical CCL may include, scope of the CCL, configured goals/targets, controlled entity, etc.

Further, Historical CCL information serves as a valuable data source for predictive analytics within the CCL system executed as Analytics step. It enables the system to move from a reactive mode, where it responds to current issues, to a proactive mode, where it anticipates and prevents problems based on historical trends and patterns. This proactive approach enhances network reliability, minimizes downtime, and improves the overall efficiency of network operations.

The Historical CCL information may be used by the management system to setup or initialize a CCL. The Historical CCL information provides the profiles of a CCL for CCL at different hierarchies. For example, CCLs that do not do coordination which are at a lower hierarchy L and CCLs responsible for coordination (as coordination entities) which are at a higher hierarchy H. For a new CCL at a lower hierarchy, the management system obtains the profiles of the several CCLs at different hierarchies and correlates the information of the new CCL (e.g. its goal information) against the profiles of the CCLs at the different hierarchies. Based on this, the management system computes the complete profile of the new CCL (including e.g. its measurement and control scope) which is then configured onto the new CCL.

### 5.2.3 Requirements

Table 5.2.3-1

| Requirement label | Description | Related use case(s)/ Motivation |
| --- | --- | --- |
| REQ-HISCCL\_01-01 | The 3GPP management system should enable authorized MnS consumer to request for information (e.g. CCL identification, configured goals/targets and the related status, scope of the CCL, conflict information) related with Historical CCL. | UC-HISCCL\_01  Clause 5.2.2.1 |
| REQ-HISCCL\_01-02 | The 3GPP management system shall have the capability to configure the profile of a CCL based on the historical CCL information that describes the profile of other CCLs at different hierarchies. | UC-HISCCL\_01  Clause 5.2.2.1 |

## 5.3 CCL Performance Monitoring - CCLPERF

### 5.3.1 Description

TBD

### 5.3.2 Use Cases

#### 5.3.2.1 Performance Evaluation of a Closed Control Loop – CCLPERF\_01

The advanced monitoring functionalities of a CCL can provide real-time insights into the performance and outcomes of a CCL. The monitoring activity for a Closed Control Loop may result in further actions that happen in the operation phase, e.g., evaluate and update, in order to change the closed control loop settings and improve its performance. So, there is a need to evaluate the performance of a Closed Control Loop itself. Such metrics are important to understand and change a CCL's behaviour and to improve its performance to pursue the assigned goal(s).

For example, certain performance aspects of a CCL can be very crucial to know in order to evaluate and decide upon a CCL's performance, such as the number of breached goals, time taken to meet a breached goal, number of conflicts occurred by a CCL etc. With the knowledge of such performance aspects of an existing CCL, a MnS consumer can more effectively update or create a new CCL.

An operator can also compare different CCLs based on these performances and choose the best one for its network deployment.

#### 5.3.2.2 MnS Consumer’s feedback on CCL actions – CCLPERF\_02

A CCL should derive its actions without the involvement of any other entity (such as the managed network object) but the actions can have different levels of satisfaction for the different MnS consumers. The MnS consumers should be able to provide feedback to the CCL indicating how satisfied the MnS consumer is with the quality of the CCL actions, which should enable the CCL to fine‑tune and optimize its decisions.

EXAMPLE: The MnS consumer feedback may grade the usefulness of the executed action on a fixed scale say from 0 (indicating a terrible and never to be re-used action) to 10 (indicating a very good action for the interests of the MnS consumer). Other criteria may be added, e.g., to address the case that two consumer experience the same outcomes but may have different grade for feedback,

To be able to gauge the satisfaction, the MnS consumer should be able to receive information about the provisioning operations executed by the CCL. This information includes operation performed, MOIs updated, etc. The CCL does not break its execution when it provides information to the MnS consumer or to wait for feedback from the MnS consumer. The feedback from an MnS consumer does not break the loop.

It may be needed to determine what impact the CCLs’ action(s) had on a given scope that is the responsibility of other CCLs. Based on the CCL actions and the resulting impact on PMs, it may be determined that new actions are needed to undo the degradation and to avoid it in future.

Based on some local policies or due to degradations observed, the consumer may prefer that a particular NF is not updated as part of the Execution step of CCL. The consumer should be enabled to request the CCL to revoke the changes made to a NF. Consumer may also update the CCL to ensure that a particular NF is never updated in future.

#### 5.3.2.3 Assessment and resolution of CCL Impact on unknown impact-scope - CCLPERF\_03

For some CCLs, the impact-scope affected by the actions of a CCL A may not be known a priori. For example, when a CCL A adjusts transmit power (e.g. to minimize interference), the neighbour cells and related CCLs acting on those cells that would be affected by any transmit power decrease or increase cannot be explicitly enumerated. Any negative effects cannot be easily anticipated, and most may not be easy to resolve by if‑then-else rules. Instead, the affected CCLs should report their observed negative or positive impacts to CCL A to determine how to resolve the impact or avoid them in future.

'Related CCLs need to be notified that CCL A has executed an action and the impact-time of the action, i.e. the maximum time within which the action is expected to have impact and at which an observed impacts should be reported. For example, the impact of load balancing is visible in a few seconds while the impact of a handover decision can take several minutes. After the notified impact time, the impacted CCLs need to report the impact that CCL A had to their performance metrics . The impact may be reported an index say in the range [0,10] where 0 implies an unacceptable action and 10 implies a good action. CCL A can then derive an appropriate remediation, e.g. by reconfiguring the candidate actions of the acting CCL (i.e. CCL A) or by undoing the action.

### 5.3.3 Requirements

Table 5.3.3-1

| Requirement label | Description | Related use case(s)/ Motivation |
| --- | --- | --- |
| REQ-CCLPERF\_01 -01 | The 3GPP management system should be able to obtain a CCL's performance with respect to the total number of occurrences of a goal breach. | UC-CCLPERF\_01  Clause 5.3.2.1 |
| REQ-CCLPERF\_01-02 | The 3GPP management system should be able to obtain a CCL's performance with respect to the time taken by CCL to meet a breached goal. | UC-CCLPERF\_01  Clause 5.3.2.1 |
| REQ-CCLPERF\_01-03 | The 3GPP management system should be able to obtain a CCL's performance with respect to the total number of conflicts occurred by a CCL | UC-CCLPERF\_01  Clause 5.3.2.1 |
| REQ-CCLPERF\_02-01 | The 3GPP management system should enable MnS consumer to provide its feedback on the action(s) taken by CCL. | UC-CCLPERF\_01  Clause 5.3.2.2 |
| REQ- CCLPERF\_02-02 | The 3GPP management system should enable MnS consumer to request for revocation of the action(s) taken by the CCL. | UC-CCLPERF\_01  Clause 5.3.2.2 |
| REQ-CCLPERF\_02-03 | The 3GPP management system should have a capability enabling the MnS consumer to receive information (e.g. operation performed, MOIs updated) about the action(s) taken by a CCL A. | UC-CCLPERF\_01  Clause 5.3.2.1  UC-CCLPERF\_02  Clause 5.3.2.2 |
| REQ-CCLPERF\_03-01 | The 3GPP management system should support a capability enabling an MnS consumer to receive a report containing an executed action and the impact that the action had to a particular impact-scope. | UC-CCLPERF\_02  Clause 5.3.2.2  UC-CCLPERF\_03 Clause 5.3.2.3 |
| REQ-CCLPERF\_03-02 | The 3GPP management system should support a capability enabling an MnS consumer to propose to a CCL a remediation against the noted impact of a CCLs’ actions, e.g. the reconfiguration of the candidate actions of the CCL. | UC-CCLPERF\_03 Clause 5.3.2.3 |

## 5.4 Closed Control Loops usage scenarios - CCLUSE

### 5.4.1 Description

Closed control loops can be used for different purposes or scenarios Two example scenarios are fault management and network performance problem recovery.

### 5.4.2 Use Cases

#### 5.4.2.1 Closed Control Loops for fault management – CCLUSE\_01

This use case describes a scenario in which an MnS consumer may request a CCL for fault management. The consumer may request to identify the root cause of the fault and take actions to mitigate and/or resolve the root cause for a given list of alarms. Furthermore, the request may include policies and actions specified by an MnS consumer in order to mitigate and/or resolve the root causes for the given alarms.

Based on the request, a CCL may take action to further enhance the correlation of alarms, for example, correlation of alarms with change in PM/KPIs and/or fault supervision events to find solutions to mitigate and/or resolve the identified root causes. In addition, a fault management CCL may clear the alarms that otherwise have to be manually cleared by the MnS consumer, which are defined as ADMC Alarms in TS 28.111.

The MnS producer reports the result of fault management. The report may include information regarding the status of each alarm, including any identified root cause and correlation information, which may indicate the successful mitigation and/or resolving of root causes for any given alarm.

#### 5.4.2.2 Closed Control Loops for network performance problem recovery CCLUSE\_02

Based on the concept in 3GPP TS 28.104 [3], MDA reports may contain root cause analysis of ongoing issues, predictions of potential issues and corresponding relevant causes and recommended actions for preventions, and/or prediction of network and/or service demands. For example:

- MDA for Coverage problem analysis can provide the following information in the MDA report:

- coverageProblemId;

- coverageProblemType;

- coverageProblemAreas; and

- recommendedActions.

The MnS consumer may decide to use CCLs to resolve the observed performance problems based on the analytics reports (e.g. provided by MDA) and other management data (e.g. historical decisions made previously) if necessary. It can be possible that one MnF is responsible for network performance problem observation and recovery, while another MnF is responsible for the decision on whether the network performance problem needs to be resolved. In this scenario, The MnF for decision can decide whether it needs the other MnF to recover from the observed network performance problems (e.g. coverage problem) based on MDA report (e.g. root cause information, recommended solutions) and other information (e.g. user experience information, information from other domains). If it decides to recover the observed network performance problems, The MnF for decision needs to request another MnF to recover the specified network performance problems observed from the MDA report. MnS consumer may specifies the network scope and time window for network performance problem recovery, which means the MnS producer needs to recover the problem at the specified time window for the network scope. During problem recovery phase, as process for network performance problem recovery is complex and time-consuming, the MnS consumer needs to obtain the progress of the recovery process. When the last step of the network performance problem process is completed, MnS producer needs to send the result of this network performance problem recovery process to the MnS consumers.

If a closed control loop instance can be used to resolve network performance problem, the MnS consumer may need to know the result of resolving the network performance problem by the closed control loop instance, including the network performance problems which are resolved by the closed control loop as well as network performance problem resolution statistics (e.g. the number of network problem resolved by the closed control loop in the specified period).

### 5.4.3 Requirements

Table 5.4.3-1

|  |  |  |
| --- | --- | --- |
| Requirement label | Description | Related use case(s)/ Motivation |
| REQ-CCLUSE\_01-01 | The 3GPP management system shall have the capability to allow MnS consumer to request a closed control loop for fault management | UC-CCLUSE\_01 |
| REQ-CCLUSE\_01-02 | The 3GPP management system shall have the capability to allow MnS consumer to get a report from the closed control loop regarding the status of fault management | UC-CCLUSE\_01 |
| REQ-CCLUSE\_02-01 | The 3GPP management system should have the capability to allow the MnS consumer to request a CCL for resolving the network performance problems. | UC-CCLUSE\_02  Closed Control Loops for network performance problem recovery |
| REQ-CCLUSE\_02-02 | The 3GPP management system should have the capability to allow the MnS consumer to obtain the result of network performance problem resolved by the closed control loop. | UC-CCLUSE\_02  Closed Control Loops for network performance problem recovery |
| REQ-CCLUSE\_02-03 | The 3GPP management system should have the capability to allow the MnS consumer to obtain the progress information of network performance problem recovery. | UC-CCLUSE\_02  Closed Control Loops for network performance problem recovery |
| REQ-CCLUSE\_02-04 | The 3GPP management system should have the capability to allow the MnS consumer to configure a CCL with the network scope and time window to be monitored for i resolving the network performance problems. | UC-CCLUSE\_02  Closed Control Loops for network performance problem recovery |

## 5.6 Triggered CCL Capability - CCLTRG

### 5.6.1 Description

### 5.6.2 Use Cases

#### 5.6.2.1 CCL instantiation based on conditions – CCLTRG\_01

The MnS consumer may want to request for a CCL to be instantiated not immediately but when certain conditions are met. For example, the MnS consumer may want that for a CCL of a stated type or that matches a set of stated characteristics (e.g. goal) to be instantiated under certain conditions and another with variations in goals to be instantiated under other conditions. The MnS consumer should be enabled to define those conditions so that the CCL is instantiated when the stated conditions are met. The MnS Producer monitors the conditions to check if they are met.

The conditions can be related to events based on management data (e.g., performance, fault, configuration).

Performance events are defined related with performance measurements and KPIs that need to be monitored by the producer to see if an CCL is to be initiated. For example, if the value of a particular performance measurement goes beyond a particular value, a CCL should be instantiated to keep the value of the same performance measurement below a defined value.

Fault events are defined by related information (e.g., alarm type, alarm severity) that need to be monitored by the producer to see if a CCL is to be initiated. For example, if the total number of alarm with type QUALITY\_OF\_SERVICE\_ALARM and perceivedSeverity MAJOR goes beyond a particular value, a CCL should be instantiated.

