**3GPP TSG-SA5 Meeting #162 *S5-253342***

Gothenburg, Sweden, 25 – 29 August 2025

**Source: Nokia**

**Title: Pseudo-CR on CCL metric-value conflict Coordination NRM**

**Document for: Approval**

**Agenda item: 6.19.4.1**

**Spec: 3GPP TS28.567**

**Version: 0.3.0**

**Work Item: Closed Control Loop Management**

**Comments**

This pCR is to add NRM and procedures for the CCL Coordinationas was agreed in the CCL study in TR28. 867

**Proposed Changes**

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

## 5.7 CCL Conflict management Capability - CONF

### 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.

Two (or more) CCLs configuring different control parameter may all influence the same metric. If the two CCLs 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 because even if the CCLs may not directly conflict for the same control parameter, their impacts are in conflict on the desired value of the metric or target. Such conflicts are thus called Metric-value conflicts.

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.

For a detected metric-value 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\_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 targets.

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 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.

\* \* \* Second Change \* \* \* \*

# 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 CCLmanagement

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

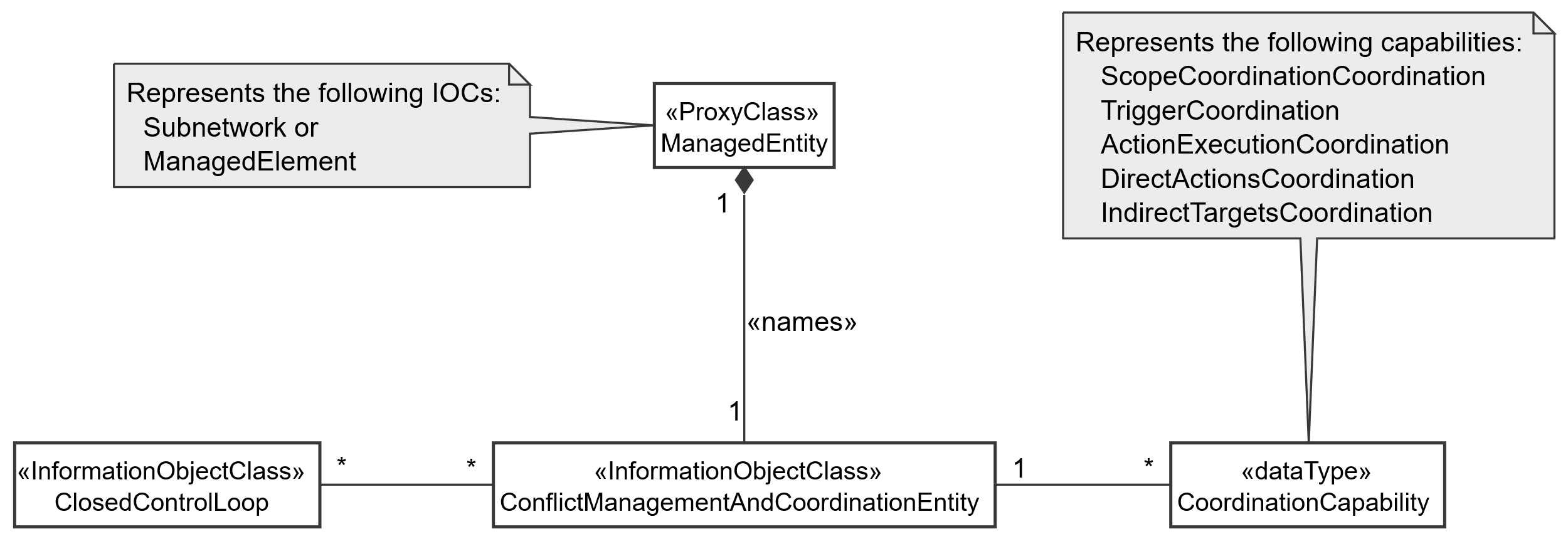


Figure 6.2.1-2: NRM fragment for conflict management and Coordination entity

### 6.2.2 Inheritance

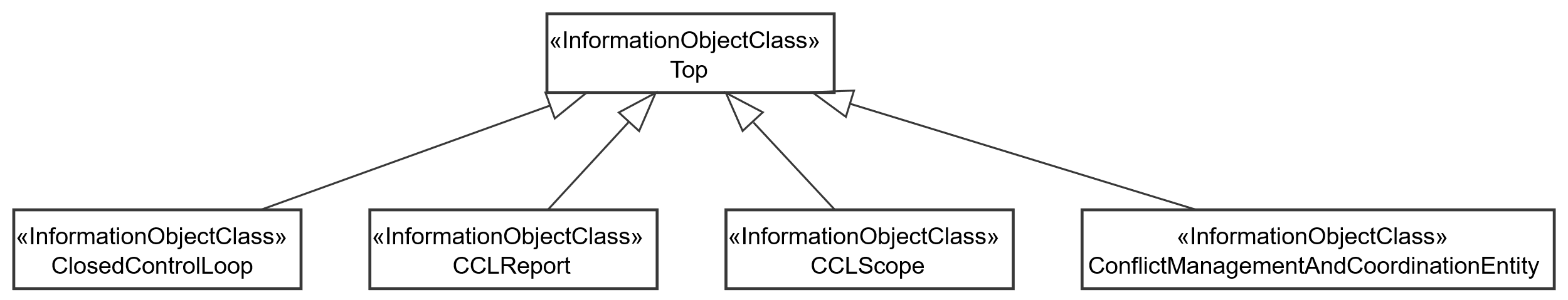


Figure 6.2.2-1: Inheritance Hierarchy for Closed Control Loops and for conflict management and Coordination entity

## 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 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 cCLComponents indicates the list of components which are combined to create a CCL.

The attribute 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 CCLControlLoop IOC includes attributes inherited from Top IOC (defined 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 |
