**3GPP TSG-SA5 Meeting #162 *S5-253861d1***

Goteborg, Sweden, 25 - 29 August 2025

**Source: Huawei**

**Title: Rel-19 pCR TS 28.567 Remove the descriptions related to target and goal**

**Document for: Approval**

**Agenda item: 6.19.4.1**

**Spec: 3GPP TS 28.567**

**Version: V0.3.0**

**Work Item: CCLM**

**Comments**

In accordance with the consensus reached during the preceding meetings, this proposal aims to eliminate descriptions related to target or goal.

**Proposed Changes**

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

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 specific requirements. 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 requirements. Any two CCLs with the same functionality may have the functionality supported in different count of steps implementing the functionality and 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 another 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 requirements or policies. Besides the provisioning needed to realize the requirement, 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])**

\* \* \* Next Change \* \* \* \*

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 Requirements, CCL Triggers, CCL actions and action plans as well as CCL scopes.

4.3.2 CCL Requirements

The CCL requirements are the set of outcomes that need to be achieved or realized by the CCL. Each expected outcome is called a CCL requirement. A CCL requirement contains for example a metric.

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.**

\* \* \* Next Change \* \* \* \*

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 requirements.

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 requirements.

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 requirement 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 requirements, 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 requirement 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 requirements 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 requirement(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 requirements, time taken to meet a breached requirements, 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 requirement 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 requirement. | 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 |

\* \* \* Next Change \* \* \* \*

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. requirement) to be instantiated under certain conditions and another with variations in requirements 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

A CCL may experience direct conflicts on its requirements, 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.

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 outcome being optimized by CCL-A |
| **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 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 optima 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.

5.7.2.2 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.

5.7.2.3 CCL concurrentmetric-value 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 management system should support the capability for avoidance of concurrent metric-value conflicts 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.

5.7.2.4 CCL non-concurrentactions conflicts handling –CONF\_04

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 requirements.

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

5.7.2.5 CCL non-concurrentmetric-valueconflicts handling – CONF\_05

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 couple, 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 CL desire different values for the metric, or different values for two metrics Y1 and Y2 but the requirements are coupled, the CCLs are in conflict for the metric resulting into a metric-value conflict.

Two 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 non-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** |

\* \* \* Next Change \* \* \* \*

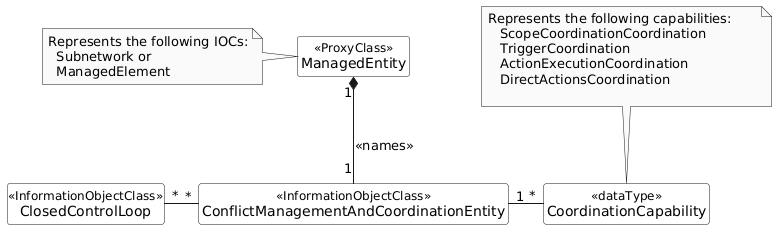
6.2 Class diagram

6.2.1 Relationships

A diagram of a computer program

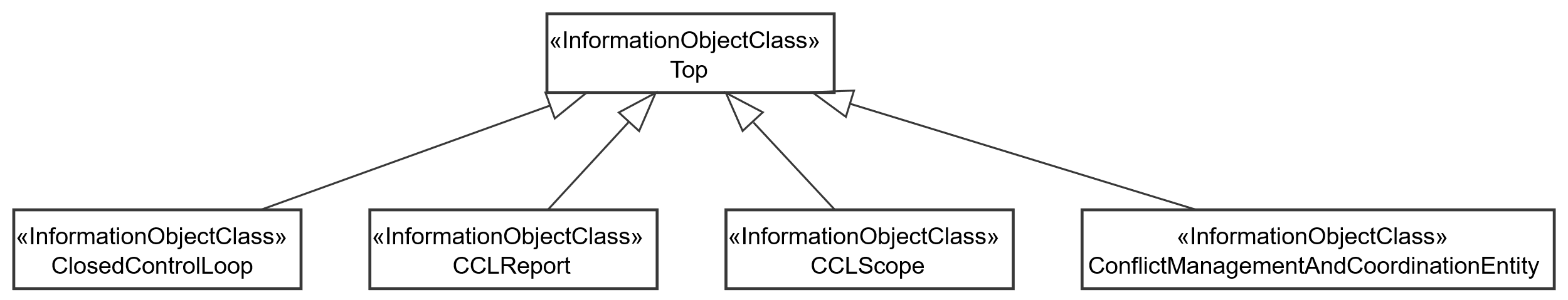
AI-generated content may be incorrect.

**Figure 6.2.1-1: Relations for common information models for CCLmanagement**



**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**

\* \* \* Next Change \* \* \* \*

6.3.14.2 Attributes

**Table 6.3.14.2-1**

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

\* \* \* Next Change \* \* \* \*

## 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\_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 a 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: Booelan  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: Booelan  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: Booelan  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 ating 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 caes where a components fulfile more than 1 role or where the role can be siml y described by the four options.  The cCLComponents are sequenced, i.e., cCLComponents is an ordred 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 caes where a components fulfile more than 1 role or where the role can be siml y 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 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 behaviors 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 |

\* \* \* Next Change \* \* \* \*

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 requirements 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 requirement breach, time taken by CCL to meet a breached requirement, 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 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, 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.

\* \* \* Next Change \* \* \* \*

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 requirement breach:

a) This measurement provides the total number of occurrences when a requirement (e.g. a metrix), 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 requirement 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 TotalRequirementBreach.

f) ClosedControlLoop

g) Valid for packet switched traffic.

h) 5GS.

8.2 Time taken by CCL to meet a breached requirement:

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

b) DER.

c) This is measured by considering the time stamp when a requirement is breached and subtracting it from the time stamp when that requirement 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 TimeBreachedRequirementRecovery.

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 explicit 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.

\* \* \* Next Change \* \* \* \*

## 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

end note

@enduml

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

\* \* \* Next Change \* \* \* \*

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 requirements, 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; 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.

\* \* \* End of Changes \* \* \* \*