Provisioning events (e.g., CreateMOI) are defined that need to be subscribed by the producer to see if a CCL is to be instantiated. For example, the creation on an Intent MOI can be a trigger to instantiate a CCL. For another example, when a pre-defined system time specified by operators can be a trigger to instantiated a CCL.

NOTE: The use case requires to set the conditions and then the conditions need to be continuously monitored and tracked.

#### 5.6.2.2 CCL action execution based on conditions – CCLTRG\_02

For the CCLs that have been instantiated, the MnS consumer may want to define conditions under which a CCL may execute actions on the network, e.g. when the performance on a certain threshold is crossed, or when the confidence on the decision is above a stated threshold. The consumer does not need to be aware of all decisions, but by providing conditions under which decisions may be activated or not, it is able to have supervision over the CCL without having to continuously track the decisions. The MnS consumer should be enabled to define those conditions for executing the CCL actions. Otherwise, the consumer should be enabled to define alternative actions, e.g. to notify the consumer of the decision that is not executed.

By supporting this, the execution can be affected by producer based on consumer's conditions or requirements. To ensure oversight or accountability by the MnS consumer, the MnS consumer may be notified by the CCL about the executed action for any conditionally execution. The MnS consumer can intervene as needed.

NOTE: The use case requires to set the conditions and then the conditions need to be continuously monitored and tracked.

### 5.6.3 Requirements

Table 5.6.3-1

| Requirement label | Description | Related use case(s) |
| --- | --- | --- |
| REQ-CCLTRG\_01-01 | The 3GPP management system should enable authorized consumers to define conditions related to performance, fault and configuration data that can be monitored and used to trigger CCL instantiation. | UC–CCLTRG\_01  Conditional trigger of a CCL |
| REQ-CCLTRG\_01-02 | The 3GPP management system should enable authorized consumers to define conditions related to performance, fault and configuration data that can be monitored and used to trigger CCL update. | UC–CCLTRG\_01  Conditional trigger of a CCL |
| REQ-CCLTRG\_01-03 | The 3GPP management system should enable authorized consumers to define conditions related to performance, fault and configuration data that can be monitored and used to trigger CCL deletion. | UC–CCLTRG\_01  Conditional trigger of a CCL |
| REQ-CCLTRG\_02-01 | The 3GPP management system should enable authorized consumers to define conditions related to performance, fault and configuration data under which a CCL may execute its actions. | UC–CCLTRG\_02  Conditional execution of CCLs network changes |

## 5.7 CCL Conflict management Capability - CONF

### 5.7.1 Description

#### 5.7.1.1 Overview

A CCL may experience direct conflicts on its goals, targets, scopes, trigger time and execution time. The management system needs to support capabilities to avoid, detect and resolve the conflicts.

The possible conflict scenarios are defined as follows:

- **CCL Scope conflicts:** These are conflicts among the scopes of the CCLs, specifically the scenarios where a given scope is considered differently by distinct CCL instances. An example is where the measurement scope of one CCL is the control scope of another CCL. Where applicable, it is desirable that the scopes are allocated such that that one CCL instance does not read a scope that is concurrently being controlled or adjusted by another CCL. These also include conflict among the desired outcomes of the individual CCLs sharing a given scope.

- **CCL actions conflicts:** These are conflicts among the actions of the CCLs, specifically the scenarios where two CCL instances attempt to differently control the same parameters of the same managed objects. Where applicable, it is desirable that the actions are decided and allowed such that that two CCL instances will not control or adjust the same set of parameters on the same set of managed objects.

There are 2 subtypes of CCL actions conflicts – concurrent and non-concurrent actions conflicts.

- **CCL concurrent actions conflicts:** These are conflicts where the actions are executed within a time period less than the impact time of the action, i.e., the action of the second CCL instance is executed before the impact of the first CCL instance is registered. In the simplest scenario, the two CCL instances try to execute the contradictory actions at exactly the same time. Concurrent actions conflicts are also called “action-execution-time conflicts”

- **CCL non-concurrent actions conflicts:** These are conflicts where the actions are executed within a time period longer than the impact time of the action, i.e., the action of the second CCL instance is executed after the impact of the first CCL instance is registered. The second CCL instance in effect tries to undo the impact of the CCL instance.

- **CCL metric-value conflicts:** These are conflicts for the desired value of one or more performance metrics by two CCL instances that do not have conflicts for desired outcomes on stated scopes or actions. The two CCL instances which have different desired outcome and two distinct control and measurement scopes but the actions of one CCL instance have impact on the measurement scope of the other CCL instance, i.e. one CCL’s actions will indirectly affect the network performance metrics that the other CCL is responsible for. For example, a conflict could occur among the metrics if a CCL that optimizes energy consumption affects handover performance metrics which are supposed to be optimized by another CCL.

There are 2 subtypes of CCL metric-value conflicts – concurrent and non-concurrent metric-value conflicts.

- **CCL concurrent metric-value conflicts**: These are metric-values conflicts between CCLs with close trigger times, i.e., where the CCL instances are triggered to act concurrently or to execute actions within the same time.

- **CCL non-concurrent metric-value conflicts**: These are conflicts where the CCL instances are triggered to act in different time periods, e.g. where one CCL instance is active while the other is only monitoring its measurement scope.

#### 5.7.1.2 Example conflicts

Examples characterizing the differences among the conflicts are summarized by Table 5.7.1-1.

Table 5.7.1-1: Types of potential conflicts among CCL instances for desired outcome g1, g2 and g3

| Conflict Type | Description | CCL-A | CCL-B | Comments |
| --- | --- | --- | --- | --- |
| Scope conflict | For CCLs CCL-A and CCL-B, CCL-A and CCL-B have different desired outcomes and actions but their scopes are overlapping - e.g. CCL-A's control scope (i.e. the controlled entities in the network) is part of CCL-B's measurement scope (i.e. the measured entities in the network). | Measurement scope:  - cells g1  Control Scope:  - g1  Desired outcome:  - EC/bit is < 1WA  Actions:  - Entity: gNB-g1  - Change: switch off g1 | Measurement scope: cells g1, g2, g3, g4  Control Scope:  - g2  Desired outcomes:  - Load < 80 %  Actions:  - Entity: gNB-g2  - Change: change CIO | By switching off g2, CCL-A affects the scope which CCL-B reads for its load distribution measurements |
| Action Conflict | **Concurrent direct actions conflicts:**  For CCLs CCL-A and CCL-B, when both CCL-A and CCL-B are trying to configure the same characteristics of same entity (gNB-g1) in contradiction, the actions executed within a short time period e.g. less than the impact period of their actions | expected outcomes:  - Throughput > 10 Gbps  Actions:  - Entity: gNB-g1  - Change: scale-out  - Time: 04:00 | expected outcomes:  - EC is < 10KVA  Actions:  - Entity: gNB-g1  - Change: scale-in  - Time: 04:00 | Conflict due to the time of executing the configuration actions on the same scope at the execution step |
| **Non-concurrent direct actions conflicts:**  For CCLs CCL-A and CCL-B, when both CCL-A and CCL-B is trying to configure the same characteristics of same entity (gNB‑g1) in contradiction, the actions far apart from each other; e.g. in a time period longer than the impact period of their actions | Example 1 | | Conflict due to configuration actions at execution step because both CCL want contradicting values for a particular characteristic of gNB-g1.  Effect: the value may ping-pong continuously. |
| expected outcomes:  - Throughput > 10 Gbps  Actions:  - Entity: gNB-g1  - Change: scale-out virtual resource | expected outcomes:  - EC is < 10 KVA  Actions:  - Entity: gNB‑g1  - Change: scale-in virtual resource |
| Example 2 | |
| expected outcome:  - HO failure is < 2 %  Actions:  - Entity: gNB-g1  - Change: set CIO to a small **positive** value{to guarantee HOs with low chances of HO failure} | expected outcome:  - Load < 80 %  Actions:  - Entity: gNB-g1  - Change: set CIO to a small negative value [to advance HOs and move load to other cells] |
| Metric-value conflict | **CCL concurrent metric-value** **conflicts:** For CCLs CCL-A and CCL-B, when CCL-A [optimize handover] and CCL-B [minimize interference] have different desired outcomes but are executed within a short time intervals between each other and the actions of CCL-A affect the desired outcomes of CCL-B. | expected outcome:  - HO failure is < 2 %  Actions:  - Entity: gNB-g1  - Change: reduce CIO {to reduce chances of HO failure} | expected outcome:  - SINR > 10 dB  Actions:  - Entity: gNB‑g1  - Change: lower antenna tilt | By reducing antenna tilt to minimize interference CCL-B affect the HO desired outcomes that are being optimized by CCL-A |
| **CCL non-concurrent metric-value** **conflicts:** For CCLs CCL-A and CCL-B, when CCL-A [optimize handover] and CCL-B [minimize interference] have different desired outcomes but are executed far apart from each other but the actions of CCL-A affect the desired outcomes of CCL-B. | expected outcome:  - HO failure is < 2 %  Actions:  - Entity: gNB-g1  - Change: reduce CIO {to reduce chances of HO failure} | expected outcome:  - SINR > 10 dB  Actions:  - Entity: gNB‑g1  - Change: lower antenna tilt | By reducing antenna tilt to minimize interference CCL-B affect the HO outcomes that are assumed optimal and stable by CCL-A |

The CCL may detect or observe events that identify the possibility of any one of the above conflicts. The conflict can be avoided using information or the policies (e.g. priority) provided by the consumer. The respective information is described in the use cases below. If the conflict actually occurs, the CCL MnS producer should support services to inform MnS consumers the confirmed detected conflicts. This may also include informing MnS consumer about the potential conflict.

### 5.7.2 Use Cases

#### 5.7.2.1 CCL scope conflicts handling – CONF\_01

Each CCL should have specific scopes for which it is responsible. The network may be assumed to be a muti--dimensional space, with say n dimensions, i.e., the network has full scope S of n dimensions including, e.g., time, geography, etc. A CCL is assigned a sub scope D that is only a portion of the network’s scope (illustrated by Table 5.7.2.1-1). Scope assignment is the mapping of CCLs to sub scopes S that are part of the network's full scope. A scope conflict occurs if the scope assigned to a CCL overlaps in an undesirable way with another scope assigned to another CCL. The 3GPP management system should support the capability to coordinate the scope assignment to enable detection and avoidance of potential scope conflicts. The 3GPP management system should also support the capability to coordinate the outcomes desirable for the different scopes to enable detection, avoidance and resolution of conflicts on the CCL’s outcomes for those scopes. It may be desirable to define the full scope space S and a set of scope rules to be used to derive the best scope to be assigned to each CCL. An example rule may be that the defined CCL scope should not overlap. The rules may for example be defined by an operator or can be implementation specific depending on the types of CCLs that are to be configured.

Table 5.7.2.1-1: Example of a network scope-space from which the scope of CCL may be derived

|  |  |  |
| --- | --- | --- |
| Scope dimension | Granularity | Example values to be assigned |
| Time | Seconds, minutes, days | Every hour,  Every Saturday at 2:00 hours |
| Network domains |  | Radio  Core |
| Geography | Region/City | City x  Street y in City x |
| Network Elements | gNB | gNB X |
| Cells | Cell A on gNB X |
| Terminals, e.g. types of users | users |
| Resources | Slices |  |
| Network Function | Virtual Network Function A  Physical Network Function B |
| Transport containers (links, flows, etc.) | an identifiable link,  a specific flow |
| Purpose | The purpose of the CCL | Coverage, Performance, Energy Efficiency, Fault Management, UE specific mobility |

NOTE: Table 5.7.2.1-1 is not complete and can be improved and/or extended as needed. Scope conflicts are only considered actual if the application of the defined scopes results in negative outcomes. The management system should support the capability to coordinate the scope assignment to detect and resolve actual scope conflicts. The CCLs monitor changes in their scope. If the scope is changed, it is desirable for the CCLs to notify the scope assignment MnS consumer of the changes or differences between what was configured and the actual scopes. The scope assignment MnS consumer may then trigger scope conflict evaluation based on the actual scope.

#### 5.7.2.3 CCL Concurrent actions conflicts handling - CONF\_02

Several CCLs may want to execute actions onto the network. It may not be desirable that their actions are executed within the same time frame. For example, if executed so close to one another, their effects will be super-imposed and neither CCL can identify the effect of its actions on the network.

The management system should support the capability for detection of potential concurrent actions conflicts. A coordination entity acting as a supervisory action-critic oversees the actions of the different CCLs may need to receive information enabling the detection of such conflicts. The action-critic functionality takes the responsibility for the end-to-end performance across several CCLs enabling evaluation of cases when the actions of multiple CCLs collide.

For a given CCL, the MnS consumer may need to receive the recommended changes from the CCLs, to evaluate them and see if they overlap with other proposed changes from other CCLs. Where there are likely conflicts and expected undesired impacts, the MnS consumer may propose to the CCLs, the changes that should be undertaken to minimize concurrent changes on the same network resources. The MnS consumer may need to provide feedback to the CCL instance (s) regarding their recommended actions.

In some instances, the conditions in the network may be such that it is not clear which CCL should be triggered, requiring to trigger multiple CCL in sequence. The CCLs may operate in a hierarchy with each CCL having an operational profile indicating the specific level of hierarchy. The MnS consumer that coordinates the execution times of the CCLs needs to configure the appropriate hierarchy for the CCLs. The triggering by a coordination capability based on information from the CCL allows resolution of CCL Concurrent actions conflicts.