| desiredMetrics | M | 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 |
| cCLMetricValueCoordinationCapability | 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 alist 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 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 alist 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 |
| detectedMetricValueConflicts | M | T | F | 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.D1 MetricValueConflict <<datatype>>

#### 6.3.D1.1 Definition

This data type represents the information on a metric-value conflict.

Each conflict includes an indication in ConflictType attribute for whether it is a potential conflict or an actual conflict that is observed.

#### 6.3.D1.2 Attributes

Table 6.3.13.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| conflictID | M | T | T | F | T |
| conflictingCCLs | M | T | T | F | T |
| conflictingMetrics | M | T | T | F | T |
| ConflictType | M | T | T | F | T |
| correlatedOscillationMetrics | M | T | T | F | T |

#### 6.3.D1.3 Attribute constraints

None

#### 6.3.D1.4 Notifications

### The subclause 6.x of the <<IOC>> using this <<dataType>> as one of its attributes, shall be applicable.

### 6.3.D2 CCLMetricValueCoordinationCapability

#### 6.3.D2.1 Definition

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

#### 6.3.D2.2 Attributes

Table 6.3.11.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | S | isReadable | isWritable | isInvariant | isNotifyable |
| cCLCoordinationCapabilityID | M | T | T | T | T |
| proposedReviseddActionPlan | M | T | F | T | T |
| observedMetricValueConflicts | M | T | F | F | T |
| actionPlanFailedCriteria | M | T | F | F | T |
| TrustedCCLs | M | T | F | F | T |
| flipflopMetrics | M | T | T | F | T |

#### 6.3.D2.3 Attribute constraints

None.

#### 6.3.D2.4 Notifications

The subclause 6.x of the <<IOC>> using this <<dataType>> as one of its attributes, shall be applicable.

### 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 |
| cCLMetricValueCoordinationCapability | It indicates a specific type of CCL conflict coordination functionality of the ConflictManagementAndCoordinationEntity | type: CCLMetricValueCoordinationCapability  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 |
| desiredMetrics | It indicates the set of metrics that the CCL intends to optimize. These need to be coordinated among several CCLs, e.g. so that 2 CCLs don’t aim to optimize the same metric | Type: string  multiplicity: 1...\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| proposedReviseddActionPlan | It indicates a compromise action plan proposed by the coordination entity for the case where the action plan executed by a CCL resulted in metric value conflict | Type: ActionPlan  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| actionPlanFailedCriteria | It indicates criteria which an action plan for which an action plan failed and caused metric value conflicts. | Type: string  multiplicity: 1...\*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| TrustedCCLs | It indicates the list of CCL that have performed consistently well and have achieved full trust that not further check of their actions is necessary. | Type: DN  multiplicity: \*  isOrdered: False  isUnique: True  defaultValue: None  isNullable: False |
| observedMetricValueConflicts | It indicates the list of observed metric value conflicts | Type: MetricValueConflict  multiplicity: \*  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| correlatedOscillationMetrics | It indicates the metrics noted to be experiencing correlated oscillations | Type: string  multiplicity: 1..\*  isOrdered: False  isUnique: TruedefaultValue: None  isNullable: False |
| conflictingMetrics | It indicates the list of metrics that are in conflict | Type: string  multiplicity: 1...\*  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| detectedMetricValueConflicts | It indicates the list of MetricValueConflicts that are detected by the coordinationEntity. Each entry is of type: MetricConflict | Type: MetricValueConflict  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| toleranceLimits | It indicates the limits within which the compromise on the parameters and metrics can still be acceptable. It is an integer indicting the acceptable percentage change in the values on parameters in a specific action plan.  allowedValues: [0, 100] | Type: integer  multiplicity: 1  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |
| flipflopMetrics | It indicates the list of metrics that are observed by a CCL as flip flopping. It is a pair <objDN, ffmetric> where objDN is DN of the managed object whose metric is flipflopping and ffmetric is identifier of the flip flopping metric. | Type: pair <DN, string>  multiplicity: \*  isOrdered: N/A  isUnique: N/A  defaultValue: None  isNullable: False |

\* \* \* Third Change \* \* \* \*

# 7 Procedures

## 7.E CCL metric-value conflicts avoidance and detection

### 7.E.1 Avoiding concurrent and non-concurrent metric-values conflicts

Each CCL has a control scope including a set of metrics. The metrics ma have prioritization among them, e.g. a handover optimization CCL may have more interest (higher priority) in controlling Cell individual offsets compared to controlling antenna tilts. To support detection and avoidance of potential non-concurrent metric-value conflicts, if the CCL has been pre-configured e.g., by the operator with the expected outcomes, the CCL may register its desired metrics, their priorities and outcomes with the CCLCoordinationEntity. This triggers the first evaluation for potential conflict, i.e. whether these metrics and outcomes are likely to conflict with those of another CCL.

Subsequently, potential metric-value conflicts are avoided using likely-impact of planned actions. For any CCL, large and frequent changes to network parameters may affect network stability since they increase the probability of occurrence of conflicts, i.e. avoiding making unnecessary configuration changes to the managed objects guarantees network stability and minimize the probability of conflicts between CCLs. This may then imply that executing large changes, e.g. to quickly improve the performance, in case of a poor decision, may also result in significant degradation. So, it is preferred to take small smooth changes in the case where the impact is not so clear, and only make the large changes when the CCL is sure that the impact is positive.

In case of a plan that results in a conflict, the CoordinationEntity sends its decision and possibly the failed criteria to the CCL - to either be executed or to be used to compute better decisions. It is assumed that based on feedback on the quality of its decisions, the CCL updates its decision-making engine and repeats the decision evaluation process. Then if the CCL has consistently made good large action-decisions, the coordinator CCL can consider the CCL as trusted to make such large decisions. The coordinator CCL informs the CCL that the CCL has consistently made good decisions and achieved its ultimate trust.

### 7.E.2 Detecting concurrent metric-values conflicts

For metric-values conflicts where actions are executed in a short interval form one another, detection can be possible. Two CCLs (CCL1 and CCL2) may optimize 2 target metrics Y1 and Y2, e.g. one intending to ensure "HO failure is < 2 %" while the other wants "SINR > 10dB". Due to coupling between Y1 and Y2, actions to optimize these by CCLs may lead to correlated oscillations/degradations in Y1 or Y2. 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 CoordinationEntity analyses the behavior of Y1 and Y2 to see if there are correlated oscillations as result of actions by any of the CCLs which then indicates potential conflict between CCL1 and CCL2. When the oscillations are observed, the CoordinationEntity informs the related CCLs (i.e. CCL1 and CCL2) about the detected potential conflict.