#### 5.7.2.2 CCL trigger conflicts handling - CONF\_03

Typically, a CCL whose start is triggered based on conditions, needs to be triggered to run at a specific time and terminate when certain conditions are met, to run when a certain performance threshold is crossed. If triggered independently, there may be conflicts among the CCLs. The triggers for different CCLs to be executed need to be coordinated to avoid conflicts among the CCLs. The triggers for execution of different CCLs need to be coordinated to avoid conflicts among the CCLs.

#### 5.7.2.6 CCL non-concurrentmetric-value conflicts handling - CONF\_04

Two (or more) CCLs configuring different control parameter may all influence the same metric. If the two CL desire different values for the metric, the CCLs are in conflict for the metric resulting into a metric-value conflict. In effect the actions of the two CCLs are in conflict but indirectly since they are conflicting for the same control parameter but their impacts are conflict on the desired value of the metric or target. Such conflicts are metric-value conflicts and if their actions are far enough apart that their effects cannot be related to one another, they are non-concurrentmetric-value conflicts.

The management system should support the capability for avoidance of concurrent metric-value conflicts. Since each CCL focuses on a smaller scope of the network problem space, several CCLs may need to be executed. For actions in a given network scope, the CCLs can be explicitly scheduled by the management system. Where the scopes overlap, the CCLs need to align the action plans, for example, which action plan to execute and when. There is a need to assess each plan and choose the most appropriate combination of action plan(s) based on the selection policy and then notify the selected action plan(s) to the related CCLs. The MnS consumer may also be notified when it is safe to ignore the conflict. The MnS consumer may configure the criteria for evaluating the severity of conflicts.

For a detected metric-values conflict, the coordinator CCL can trigger one or more CCLs to respond to the detected potential conflict. If the CCLs that has been requested to resolve potential conflict is unable to resolve that conflict, the CCL should inform the CCL coordination MnS producer about the failure to resolve the problem.

#### 5.7.2.4 CCL non-concurrentactions conflicts handling –CONF\_05

When two (or more) CCLs attempt to adjust the same network parameter but with different and contradicting values, the desired actions of the 2 CCL will be in conflict. For example, a CCL assuring throughput of a slice may be scaling-out the virtual resources of the slice. Whereas a CCL minimizing the energy consumption may be scaling-in the virtual resource of the same slice. It can be when the CCLs execute actions at the same time. However, it also happens when the CCLs execute at different times, and the scenario for actions to be separated in time is the more likely than actions occurring simultaneously. casein these conflict scenarios, the network parameter continuously ping-pongs between the two values. Such a conflict may be called an action conflict.

NOTE: A potential conflict can for example be detected if a CCL observed that PMs on a certain object keep flipping between two values. The constant flipping can be an indication that 2 CCL instances are attempting to change the same scope.

The CCL may detect or observe events that identify the conflicts. The conflict can be avoided using some information or the policies (e.g. priority) provided by the consumer. If the conflict actually occurs, the CCL MnS producer should support services to inform MnS consumers the confirmed detected conflicts. It is needed to maximize the avoidance of conflict, including “requesting” information from MnS consumer and to inform MnS consumer about the potential conflict. CCL MnS Producer may also provide recommendations, for updating/deleting the conflicting CCLs, that would result in the resolution of detected conflict. The recommendation for update may include suggestions for modified targets.

Editor’s Note: The exact information that can be exchanged is FFS

#### 5.7.2.5 CCL concurrentmetric-valueconflicts handling – CONF\_06

Two (or more) CCLs configuring different control parameter may all influence the same metric. In other cases, the two CCLs influence two metrics Y1 and Y2 that are coupled, i.e., which have a logical relationship between them. E.g. handover (HO) failure and SINR are coupled since a bad SINR can lead to more HO failures. If the two CLs desire different values for the metric, or different values for two target metrics Y1 and Y2 but the targets are coupled, the CCLs are in conflict for the metric resulting into a metric-value conflict. The concurrentmetric-value conflict is observed from oscillations in the metrics.

Two target metrics Y1 and Y2 may be coupled such that actions to optimize any of them lead to correlated oscillations/degradations in Y1 or Y2, e.g. Y1 ensuring "HO failure is < 2 %" and Y2 wanting "SINR > 10dB". The correlated oscillations indicate a potential conflict, but the CCLs may not see the oscillations in the metric that is not of their interest. The management system should support the capability for detecting potential metric-value conflicts. An MnS consumer may analyse the correlations to detect the potential conflict between CCL1 and CCL2. The MnS consumer should be able to inform CCL1 and CCL2 about the detected potential conflict represented by the correlated oscillations.

This severity of degradation in the performance metrics of the related CCLs could be the confirmation that a detected potential conflict is an actual harmful conflict. The management system should support the capability for detecting or confirming actual metric-value conflicts. The threshold to determine the severity may be defined by the MnS consumer (e.g. the operator) so that if the degree of degradation is higher than the threshold then it is a confirmed conflict that requires resolution.

The management system should support the capability for avoiding potential concurrent metric-valueconflicts. CCLs need to avoid large and frequent changes to network parameters which may affect network stability since they increase the probability of occurrence of conflicts. CCLs should take small smooth changes in the cases where the impact is not so clear and only make the large changes when the CCL is sure that the impact is positive. It is desirable for the CCL to notify to the MnS consumer the planned change, its claimed/predicted performance improvement and reliability/confidence in that action/decision. The MnS consumer may evaluate the claimed performance improvement and reliability/confidence to determine if the action should be allowed or not. The MnS consumer should be enabled notify the decision and possibly the failed criteria to the CCL - to either be executed or to be used to compute better decisions. Based on the inputs, the CCL may update its decision-making and repeat the decision evaluation process. If the CCL has consistently made good large-action-decisions, the MnS consumer should be enabled to inform the CCL that the CCL has consistently made good decisions and achieved its ultimate trust and that no more coordination of its decisions is needed.

The management system should support the capability for resolving detected metric-value conflict. The MnS consumer should be enabled to trigger one or more CCLs to respond to the detected potential conflict. And if the triggered CCLs is unable to resolve that conflict, the CCL should inform the MnS consumer about the failure to resolve the problem. The MnS consumer can set the thresholds for performance degradation that triggers conflict detection and resolution.

Editor’s Note 1: The criteria for accurately setting the thresholds for performance degradation is FFS.

Editor’s Note 2: The name and description of this type of conflict will be revisited.

### 5.7.3 Requirements

Table 5.7.3-1

| Requirement label | Description | Related use case(s) |
| --- | --- | --- |
| **REQ-CONF\_01-01** | The 3GPP Management System should support a capability to detect and inform an authorized MnS consumer about a potential or actual CCL scope conflicts. | **CONF-CONF\_01** |
| **REQ-CONF\_01-02** | The 3GPP Management System should support a capability to confirm a potential CCL scope conflict as an actual CCL scope conflict and inform an authorized MnS consumer about a confirmed actual CCL scope conflict. | **CONF-CONF\_01** |
| **REQ-CONF\_01-03** | The 3GPP Management System should support a capability to avoid or resolve a CCL scope conflict that has been detected | **CONF-CONF\_01** |
| **REQ-CONF\_01-04** | The 3GPP Management System should support a capability to coordinate the resolution of CCL scope conflicts among multiple CCLs | **CONF-CONF\_01** |
| **REQ- CONF\_02-01** | The 3GPP Management System should support a capability to detect and inform an authorized MnS consumer about a potential CCL concurrent actions conflict. | **CONF-CONF\_02** |
| **REQ-CONF\_02-02** | The 3GPP Management System should support a capability to confirm a potential CCL concurrent actions conflict as an actual conflict and inform an authorized MnS consumer about the confirmed actual CCL concurrent actions . | **CONF-CONF\_02** |
| **REQ-CONF\_02-03** | The 3GPP Management System should support a capability to avoid or resolve a CCL concurrent actions conflict that has been detected | **CONF-CONF\_02** |
| **REQ-CONF\_02-04** | The 3GPP Management System should support a capability enabling the MnS consumer to configure a hierarchy of a CCL | **CONF-CONF\_02** |
| **REQ-CONF\_03-01** | The 3GPP Management System should support a capability to detect and inform an authorized MnS consumer about a potential or actual CCL trigger-time conflicts. | **UC-CONF\_03** |
| **REQ-CONF\_03-02** | The 3GPP Management System should support a capability to confirm and inform an authorized MnS consumer about a detected CCL trigger-time conflict after it is confirmed. | **UC-CONF\_03** |
| **REQ-CONF\_03-03** | The 3GPP Management System should enable authorized MnS Consumer to provide information that can be used to support a capability to avoid or resolve a CCL trigger-time conflict. | **UC-CONF\_03** |
| **REQ-CONF\_04-01** | The 3GPP Management System should support a capability to detect and inform an authorized MnS consumer about a potential action conflict. | **UC-CONF\_04** |
| **REQ-CONF\_04-02** | The 3GPP Management System should support a capability to confirm and inform an authorized MnS consumer about an actual action conflict. | **UC-CONF\_04** |
| **REQ-CONF\_04-03** | The 3GPP Management System should enable authorized MnS consumers to provide information that can be used to resolve a CCL action conflict. | **UC-CONF\_04** |
| **REQ-CONF\_04-04** | The 3GPP Management System should enable authorized MnS consumers to provide information that can be used to avoid the action conflict. | **UC-CONF\_04** |
| **REQ-CONF\_04-05** | The 3GPP Management System should support a capability to coordinate the resolution of CCL action conflicts among multiple CCLs | **UC-CONF\_04** |
| **REQ-CONF\_05-01** | The 3GPP Management System should support a capability to detect and inform an authorized MnS consumer about a potential or actual CCL Metric-value conflicts. | **UC-CONF\_05** |
| **REQ-CONF\_05-02** | The 3GPP Management System should support a capability to confirm and inform an authorized MnS consumer about a detected CCL Metric-value conflict after it is confirmed. | **UC-CONF\_05** |
| **REQ-CONF\_05-03** | The 3GPP Management System should support a capability to avoid or resolve a CCL Metric-value conflict that has been detected | **UC-CONF\_05** |

## 5.8 CCL decision escalation – ESC

### 5.8.1 Description

This use case related to the capability to escalate decision making to another entity e.g. another CCL.

### 5.8.2 Use Cases

#### 5.8.2.1 Triggering CCL decision escalation – ESC\_01

Not all decisions made by CCLs in different network contexts (states, status, conditions, etc.) are equally effective. The CCL may need to inform another entity about its lack of confidence in its decision with a request to escalate its decision making to that entity. For example, a CCL for optimizing energy saving may fail to decide the sequence in which cells may be deactivated when there is a failure for some cells. The CCL may escalate the scenario to a CCL on problem recovery.

The MnS consumer should be able to configure MnS producer regarding the escalation recipient to which the decision is escalated. The degree to which the CCL can independently execute decisions or escalates them, should be configurable by the MnS consumer through a confidence threshold. The confidence threshold is an index on a fixed scale say from 0 (indicating lowest confidence) to 10 (indicating highest confidence). It could be configured based on the sensitivity of the operations under the CCLs’ control, the trust level in the decisions of the CCL and the necessity to consider a bigger picture at times. Then, based on how much confidence the CCL has in its decisions, the CCL can escalate a decision or situation to an escalation recipient (e.g. another CCL or a CCL coordination entity) which has this bigger picture (say has wider scope), can execute a different(larger) set of actions or has better capabilities, e.g. a larger and more capable ML model.

NOTE: The computation of confidence within the CCL is up to implementation as it depends on the CCL's purpose and the scenario that the CCL is addressing. The escalation recipient CCL should enable the escalator CCL to request for escalation for a given network context or state with e.g. information about the escalator CCL preferences and observed constraints when driving decisions. Based on its evaluations, the escalation recipient CCL should provide to the escalator CCL a report that holds the outcomes for a given escalation request.



Figure 5.8.2.1-1: required interactions for CCL decision escalation

### 5.8.3 Requirements

Table 5.8.3-1

|  |  |  |
| --- | --- | --- |
| Requirement label | Description | Related use case(s) |
| **REQ-ESC\_01-01** | The 3GPP management system should have a capability to enabling an authorized MnS consumer to configure a CCL with the degree of autonomy of to define when the CCL can escalate and the entity to which to escalate decision making. | **UC-ESC\_01**  **Clause 5.8.2.1** |
| **REQ-ESC\_01-02** | The 3GPP management system should have a capability to enabling an authorized MnS consumer (e.g. an escalator CCL) to request to escalate decision-making for a network context or state to an escalation recipient. | **UC-ESC\_01**  **Clause 5.8.2.1** |
| **REQ-ESC\_01-03** | The 3GPP management system should have a capability enabling an escalation recipient CCL to report to an authorized MnS consumer (e.g. an escalator CCL) the outcomes for a given escalation request | **UC-ESC\_01**  **Clause 5.8.2.1** |

## 5.9 CCL Coordination – COORD

### 5.9.1 Description

This use case related to the coordination of CCL, e.g. with other management capabilities

### 5.9.2 Use Cases

#### 5.9.2.1 Coordinating CCLs with other management functions – COORD\_01

A CCL can make and execute decisions in different network contexts and for different network functions and parameters. Yet within the network, there may be other management functions or features including MDA functions, SON functions, and AIML Functions, which also make decisions that affect the same network functions and parameters as the CCL. The operation of CCLs needs to be coordinated with the other management functions.

NOTE 1: This use-case only focuses on coordinating CCLs with other management functions for executing decisions.

For a given context, the CCL should indicate the set of network functions and corresponding parameters which it is interested in changing. Accordingly, the MnS consumer, say responsible for coordinating the CCLs with management functions may subscribe to be notified of changes on network functions and parameters. The MnS consumer should be able to inform the CCL of the latest changes to a network function or its parameter and a management entity/function (e.g. CCL, MDA, SON, AI/ML inference Function) responsible for the change to the parameter.

The CCL may want to obtain the history of previous values of the parameter. The history includes, for each previous value, the identifier of a respective management entity/function responsible for that change to the parameter. The CCL may define a favourable range of values of the parameter based on the received information on the latest change and the history of previous changes to the parameter. The CCL can calculate a new value of the parameter considering the favourable range as a constraint for the new value. The CCL needs then to update the value of the parameter of the network function to the new value.

NOTE 2: The MnS consumer may for example be the functionality that is responsible for coordinating CCL and other management functions.

### 5.9.3 Requirements

Table 5.9.3-1

|  |  |  |
| --- | --- | --- |
| Requirement label | Description | Related use case(s) |
| REQ-COORD\_01-01 | The CCL MnS producer should have a capability to indicate to an MnS consumer the set of network functions including their parameters which it is interested in changing | UC-COORD\_01 |
| REQ-COORD\_01-02 | The management system should have a capability enabling an authorized CCL instance acting as MnS consumer to receive information on the latest changes to a network function parameter and an identifier of a management entity/function including MDA Function, a SON Function or an AI/ML inference Function that responsible for the change to the parameter. | UC-COORD\_01 |
| REQ-COORD\_01-03 | The management system should have a capability enabling an authorized MnS consumer to receive the history of previous values of the parameter, including, for each previous value, the identifier of a respective management entity/function responsible for that change to the parameter. | UC-COORD\_01 |

# 6 Model

## 6.1 Imported and associated information entities

TBD

### 6.1.1 Imported information entities and local labels

TBD

### 6.1.2 Associated information entities and local labels

TBD

## 6.2 Class diagram

### 6.2.1 Relationships

A diagram of a computer program

AI-generated content may be incorrect.

Figure 6.2.1-1: Relations for common information models for CCL management

Editor’s Note: The handling of Goal, targets or objectives for the general closed control loops is FFS

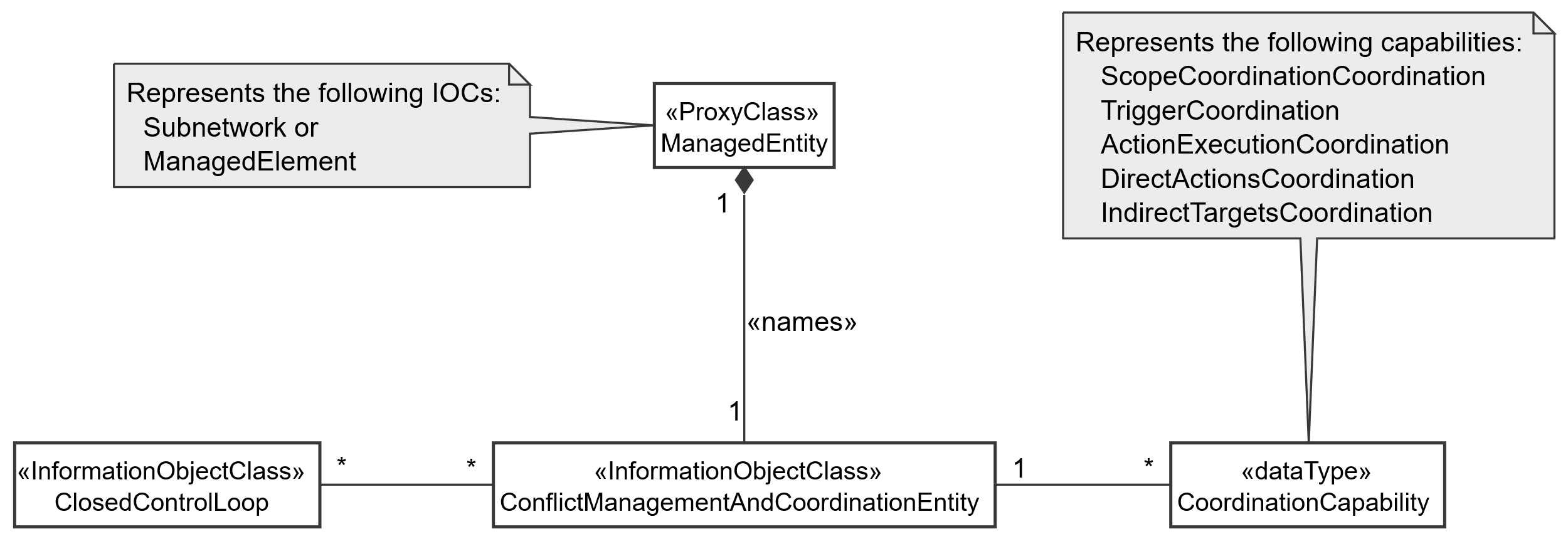


Figure 6.2.1-2: NRM fragment for conflict management and Coordination entity

### 6.2.2 Inheritance

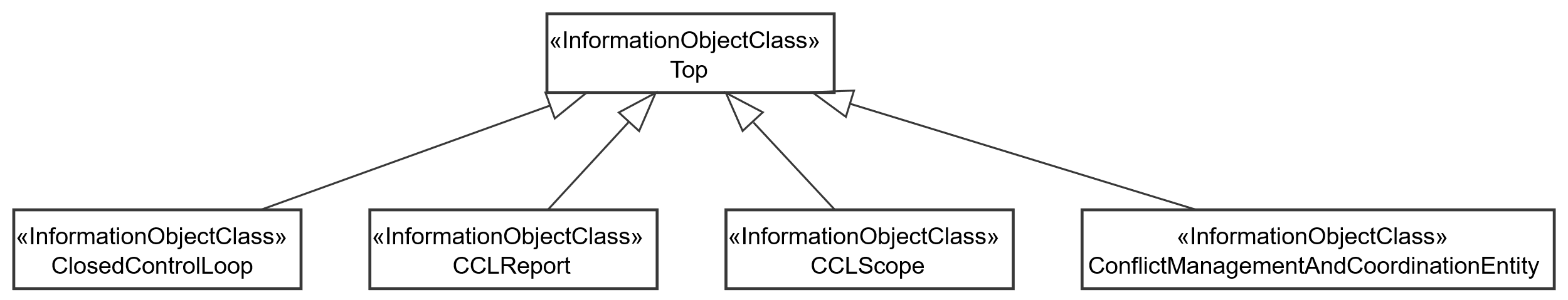


Figure 6.2.2-1: Inheritance Hierarchy for Closed Control Loops and for conflict management and Coordination entity

## 6.3 Class definitions

### 6.3.1 ClosedControlLoop

#### 6.3.1.1 Definition

This IOC represents the closed control loop. It represents the information for controlling and monitoring a CCL associated with a stated scope.

The ClosedControlLoop is name-contained by SubNetwork or ManagedElement and is associated with a CCLreport that contains reported information about the CCL. Accordingly, the report about a CCL can exist even when the CCL is deleted.

The capabilities of the CCL are contained in one or more CCLPurposes that describe what the CCL is capable of doing or can be configured to do - including information about the network resources for which the CCL can execute decisions and actions. So, the ClosedControlLoop is associated with one or more CCLPurpose(s) that indicate(s) a list of characteristics that describe what a CCL can/is expected to be able to do. The purpose describes the type of functionality that can be executed including problem recovery and fault management .

The operational information about the CCL is contained in the CCLScope(s), so the ClosedControlLoop is associated with one or more CCLScope(s). The CCLScope defines what the CCL has been configured to read, evaluate, control, etc.

A CCL can be created from several components that are dynamically composed from a set of management services, each representing one component of the CCL. The attribute cCLComponentList indicates the list of components which are combined to create a CCL.

The attribute cCLType identifies the type of CCL that needs to be composed. The specific details of the purpose that is fulfilled by the CCL are then written into the CCL purpose.

#### 6.3.1.2 Attributes

The ClosedControlLoop IOC includes attributes inherited from Top IOC (defined in TS 28.622[5]) and the following attributes:

Table 6.3.1.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| cCLComponentsInfo | O | T | T | F | T |
| operationalState | M | T | F | F | T |
| administrativeState | M | T | T | F | T |
| cCLPriority | M | T | T | F | T |
| cCLComponentList | O | T | T | T | T |
| cCLType | O | T | T | T | T |
| cCLActionTrigger | M | T | T | F | T |
| desiredBehavior | O | T | T | F | T |
| **Attribute related to role** |  |  |  |  |  |
| cCLPurposeRefList | M | T | T | T | T |

#### 6.3.1.3 Attribute constraints

None

#### 6.3.1.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.2 CCLScope

#### 6.3.2.1 Definition

It indicates a scope of a CCL. It may be the measurement scope, control scope or impact scope.

The CCLScope includes the attribute scopeType that indicates the type of scope that represented by the particular scope instance.

#### 6.3.2.2 Attributes

The CCLScope IOC includes attributes inherited from Top IOC (defined TS 28.622[5]) and the following attributes:

Table 6.3.2.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| scopeType | O | T | F | F | T |
|  |  |  |  |  |  |

#### 6.3.2.3 Attribute constraints

None.

#### 6.3.2.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.3 CCLReport

#### 6.3.3.1 Definition

This class represents the reported outcomes on a CCL instance, e.g., the information about the outcomes on one or the executing of the CCL. An CCLReport is contained by the entity containing the CCL, since the CCLReport can exist beyond the life of the CCL on which it is reporting.

There is one CCLReport per CCL for an observation time. The content of the CCLReport may be different for different observation time.

#### 6.3.3.2 Attributes

The CCLReport IOC includes attributes inherited from Top IOC (defined TS 28.622[5]) and the following attributes:

Table 6.3.3.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| FaultManagementCCLReport | CM | T | F | F | T |
| **Attributes related to role** |  |  |  |  |  |
|  |  |  |  |  |  |

#### 6.3.3.3 Attribute constraints

Table 6.3.3.3-1

|  |  |
| --- | --- |
| Name | Definition |
| FaultManagementCCLReport | Condition: fault management is supported by CCL |

#### 6.3.3.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.4 ConflictManagementAndCoordinationEntity

#### 6.3.4.1 Definition

This defines the conflict management functionality.

The IOC represents the ConflictManagementAndCoordinationEntity that is responsible for coordinating closed control loops to avoid, detect or resolve CCL conflicts.

The ConflictManagementAndCoordinationEntity is name-contained by SubNetwork or ManagedElement and is associated with one or more CCLs which the ConflictManagementAndCoordinationEntity shall be responsible for coordinating.

#### 6.3.4.2 Attributes

Table 6.3.4.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Support Qualifier | isReadable | isWritable | isInvariant | isNotifyable |
| coordinationCapability | M | T | T | F | T |
| coordinatedCCLsScopes | M | T | T | F | T |
| cCLActionConflictsHandling | M | T | T | F | T |
| **Attribute related to role** |  |  |  |  |  |
|  |  |  |  |  |  |

#### 6.3.4.3 Attribute constraints

None

#### 6.3.4.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.5 FaultManagement <<IOC>>

#### 6.3.5.1 Definition

This IOC represents the Fault Management CCL purpose, which a list of attributes that describe the capabilities of the Fault Management CCL.

#### 6.3.5.2 Attributes

**Table 6.3.5.2-1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **S** | **isReadable** | **isWritable** | **isInvariant** | **isNotifyable** |
| FaultManagementAlarmIdList | M | T | T | F | F |
| FaultManagementTimeWindow | M | T | T | F | F |
| FaultManagementBackUpObjectRequirement | O | T | T | F | F |
| FaultManagementIsolateObjectRequirement | O | T | T | F | F |
| clearUserId | CM | T | T | F | F |

#### 6.3.5.3 Attribute constraints

**Table 6.3.5.3-1**

|  |  |  |
| --- | --- | --- |
| **Name** | | **Definition** |
| clearUserId | These attributes shall be supported for Fault Management CCL that clears ADMC alarms, as specified in TS 28.111 [4]. | |

#### 6.3.5.4 Notifications

None.

### 6.3.6 CCLComponentInfo <<dataType>>

#### 6.3.6.1 Definition

This data type represents a single purpose that describes what a CCL can do. The purpose is a list of characteristics that describe the capabilities of the CCL.

#### 6.3.6.2 Attributes

Table 6.3.6.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| cCLComponentId | M | T | F | F | T |
| cCLSteps | M | T | F | F | T |

#### 6.3.6.3 Attribute constraints

None.

#### 6.3.6.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.7 CCLComponent <<dataType>>

#### 6.3.7.1 Definition

This dataType defines a CCL component that can be used or has been used to dynamically compose a closed control loop by the MnS consumer.

#### 6.3.7.2 Attributes

The CCLComponent IOC includes attributes inherited from Top IOC (defined in TS 28.622[5]) and the following attributes:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| cCLComponentRole | M | T | T | T | T |
| cCLComponentIdentification | M | T | T | F | T |

#### 6.3.7.3 Attribute constraints

None

#### 6.3.7.4 Notifications

The common notifications defined in subclause 4.1.2.5 are valid for this IOC, without exceptions or additions.