For detected potential conflict the CCL coordination service producer needs to confirm that it is an actual harmful conflict. This can be determined based on the severity of degradation in the performance metrics of the related CCLs. The threshold to determine the severity may be defined by the MnS consumer (e.g. the operator or coordinator CCL). If the degree of degradation is higher than the threshold then it is a confirmed conflict that requires resolution. Otherwise, no action is needed.

### 7.E.3 Detecting non-concurrent metric-values conflicts

For actions that are not executed within the same time frame, there are no correlated oscillations in the metrics, so the CCLs should detect potential conflicts themselves. The CCLs attempt to fulfil desired outcomes, and where they ae unable to, the CCL sends feedback to the CoordinationEntity indicating which outcomes on which metrics cannot be fulfilled. A CCL may for example indicate that there are ping-pong effects on a target, i.e. whenever the target is pushed in a given direction, it flips back to a previous state. The flipflop is an indication of a potential conflict which the CCL should notify to the CoordinationEntity. The CCL should notify the CoordinationEntity, e.g., the response could be that “desired outcomes on metric x cannot be achieved because it causes problems on higher priority metric y.”. Based on the feedback, the CoordinationEntity can confirm the existence of conflict, e.g. that other CCLs are requesting to readjust related parameters. The CoordinationEntity derives recommendations to the CCL including whether the CCL should change the prioritisation of its desired control metrics. The CoordinationEntity notifies the proposed changes to the CCL including setting control metrics or their priorities.

Note the resolution of concurrent and non-concurrent metric-values can apply the procedure for CCL-impact assessment and metric conflicts resolution as described in clause 7.5.