### 6.3.8 FaultManagementCCLReport <<dataType>>

#### 6.3.8.1 Definition

This data type represents the Fault Management CCL report, which is a list of attributes that describe the result of the Fault Management.

#### 6.3.8.2 Attributes

**Table 6.3.8.2-1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **S** | **isReadable** | **isWritable** | **isInvariant** | **isNotifyable** |
| GeneratedAlarmResultList | M | T | F | T | T |
| FaultManagementCCLReportTime | M | T | F | T | T |

#### 6.3.8.3 Attribute constraints

None.

#### 6.3.8.4 Notifications

None.

### 6.3.9 GeneratedAlarmResult <<dataType>>

#### 6.3.9.1 Definition

This data type represents the alarm result information generated by the CCL, which is a list of attributes that describe the result of the Fault Management for each alarm.

#### 6.3.8.2 Attributes

**Table 6.3.9.2-1**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Attribute name** | **S** | **isReadable** | **isWritable** | **isInvariant** | **isNotifyable** |
| alarmId | M | T | F | T | F |
| alarmClearedStatus | M | T | F | T | F |
| identifiedRootCauseInformation | M | T | F | T | F |
| enhancedCorrelationInformation | M | T | F | T | F |

#### 6.3.9.3 Attribute constraints

None.

#### 6.3.9.4 Notifications

None.

### 6.3.10 CCLPurpose <<dataType>>

#### 6.3.10.1 Definition

This data type represents a single purpose that describes what a CCL can do. The purpose is a list of characteristics that describe the capabilities of the CCL.

#### 6.3.10.2 Attributes

Table 6.3.10.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
|  |  |  |  |  |  |
| **Attributes related to role** |  |  |  |  |  |
|  |  |  |  |  |  |

#### 6.3.10.3 Attribute constraints

None.

#### 6.3.10.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions..

### 6.3.11 CCLScopeCoordinationCapability <<dataType>>

#### 6.3.11.1 Definition

This data type represents the information and a capability of the ConflictManagementAndCoordinationEntity for Coordinating CCL instances to handle different CCL conflicts.

#### 6.3.11.2 Attributes

Table 6.3.11.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| cCLCoordinationCapabilityID | M | T | T | T | T |
|  |  |  |  |  |  |

#### 6.3.11.3 Attribute constraints

None.

#### 6.3.11.4 Notifications

The common notifications defined in clauses 6.1 are valid for this IOC, without exceptions.

### 6.3.12 CCLActionConflictsHandling <<datatype>>

#### 6.3.12.1 Definition

This defines the handling of CCL action conflict between the two existing CCLs.

#### 6.3.12.2 Attributes

Table 6.3.12.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| conflictInformation | M | T | T | F | T |
| conflictResolution | M | T | T | F | T |
| targetCCL | M | T | F | F | T |

#### 6.3.12.3 Attribute constraints

None

#### 6.3.12.4 Notifications

The common notifications defined in subclause 4.1.2.5 are valid for this IOC, without exceptions or additions.

### 6.3.13 ConflictInformation <<datatype>>

#### 6.3.13.1 Definition

This defines the information related with a conflicting CCLs that have been detected.

#### 6.3.13.2 Attributes

Table 6.3.13.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| conflictingCCLId | M | T | T | F | T |
| conflictingActions | M | T | T | F | T |

#### 6.3.13.3 Attribute constraints

None

#### 6.3.13.4 Notifications

The common notifications defined in subclause 4.1.2.5 are valid for this IOC, without exceptions or additions.

### 6.3.14 ActionConflictResolution <<datatype>>

#### 6.3.14.1 Definition

This defines the information related with conflict resolution configured by the MnS Consumer.

#### 6.3.14.2 Attributes

Table 6.3.14.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| conflictingCCLId | M | T | T | F | T |
| cCLGoalBreachPercentage | M | T | F | F | T |

#### 6.3.14.3 Attribute constraints

None

#### 6.3.14.4 Notifications

The common notifications defined in subclause 4.1.2.5 are valid for this IOC, without exceptions or additions.

## 6.4 Attribute definitions

### 6.4.1 Attribute properties

Table 6.4.1-1

| Attribute Name | Documentation and Allowed Values | Properties |
| --- | --- | --- |
| scopeType | It indicates the type of scope that represented by the particular scope instance.  allowedValues: CCL\_MEASUREMENT\_SCOPE, CCL\_TARGET\_SCOPE, CCL\_CONTROL\_SCOPE, CCL\_IMPACT\_SCOPE  Editor’s Note: The allowed values will be revisited | type: Enum  multiplicity: 1..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| coordinationCapability | It indicates a capability of a coordination entity to coordinate CCL conflicts | type: CoordinationCapability  multiplicity: \*  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| cCLCoordinationCapabilityID | It indicates an identifier for a specific CCL conflicts coordination capability | type: String  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| closedControlLoopRefList | It indicates a list of DN for ClosedControlLoop Instances.  allowedValues: N/A | type: DN  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLScopeCoordinationCapability | It indicates a specific type of CCL conflict coordination capacity | type: CCLScopeCoordinationCapability  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| coordinatedCCLsScopes | It indicates the scopes of the CCL that are coordinated by the coordinationEntity  It is a pair <string\_1, string\_2 > where string\_1 is the DN of a CCL being coordinated and string\_2 the DN of that CCL’s CCLScope. | type: pair <string, string >  multiplicity: 2 ..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| operationalState | It indicates the operational state of the ClosedControlLoop instance. It describes whether the resource is installed and partially or fully operable (Enabled) or the resource is not installed or not operable (Disabled).  AllowedValues; Enabled/Disabled  allowedValues: "ENABLED", "DISABLED".  The meaning of these values is as defined in 3GPP TS 28.625 [14] and ITU-T X.731 [15]. | type: ENUM  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: Disabled  isNullable: False |
| administrativeState | It indicates the administrative state of the ClosedControlLoop instance. It describes the permission to use or the prohibition against using the ClosedControlLoop instance. The administrative state is set by the MnS consumer.  AllowedValues; Locked/Unlocked  allowedValues: "LOCKED", "UNLOCKED".  The meaning of these values is as defined in 3GPP TS 28.625 [14] and ITU-T X.731 [15]. | type: ENUM  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: Locked  isNullable: False |
| cCLComponentsInfo | It indicates information on the constituent components of a CCL.  allowedValues: N/A | type: CCLComponentInfo  multiplicity: 1..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLComponentId | It indicates the identifier of a CCL component. It is the DN of an object instantiated to act as a component of the CCL | type: DN  multiplicity: 1..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLSteps | It indicates the CCL steps or functionality that is accomplished by a CCL component.  allowedValues: DATA\_COLLECTION, ANALYSIS, DECISION, EXECUTION | type: Enum  multiplicity: 1..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| FaultManagementAlarmIdList | It describes the list of IDs of alarms to be managed by Fault Management CCL.  allowedValues: A list of alarmIds as specified in TS 28.111 [4], clause 7.4.1 | type: List  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: True |
| FaultManagementTimeWindow | It describes the information of a time window (including start and end time) specified by the consumer for fault management to carry out troubleshooting and to clear the alarms.  allowedValues: timeWindow as defined in 3GPP TS 28.622 [5], clause 4.4.1 | type: TimeWindow  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: True |
| FaultManagementBackUpObjectRequirement | It describes whether to back-up the alarmed object is required by the consumer before fault management.  allowedValues: True, False | type: Boolean  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| FaultManagementIsolateObjectRequirement | It describes whether to isolate the alarmed object from interaction with other objects is required by the consumer before fault management.  allowedValues: True, False | type: Boolean  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| clearUserId | It carries the identity of the Fault Management CCL who is the consumer that invokes the clearAlarms operation.  allowedValues: clearUserId as defined in 3GPP TS 28.111 [4], clause 7.4.1 | type: string  multiplicity: 0..1  isOrdered: N/A  isUnique: N/A defaultValue: None  isNullable: False |
| FaultManagementCCLReport | It describes the Fault Management CCL report.  allowedValues: Not Applicable | type: FaultManagementCCLReport  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| GeneratedAlarmResultList | It describes the list of generated alarm results  allowedValues: A list of GeneratedAlarmResult | type: List  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| GeneratedAlarmResult | It describes the result for each alarmId listed in FaultManagemetAlarmIdList  allowedValues: Not Applicable | type: GeneratedAlarmResult  multiplicity: 1..\*  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| FaultManagementCCLReportTime | It describes the time when the FaultManagementCCLReport is created.  allowedValues: DateTime as specified in TS 28.622 [5]. | type: DateTime  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| alarmId | It identifies an AlarmRecord as specified in TS 28.111 [4]  allowedValues: A string as specified in TS 28.111 [4] | type: string  multiplicity: 1  isOrdered: N/A  isUnique: N/A defaultValue: None  isNullable: False |
| alarmClearedStatus | It describes whether an alarm is cleared by the Fault Management CCL when the identified root cause is resolved.  allowedValues: True, False | type: Boolean  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| identifiedRootCauseInformation | It describes root cause information identified by the Fault Management CCL.  allowedValues: String | type: string  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| enhancedCorrelationInformation | It describes the list of correlated alarm Ids identified by the Fault Management CCL  allowedValues: A list of alarmId | type: List  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| cCLActionConflictsHandling | This defines the handling of CCL action conflict between the two existing CCLs. | Type: cCLActionConflictsHandling  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| conflictInformation | This defines the information related with a conflicting CCL. | Type: ConflictInformation  multiplicity: \*  isOrdered: True  isUnique: False  defaultValue: None  isNullable: False |
| conflictResolution | This defines the information related with conflict resolution. | Type: ConflictResolution  multiplicity: \*  isOrdered: True  isUnique: False  defaultValue: None  isNullable: False |
| targetCCL | The identification of the CCL that need to be deleted or updated to resolve conflict. This will be decided as per the information ConflictResolution. | Type: Dn  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| conflictingCCLId | This indicates the CCL identification | Type: Dn  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| conflictingActions | This provides the set of actions that have been taken by the CCL as part of the Execute step. | Type: String  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLPriority | This provides the priority of the CCL. This will be the numerical value between 1 to 10, with 1 being the least priority. | Type: String  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| cCLMetricBreachPercentage | It defines the breach percentage per metric in terms of how bad the metric(s) is breached. For example, if the metric of guaranteed throughput is 200mbps and the actual throughput is coming to be 100mbps then the breach percentage would be 50%. The CCL that have higher percentage of breach will be prioritized | Type: Integer  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| cCLComponentList | It indicates the list of components acting as steps of the CCL, each either a MnF or a MnS producer whose services can be part of the CCL. The cCLComponent may have a role among MONITOR; ANALYSIS; DECISION; EXECUTION. Or OTHER. OTHER. Is used for example in the cases where a components fulfils more than 1 role or where the role can be simply described by the four options.  The cCLComponents are sequenced, i.e., cCLComponents is an ordered list. For example, if there are 2 steps that contribute to the analysis role, it is necessary to show how those steps are sequenced. The order in which they are listed indicates the order in which their services should be chained to complete the CCL | type: CCLComponent  multiplicity: 1..\*  isOrdered: True  isUnique: True  defaultValue: None  isNullable: False |
| cCLType | It indicates a type or Category of CCL that is to be instantiated or dynamically composition. It indicates the kind of capability that will be accomplished by the CCL instance, e.g. ENERGYOPTIMIZATION, SLICEASSURANCE, etc.  The specific details, characteristics and behavior of a CCL for a given CCL type are then written into the CCL purpose.  Editor’s Note: Documentation and Allowed values will be revisited | type: String  multiplicity: 1  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLComponentRole | It indicates a role accomplished by CCL component.  AllowedValues: MONITOR; ANALYSIS; DECISION; EXECUTION, OTHER. Is used for example in the cases where a components fulfils more than 1 role or where the role can be simply described by the four options | type: Enum  multiplicity: 1..\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLComponentIdentification | It indicates the entity accomplishing the component.  It may be the DN of an MOI or the combination of URI and DN that can be used to fulfil that role. | Type: String  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| cCLActionTrigger | This defines the criteria/conditions under which the CCL is allowed to take actions. | Type: CCLTrigger  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| desiredBehavior | This will define the corresponding behavior of the CCL. The behaviours can be represented by an ENUM to include:  - DECISION\_ACTIVATION: The CCL executes the recommendations that it derives on to the network.  - NOTIFY\_RCOMMENDATION: The CCL starts processing input to derive recommendations but without the corresponding actions executed on the network. Instead, the recommendation is notified to the consumer who then considers whether it should be applied or not.  - DO\_NOTHING: do not do anything. | Type: ENUM  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |

# 7 Procedures

## 7.1 Procedure for conditional trigger/instantiation of CCLs

A diagram of a product

AI-generated content may be incorrect.

Figure 7.1-1: Procedure and interactions for conditional trigger/instantiation of CCLs

Step 0: There exists an object exposing the MnS producer responsible for instantiating CCLs. This object may be represented by a subnetwork or managed element.

Step 1: The MnS consumer creates on the MnS producer responsible for instantiating CCLs the set of conditions to be evaluated for instantiation of the CCL. These conditions are created as an instance of TriggerConditionDescriptor defined in 28.572. TriggerConditionDescriptor describes the conditions that should be evaluated including performance, provisioning and fault management conditions. The performance conditions includes managed object, measurement/KPI name and the trigger value. The provisioning conditions includes the managed object, location, event and time of the provisioning events. The fault conditions includes managed object, alarmSeverityThreshold and alarmTypeThreshold.

Step 2: The MnS producer monitors the network to detect when the conditions defined in TriggerConditionDescriptor evaluate to TRUE.

Step 3: If conditions in TriggerConditionDescriptor evaluate to TRUE, the MnS producer instantiates the CCL.

Step 4: For the instantiated CCL, the MnS producer may notify the conditions that triggered the CCL.

## 7.2 Procedure for conditional composition of CCLs

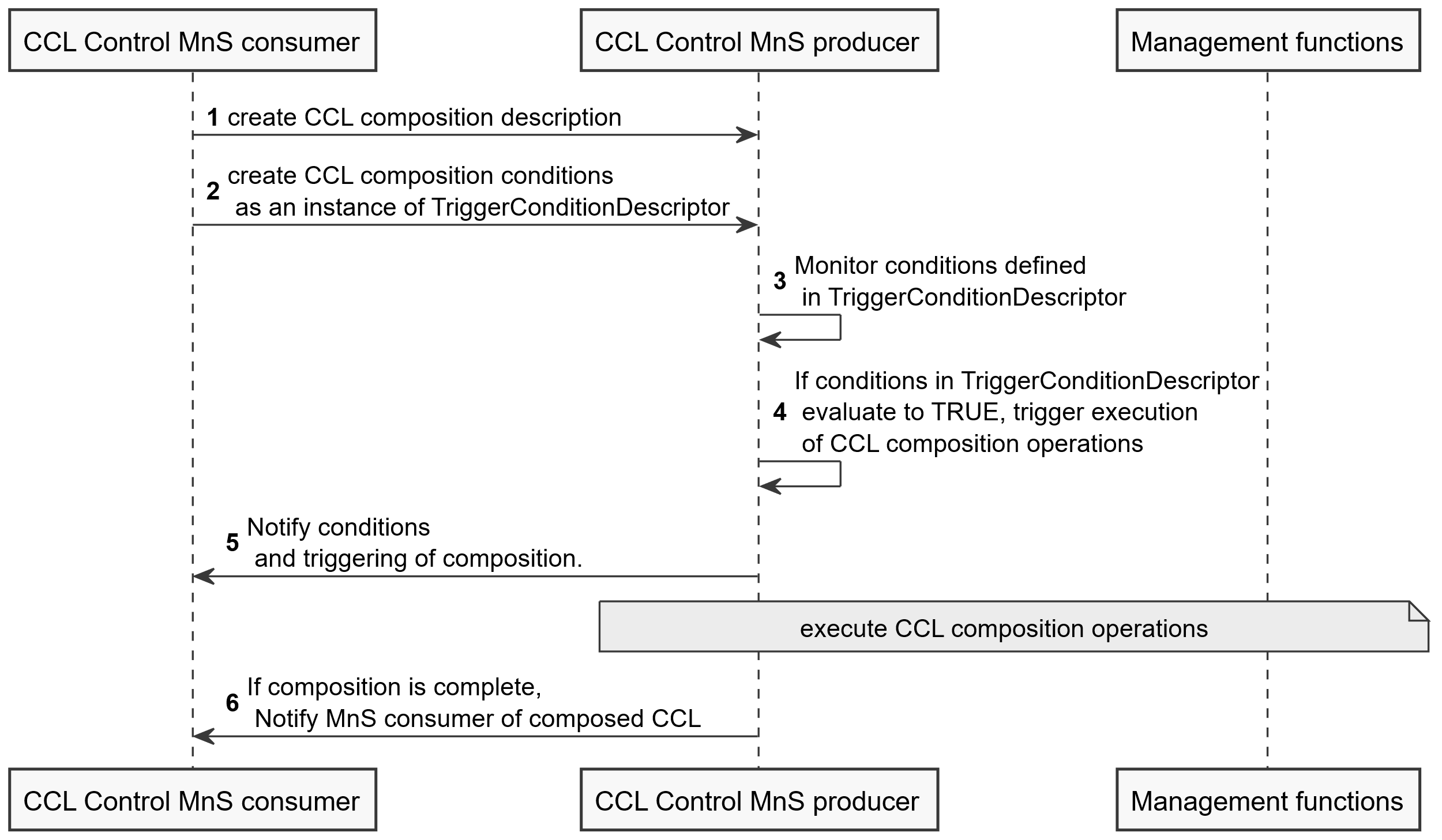


Figure 7.2-1: Procedure and interactions for conditional composition of CCLs

Step 0: There exists an object exposing the MnS producer responsible for instantiating CCLs. This object may be represented by a subnetwork or managed element.

Step 1: The MnS consumer creates on the MnS producer responsible for instantiating CCLs the CCL composition operations description which contains the details on the provisioning actions to be undertaken – in this case the operations for composing the CCL. These may include

- createMOI operations for instantiating the objects to be used as components of the closed control loop, e.g., a PMJob to be used to collect data

- modifyMOI operations for configuring the instantiated components to enable them to operate as a single loop, e.g., to configure the PMJob to deliver data to an analytics instance

Step 2: The MnS consumer creates on the MnS producer responsible for instantiating CCLs the set of conditions to be evaluated for composing the CCL. These conditions are created as an instance of TriggerConditionDescriptor defined in 28.572. TriggerConditionDescriptor describes the conditions that should be evaluated including performance, provisioning and fault management conditions

Step 3: The MnS producer monitors the network to detect when the conditions defined in TriggerConditionDescriptor evaluate to TRUE.

Step 4: If conditions in TriggerConditionDescriptor evaluate to TRUE, the MnS producer triggers execution of CCL composition operations.

Step 5: For the triggered CCL composition, the MnS producer may notify the conditions that triggered the composition or the composed CCL.

Step 6: The MnS producer executes the CCL composition operations through interaction with other management functions and services. When the composition is complete, the MnS producer may notify the MnS consumer of composed CCL.

## 7.3 CCL Performance Monitoring

When the PA (Performance Assurance)MnS consumer notices that a slice or network performance is degrading, it may require to know information about available CCLs that have the goals related to this performance degradation. This may imply that the performance of the related CCL is not as expected. This requires performance management to be done on the available CCL including further actions such as evaluating and updating closed control loops. The metrics for assessing performance of CCLs, for example, total number of occurrences of a goal breach, time taken by CCL to meet a breached goal, total number of conflicts occurred by a CCL are defined in clause 8. A procedure for performance management of CCLs involving these performance metrics is described below



Figure 7.3-1: Performance monitoring procedure for a closed control loop

Step 1. PA/CCL MnS consumer notices that a certain performance metric of a SLS or a network starts degrading.

Step 2. PA/CCL MnS consumer sends getMOIAttributeRequest message to PA/CCL MnS producer for getting information about all CCLs attributes.

Step 3. PA/CCL MnS producer provides this information of all CCLs to the consumer in getMOIAttributeResponse message.

Step 4. PA/CCL MnS consumer identifies the CCL (n) which is responsible for maintaining the performance of slice or network.

Step 5. PA/CCL MnS consumer sends createMOI(PerfMetricJob) request to PA/CCL MnS producer for obtaining status of following performance metrics for that particular CCL(n) as defined in clause 8 - TotalAssuranceGoalBreach, TimeCorrectiveGoalMeet, TotalCclConflicts\_Filter.

Step 6. PA/CCL MnS producer provides requested performance metric values via createMOI() Response message to PA/CCL MnS consumer.

Step 7. PA/CCL MnS consumer has two choices – either to update the existing CCL n (of step 4) to achieve the desired goal or to create a new CCL for the same. If PA/CCL MnS consumer chooses to modify an existing CCL, it sends a modifyMOIAttributes request message for that CCL or it can also update by sending changeMOIs request message to PA/CCL MnS producer.

Step8. Accordingly, PA/CCL MnS producer sends modifyMOIAttributes Response or changeMOIs response message to PA/CCL MnS consumer for the updated attributes of CCL n.

Step9. If PA/CCL MnS consumer chooses to create a new CCL for the desired goal, it does so by sending createMOI Request message to PA/CCL MnS producer.

Step10. PA/CCL MnS producer provides createMOI() Response message for the newly created CCL MOI to PA/CCL MnS consumer.

## 7.4 CCL decision escalation

To enable escalation, there has to be entities to which decision can be escalated, called escalation recipients. These are mainly closed control loops but other decision makers, e.g. AIML inference functions could be used as escalation recipients. The CCL which wishes to escalate a decision is named an escalator CCL.

To enable escalation, each CCL contains an attribute identifying an entity acting as an escalation recipient to which a decision is escalated. The CCL also contains an attribute for defining the condition that triggers the escalation. For example, the CCL may trigger escalation when its level of confidence in the derived decision is below some threshold, in which case the confidence threshold is the condition for triggering the escalation. The confidence threshold attribute enables the CCL to autonomously make decisions for each situation and context based on its computed confidence level in the given situation. If the confidence level is lower than the confidence threshold the decision is escalated otherwise the decision is executed.

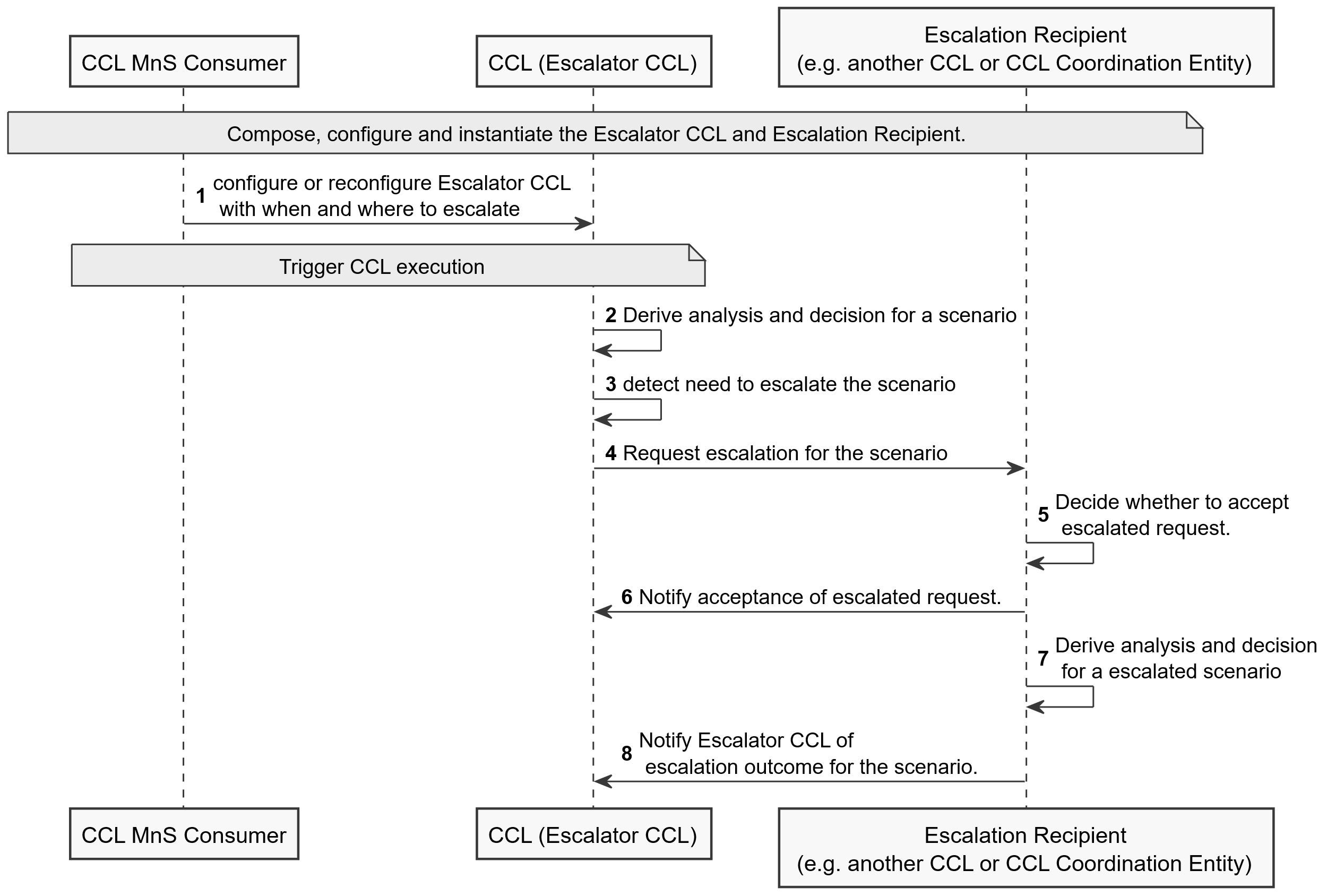


Figure 7.4-1: Procedure and interactions for CCL decision escalation

Step 0. The escalator CCL and the escalation recipient are composed, configured and instantiated.

Step 1. The MnS consumer configure the escalator CCL with information about the conditions under which to escalate and when to escalate to (the escalation recipient ). The escalator CCL can the trigger escalation either on its own or based on extra information form the Mns consumer.

Step 2. The escalator CCL executes analysis and decision making for a scenario. If the escalator CCL is confident with its decision it executes as normal

Step 3. The escalator CCL detect the need to escalate, e.g., for the case where it is not confident with its decision, the lack of confidence is the indicator of a scenario that should be escalated.

Step 4. When a CCL requires an escalation, escalator CCL instantiates a request for escalation on the escalation recipient.. The escalation request includes:

- An attribute for proposed CM change as a plan containing the configuration management changes that has been proposed by the escalator CCL.

- An attribute for the context and conditions describing decision constraints observed by the escalating CCL in making the decision(s).

Step 5. Based on the information in the escalation request, the capabilities of the escalation recipient as well as its observations of the evaluated network state, the escalation recipient decides whether it can undertake the escalation or not.

Step 6. The escalation recipient can notify (send an acceptance of) the escalation request.

Step 7. For an accepted escalation request, the escalation recipient derives an outcomes for the request

Step 8. The escalation recipient provides the outcomes to the escalator CCL, by writing it into an escalation outcome attribute on the escalation recipient. The outcome may then be written into an equivalent attribute on the escalator CCL, i.e., the escalator CCL contains an attribute for the escalation outcome to which the escalation recipient writes its computed escalation outcome. The escalation recipient contains an attribute for an escalation outcomes report in which it writes the derived outcomes for each corresponding escalation request. This can then be notified to the escalator CCL which subsequently reads it to obtain the recommendations.

The escalation outcome indicates whether the escalator CCL should take any action and what that action is. Accordingly, it contains an ENUM attribute to indicate what should be done by the escalator CCL, with the values:

- "DONOTHING"- indicating that the escalator CCL does not need to take any action, i.e. the escalation recipient is addressing the scenario.

- "APPLYACTION"- indicating that the escalator CCL should apply a specific set of actions proposed by the escalation recipient. The action is written into a proposed-actions attribute of the escalation outcome, which is of the type plan according to TS 28.572[6].

- "APPLYGUIDANCE"- indicating that the escalator CCL should compute a new CM change based on the guidance from the escalation recipient. The guidance is written into the proposed-actions attribute.

## 7.5 CCL-impact assessment and metric conflicts resolution

A CCL (called the actor-CCL) may not know the full scope that its actions will impact. And this may also not be known by the CCL coordination entity, In that case, the impact can be collected from the entities that have been affected by the CCL’s actions - jointly called impacted entities. The CCL contains an attribute, called executedAction attribute, which contains information indicating that an action has been taken that may affect the other CCLs (thus requesting feedback on how much impact there has been); and the CCL-action-impact time indicating the time when the affected entities should provide feedback. Any entity which may be impacted by the CCL actions (e.g. e.g. the CCL coordination entity or other CCLs) subscribes to be notified of changes to the executedAction and the related CCL-action-impact time.

After an action, the CCL updates the executedAction so that notifications are sent to the subscribed entities to indicate that if the entity is affected, it should provide its feedback on the effect in a time not exceeding the CCL-action-impact time. The notification may also be sent to the CCL coordination entity which then notifies that respective affected entities, e.g. other CCLs or other management functions.

The Impacted entity computes its observed impact in form of an index, called the Action Quality Indicator (AQI), that describes and quantifies the observed impact, i.e. it indicates the degree to which the action was good or bad to their objectives. The Action Quality Indicator is an integer in the range [0,10] where "0" indicates that the action was completely unacceptable and should never be reused in that context while "10" indicates that the action had very good outcomes for the reporting Impacted entity (e.g. the affected CCL). An index is used instead of sending specific metrics measured by each Impacted entity because specific metrics would require the actor-CCL to understand all the different metrics in exactly the same way as the Impacted entities do, which is not guaranteed to always be true. The AQI is specific to each CCL and to each scenario thar the CCL evaluates - since it is used to check how good or bad an action was for that CCL in that scenario. Accordingly, its computation would vary depending on the CCL and scenario but can be computed in a uniform way as a weighted sum of normalized KPIs) of that CCL.

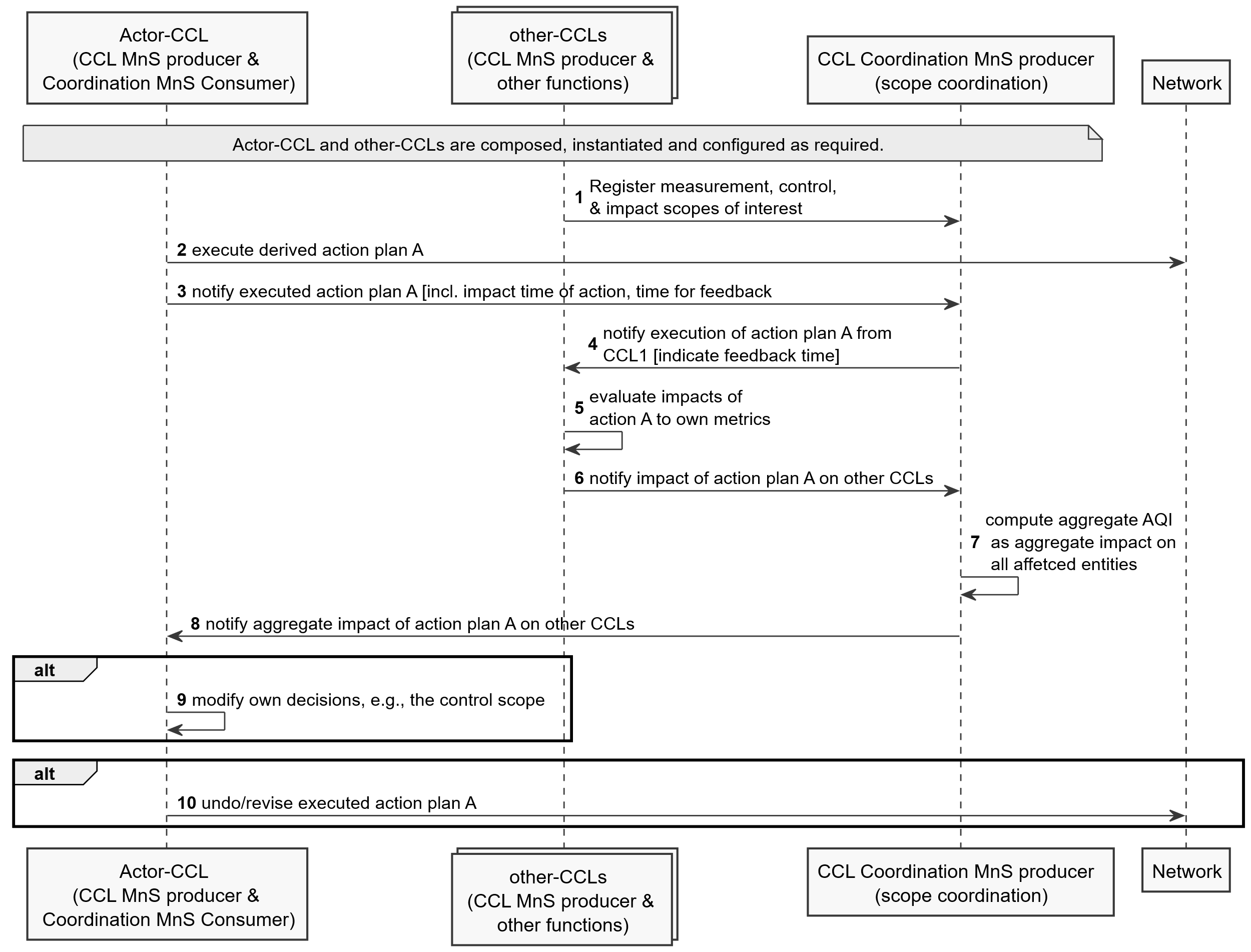


Figure 7.5-1: CCL-impact assessment and actual metric-value conflicts resolution

Step 0. The set of CCLs are composed, configured and instantiated.

Step 1 The CCLs register their scopes of interest to the coordination entity including the scopes where they take measurements, take control actions as well as where their actions are expected to impact. Where applicable, the scope have also been coordinated to ensure there are no conflicts for desired impacted scopes, the desired outcomes on the impacted scopes, cross impacts between measurement and control scopes.

Step 2. The acting CCL derives and executes an action plan onto the network.

Step 3. After an action, the CCL updates the executedAction so that notifications are sent to the CCL coordination entity to indicate that if an entity is affected, it should provide its feedback on the effect in a time not exceeding the CCL-action-impact time.

Step 4. The CCL coordination entity then notifies that respective affected entities, e.g. other CCLs or other management functions

Step 5. The impacted entity collects information on its metrics, (e.g., a using PM job), based on which, it computes its observed impact in form of an index, called the Action Quality Indicator (AQI), that describes and quantifies the observed impact, i.e. it indicates the degree to which the action was good or bad to their objectives.

Step 6. The impacted entity sends the AQI to the coordination entity within the CCL-action-impact time that was notified by the actor-CCL. The Action Quality Indicator is delivered to the actor-CCL in one of two ways:

- Impacted entity writes the AQI into an equivalent attribute called “reported AQIs” on the coordination entity, i.e., the coordination entity contains an attribute for the Action Quality Indicator to which each impacted entity writes its computed AQI. The reported AQIs attribute is a list to which each affected entity appends a value.

- The impacted entity contains an attribute for an observed AQI in which it writes the computed AQI. This can then be notified to the coordination entity which subsequently reads it to obtain the AQI for that impacted entity.

Step 7. The coordination entity determines the aggregate impact on all affected entities. To enable the CCL coordination entity to determine how much impact actor-CCL had on all the other CCLs together, the CCL coordination entity aggregates the impact based on the reported Action Quality Indicators from the respective impacted entities. The AQI can be computed as a weighted average of the AQI sent by the individual impacted entities. The AQI is delivered to the actor-CCL in one of two ways:

- The coordination entity writes the AQI into an equivalent attribute called “reported AQIs” on the actor-CCL, i.e., the actor-CCL contains an attribute for the Action Quality Indicator to which coordination entity writes the computed aggregate AQI.

- The CCL coordination entity contains an attribute for the aggregate AQI in which the coordination entity writes the computed aggregate AQI that is computed from the AQIs reported by the multiple affected CCLs. On modifying this aggregate AQI attribute, this can then be notified to the actor-CCL which subsequently reads it to obtain the aggregate AQI.

Step 8. The coordination entity sends the aggregate AQI to actor-CCL which is then used by the actor-CCL to decide an appropriate action to minimise the impact.

Step 9. The actor-CCL evaluates the impacts and if needed, the way it makes its decisions. For example, the actor-CCL can adjust the control scope (i.e., the acceptable range of values) on a given parameter.

Step 10. actor-CCL can revise the previous actions if needed. If it is computed by the CCL coordination entity, the coordination entity notifies it to the actor-CCL and may also propose a response action, e.g. to reverse the action that was taken.

NOTE: The data models for executedAction, reportedAQIs need to be extended.

# 8 CCL Performance Metrics

The performance metrics to evaluate performance of a CCL for its optimal execution are defined in order to enable operators to track the effectiveness of closed loop automation, identify areas for improvement, and make informed adjustments to CCL functionalities.

## 8.1 Total number of occurrences of a goal breach:

a) This measurement provides the total number of occurrences when a goal, as defined in CCL is breached during an observation time period i.e.granularityPeriod.

b) CC.

c) This is measured by counting each incidence when a goal is breached and incrementing the corresponding counter by one for each such occurrence within an observation time period i.e. granularityPeriod.

d) An integer value.

e) The measurement name has the form TotalGoalBreach.

f) ClosedControlLoop

g) Valid for packet switched traffic.

h) 5GS.

## 8.2 Time taken by CCL to meet a breached goal:

a) This measurement provides the time taken by a CCL to meet a breached goal after activating it again.

b) DER.

c) This is measured by considering the time stamp when a goal is breached and subtracting it from the time stamp when that goal is met after activating the CCL with required changes.

d) Each measurement is an integer representing the mean delay in milliseconds.

e) The measurement name has the form TimeBreachedGoalRecovery.

f) ClosedControlLoop

g) Valid for packet switched traffic.

h) 5GS.

## 8.3 Total number of conflicts occurred by a CCL:

a) This measurement provides the total number of conflicts that occur between a CCL under consideration and any other CCL during an observation time period i.e. granularityPeriod.

b) CC.

c) This is measured by counting each incidence when conflict occurs between a CCL under consideration and the other CCL and incrementing the corresponding counter by one for each such occurrence within an observation time period i.e. granularityPeriod

d) An integer value.

e) The measurement name has the form TotalCclConflicts\_Filter, where filter is either Implicit or Explicit. Implicit represents the action conflict i.e. conflict between two existing CCL and target represents the explicit conflict i.e. conflict between an existing CCL and a requested CCL.

f) ClosedControlLoop

g) Valid for packet switched traffic.

h) 5GS.

# Annex A (informative): UML code for model diagrams

## A.1 UML code for CCL management model diagrams

This annex contains the PlantUML source code for the NRM diagrams defined in clause 6.2 of the present document.

## A.2 CCL NRM fragment (Figure 6.2.1-1)

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

class ManagedEntity <<ProxyClass>>

class ClosedControlLoop <<InformationObjectClass>>

class CCLPurpose << ProxyClass >>

class CCLScope << InformationObjectClass >>

class CCLReport <<InformationObjectClass>>

class CCLComponent<<InformationObjectClass>>

ManagedEntity "1" \*-- "\*" ClosedControlLoop: <<names>>

ClosedControlLoop "1" <--> "\*" CCLPurpose

ClosedControlLoop "1" \*-- "\*" CCLScope: <<names>>

ClosedControlLoop "1" \*-- "\*" CCLReport: <<names>>

ManagedEntity "1" \*-- "\*" CCLComponent: <<names>>

ClosedControlLoop "1" -r-> "\*" CCLComponent

note left of ManagedEntity

Represents the following IOCs:

SubNetwork or

ManagedElement

end note

note top of CCLPurpose

Can be any of these CCL purposes:

NetworkProblemRecovery

FaultManagement

...

end note

@enduml

**Source code for Figure 6.2.1-1 CCL NRM fragment**

## A.3 NRM fragment for Coordination entity (Figure 6.2.1-2)

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

class ManagedEntity <<ProxyClass>>

class ConflictManagementAndCoordinationEntity <<InformationObjectClass>>

class CoordinationCapability <<dataType>>

class ClosedControlLoop <<InformationObjectClass>>

ManagedEntity "1" \*-- "1" ConflictManagementAndCoordinationEntity: <<names>>

ConflictManagementAndCoordinationEntity "1" -r- "\*" CoordinationCapability

ClosedControlLoop "\*" -r- "\*" ConflictManagementAndCoordinationEntity

note left of ManagedEntity

Represents the following IOCs:

Subnetwork or

ManagedElement

end note

note top of CoordinationCapability

Represents the following capabilities: ScopeCoordinationCoordination

TriggerCoordination

ActionExecutionCoordination

DirectActionsCoordination

IndirectTargetsCoordination

end note

@enduml

**Source code for Figure 6.2.1-2 NRM fragment for Conflict management and Coordination entity**

## A.4 CCL inheritance relationships (Figure 6.2.2-1)

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

class Top << InformationObjectClass >>

class ClosedControlLoop <<InformationObjectClass>>

class CCLReport <<InformationObjectClass>>

class CCLScope <<InformationObjectClass>>

class ConflictManagementAndCoordinationEntity <<InformationObjectClass>>

class CCLComponent<<InformationObjectClass>>

Top <|-- ClosedControlLoop

Top <|-- CCLScope

Top <|-- CCLReport

Top <|-- ConflictManagementAndCoordinationEntity

Top <|-- CCLComponent

@enduml

**Source code for Figure 6.2.2-1 CCL inheritance relationships**

# Annex B (informative): UML code for procedure diagrams

## B.1 UML code for CCL coordination procedure diagrams

This annex contains the PlantUML source code for the procedure diagrams in clause 7 of the present document.B.2 Procedure for conditional instantiation of CCLs (Figure 7.1-1)

@startuml Procedure for conditional composition of CCLs

skinparam Shadowing false

autonumber

skinparam monochrome true

participant "CCL MnS consumer" as CMC

participant "CCL MnS producer" as CMP

CMC -> CMP: create CCL instantiation conditions

CMP -> CMC: Monitor conditions defined

CMP -> CMP: If conditions in TriggerConditionDescriptor\n evaluate to TRUE instantiate CCL

CMP -> CMC: Notify conditions.

@enduml

**PlantUML source code for Figure 7.1-1 Procedure for conditional instantiation of CCLs**

## B.2 Procedure for conditional composition of CCLs (Figure 7.2-1)

@startuml Procedure for conditional composition of CCLs

skinparam Shadowing false

autonumber

skinparam monochrome true

participant "CCL Control MnS consumer" as MNSCS

participant "CCL Control MnS producer" as MNSPD

participant "Management functions" as MNFs

MNSCS -> MNSPD: create CCL composition desription

MNSCS -> MNSPD: create CCL composition conditions\n as an instance of TriggerConditionDescriptor

MNSPD -> MNSPD: Monitor conditions defined\n in TriggerConditionDescriptor

MNSPD -> MNSPD: If conditions in TriggerConditionDescriptor\n evaluate to TRUE, trigger execution\n of CCL composition operations

MNSPD -> MNSCS: Notify conditions\n and triggering of composition.

Note over MNSPD, MNFs: execute CCL composition operations

MNSPD -> MNSCS: If composition is complete,\n Notify MnS consumer of composed CCL

@enduml

**PlantUML source code for Figure 7.2-1 Procedure for conditional composition of CCLs**

## B.3 CCL decision escalation procedure (Figure 7.4-1)

B.2.1 CCL decision escalation procedure (Figure 7.A-1)

@startuml avoidance of potential action-execution-time conflicts - Information on detected conflict

skinparam Shadowing false

autonumber

skinparam monochrome true

participant "CCL MnS Consumer" as MNSCS

participant "CCL (Escalator CCL)" as ESCCL

participant "Escalation Recipient\n (e.g. another CCL or CCL Coordination Entity)" as ESCRP

Note over MNSCS, ESCRP: Compose, configure and instantiate the Escalator CCL and Escalation Recipient.

MNSCS -> ESCCL: configure or reconfigure Escalator CCL\n with when and where to escalate

Note over MNSCS,ESCCL: Trigger CCL execution

ESCCL -> ESCCL: Derive analysis and decision for a scenario

ESCCL -> ESCCL: detect need to escalate the scenario

ESCCL -> ESCRP: Request escalation for the scenario

ESCRP -> ESCRP: Decide whether to accept\n escalated request.

ESCRP -> ESCCL: Notify acceptance of escalated request.

ESCRP -> ESCRP: Derive analysis and decision\n for an escalated scenario

ESCRP -> ESCCL: Notify Escalator CCL of\n escalation outcome for the scenario.

@enduml

**PlantUML source code for Figure 7.4-1 CCL NRM fragment**

## B.4 CCL-impact assessment and metric conflicts resolution on unknown or unbounded impact-scope (Figure 7.5-1)

@startuml CCL-impact assessment and metric conflicts resolution on unknown or unbounded impact-scope

skinparam Shadowing false

autonumber

skinparam monochrome true

participant "Actor-CCL \n (CCL MnS producer & \n Coordination MnS Consumer)" as CL1

collections "other-CCLs \n (CCL MnS producer & \n other functions)" as CL2

participant "CCL Coordination MnS producer \n (scope coordination)" as xCL

participant "Network" as Net

Note over CL1, xCL: Actor-CCL and other-CCLs are composed, instantiated and configured as required.

CL2 -> xCL: Register measurement, control, \n& impact scopes of interest

CL1 -> Net: execute derived action plan A

CL1 -> xCL: notify executed action plan A [incl. impact time of action, time for feedback

xCL -> CL2: notify execution of action plan A from \nCCL1 [indicate feedback time]

CL2 -> CL2: evaluate impacts of \naction A to own metrics

CL2 -> xCL: notify impact of action plan A on other CCLs

xCL -> xCL: compute aggregate AQI\n as aggregate impact on\n all affected entities

xCL -> CL1: notify aggregate impact of action plan A on other CCLs

Alt

CL1 -> CL1: modify own decisions, e.g., the control scope

end

Alt

CL1 -> Net: undo/revise executed action plan A

end

@enduml

**PlantUML source code for Figure 7.5-1 CCL NRM fragment**

# Annex C (informative): Selection of conflict coordination approach

#### C.1 Alternative CCL coordination Approaches for conflicts handling

To address the conflicts, coordination interactions are required either among CCLs or between the CCLs and one or more higher hierarchy coordination functions to avoid or detect and resolve the conflicts. This is required when CCL are actuating in the same set of resources.

The coordination of CCLs could be accomplished via one of three approaches illustrated by Figure C.1-1:

- Distributed coordination with distributed execution (Figure C.1-1 a), where the CCLs directly coordinate with one another, and each manages execution of its decisions. The CCL exchange information with each other avoid, detect or resolve conflicts. The information may for example include notifications of executed actions or observed impacts.

- Hierarchical coordination with distributed execution (Figure C.1-1 b), where the CCLs coordinate through a separate coordination layer, say via a CCL coordination entity, but each manages execution of its coordinated decisions. The CCL exchange information with the CCL coordination entity to avoid, detect or resolve conflicts. A CCL may send notifications of its executed actions or observed impacts which the CCL coordination entity may relay to other CCLs. The CCL coordination entity may configure the CCLs but each CCL executes its action based the CCL coordination entity’s configuration.

- Hierarchical coordination and execution (Figure C.1-1 c), where the CCLs coordinate through a separate coordination layer, say via a coordination entity that besides coordination also manages execution of the coordinated decisions. The CCL exchange information with the CCL coordination entity to avoid, detect or resolve conflicts including notifications of their executed actions or observed impacts which the CCL coordination entity may relay to other CCLs. The CCL coordination entity may configure the CCLs and the CCL execute their actions through the CCL coordination entity.



Figure C.1-1: Closed Control Loop Coordination approaches

Distributed coordination can lead to too many exchanges between the CCLs which may unnecessarily clog the system. On the other hand, "Hierarchical coordination and execution" implies that too much responsibility is concentrated in a single CCL. A desired behavior is that the individual CCLs are responsible for their own decision execution, so it is recommended that to follow the "hierarchical coordination with distributed execution" approach. In this approach, the CCLs are responsible for making their decisions and executing actions, but they coordinate with the CCL coordinator before, during or after execution.

#### C.2 Hierarchical CCL-coordination-interactions for conflicts handling

To address the conflicts, coordination interactions are required between the CCLs and one or more higher hierarchy coordination functions to avoid or detect and resolve the conflicts among goals and their targets, control scopes or actions of the CCLs. The 3GPP management system includes at least one entity called the Coordination entity that undertakes the role of CCL coordination. The Coordination entity can be implemented as a CCL, an AIML inference engine or any other functionality that is found appropriate. The coordination entity may support coordination for conflict management for different conflicts described in clause 5.7-5.9 including Goal- and Goal-targets- Conflicts; scope conflicts, CCL-Trigger-time and CCL-action-execution-time conflicts, Direct actions conflicts as well as metric-value conflicts.

The coordination of CCLs could be required at different execution points of the CCL translating into different CCL coordination use cases with corresponding CCL coordination services required at those points as illustrated by example Figure C.2-1. The coordination of CCLs could be achieved via direct interaction among the CCLs or via a third-party entity, say called the CCLs coordination Function (or simply CCL Coordinator).



Figure C.2-1: Exemplary Closed Control Loop Coordination interaction points

NOTE: The terms at the top indicate general naming of the groupings of coordination interactions at the different execution points during the execution of the CCL. Action-space coordination implies coordinating the sets of actions that the different CCL can apply. Concurrency control implies coordinating the times at which different CCLs can execute actions. Action-impact assessment indicates interactions and processes on the evaluation of the impacts of the different CCLs.

The coordination purpose attributes contain the information and data needed or used by the coordination entity for interacting with the CCL when handling conflicts.

# Annex D: Change history

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Change history** | | | | | | | |
| **Date** | **Meeting** | **TDoc** | **CR** | **Rev** | **Cat** | **Subject/Comment** | **New version** |
| 2025-02 | - | n/a | - | - | - | Initial skeleton | 0.0.0 |
| 2025-02 | SA5#159 | S5-251107  S5-251137  S5-251123  S5-251124  S5-250944  S5-250945  S5-250943 | pCR | - | - | 28.567-001  pCR TS28.567 TS clause structure  pCR TS28.567 Concepts and overview  pCR TS28.567 Dynamic composition of CCLs  Rel-19 pCR 28.567 CCL creation based on historical data UC and Requirements  Rel-19 pCR TS 28.567 Add use case and requirements for Performance monitoring of Closed control loop  Rel19 pCR TS28.567 Add use case and requirements for CCL for Fault Mgmt | 0.1.0 |
| 2025-04 | SA5#160 | S5-251904  S5-251905  S5-251906  S5-251907  S5-251909  S5-251911  S5-251912  S5-252030  S5-252033  S5-252034 | pCR | - | - | Rel-19 pCR 28.567 Triggered CCL UC and Requirements  CCL Trigger conflicts  CCL Goal-targets and actions conflicts  CCL Indirect target conflicts  Informative Annex on CCL conflict handling approach  CCL decision escalation  pCR TS 28.567 Add use case and requirements for network performance problem recovery  CCL General NRM  Rel-19 pCR TS 28.567 Procedure and metrics for CCL performance monitoring  Coordinating CCLs with other functions | 0.2.0 |
| 2025-05 | SA5#161 | S5‑252491  S5‑252846  S5‑252848  S5‑252849  S5‑252850  S5‑252851  S5‑252852  S5‑252853  S5‑252854  S5‑252855  S5‑252856  S5‑252857  S5‑253007  S5‑253008  S5‑253009  S5‑253010  S5‑253020 | pCR |  |  | pCR TS 28.567 Correct information models for CCL management in clause 6.2.1  CCL coordination NRM  CCL concurrent-actions conflicts  CCL scope conflicts  Solution for Dynamic CCLs  Rel-19 pCR 28.567 solution for conflict management  Pseudo-CR on Rel-19 TS 28.567 Add Stage-2 Solutions for Fault Management CCL  Rel-19 pCR TS 28.567 Metrics for CCL performance monitoring  CCL-impact assessment and resolution  Procedure for CL decision escalation  CCL-impact assessment, feedback and resolution  Definitions of CCL conflicts  Control of CCLs  Example realizations of CCLs  CCL DraftTS Rapporteur corrections  Pseudo-CR on Rel-19 TS 28.567 Add Definitions of terms and abbreviations for closed control loop  Rel-19 pCR 28.567 solution for triggered CCL | 0.3.0 |