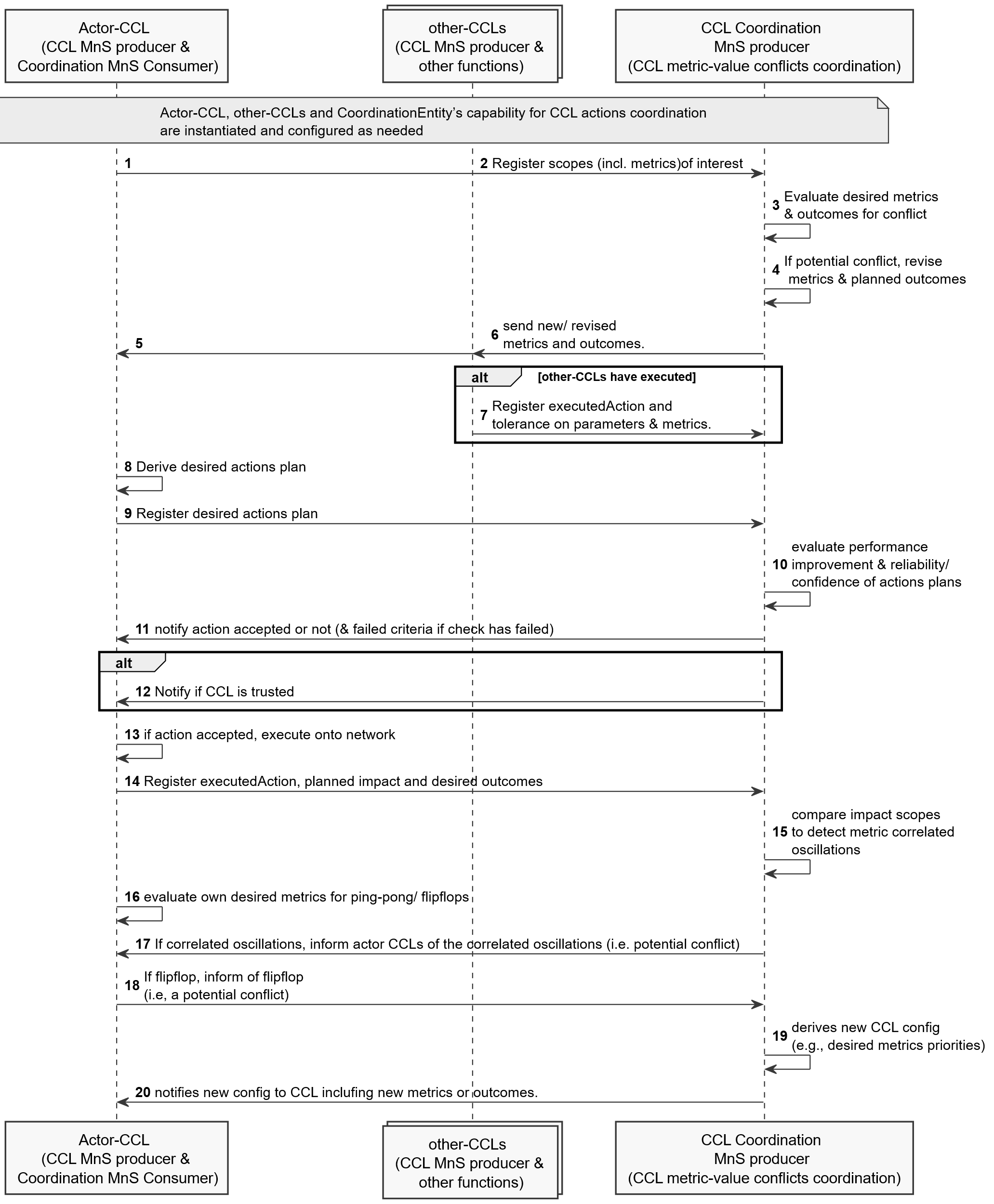


Figure 7.5-1: CCL metric-value conflicts avoidance and detection

Step 0 The set of CCLs and CoordinationEntity’s capability for metric-values conflicts coordination is instantiated and configured ( e.g., with the rules for evaluating and coordinating scopes for different use cases)

Step 1,2. The CCLs register their scopes of interest to the coordination entity including the scopes where they take measurements, take control actions, where their actions are expected to impact and their desired outcomes on those impact scopes. Where applicable, the scope have also been coordinated to ensure there are no conflicts for desired impacted scopes, the desired outcomes ion the impacted scopes, cross impacts between measurement and control scopes.

Step 3. The CCL coordination entity evaluates the metrics of interest and desired outcomes to see if they conflict with other CCLs. For example, based on the defined general objectives for the network scope (e.g. derived form an intent), the CCLCoordinationEntity may select the appropriate metrics and outcomes for the CCL.

Step 4. In case of a potential conflict, the CCL coordination entity derives revisions in the assigned metrics of interest and planned outcomes, to minimize contradictions or conflicts among the metrics and outcomes.

Step 5,6. The CCL coordination entity sends the selected new or revised metrics and outcomes to each CCL.

Step 7. Previous CCLs that have executed actions have registered to the CoordinationEntity, their executed actions, the scopes they expect to impact and their desired outcomes on those impact scopes.

Step 8. The actor CCL derives its desired action plan on to the network

Step 9. The actor CCL registers to the CoordinationEntity its desired action plan and the expected impact of that action plan (its claimed/predicted performance improvement) and reliability/confidence in that action/decision to be evaluated for potential significant degradation, i.e., that the actions are no unnecessarily too large.

Step 10 The coordinator CCL evaluates the claimed performance improvement and reliability/confidence to determine if the action should be allowed or not to avoid counter-productive actions - CCL making large changes, should have high reliability/ confidence and significant improvement in performance.

Step 11. The CoordinationEntity sends to the actor CCL its decision and the failed criteria in case the action plan has failed the evaluation For this, the CoordinationEntity updates the proposedReviseddActionPlan which is then notified to the respective CCL

Step 12. The coordinator CCL may also inform the CCL that the CCL has consistently made good decisions and achieved its ultimate trust. The CCL would not need to recheck its decision for appropriateness of the step change . For this the coordinationEntity updates the TrustedCCLs attribute with the DN of the CCL that has achieved full trust. The change is then notified to the CCL

Step 13. If the action is accepted, the actor CCL executes its desired action plan on to the network

Step 14. The actor CCL registers the executed action plans to the CoordinationEntity including the scopes they expect to impact and their desired outcomes on those impact scopes. The CCL writes into the desiredCCLActions attribute on the CoordinationEntity.

Step 15. The CoordinationEntity evaluates the impact scopes of the previous CCLs to detect metric oscillations which indicate a potential conflict.

Step 16. The CCL evaluates its desired metrics to see if there are ping-pong/ flipflop effects. The flipflop is an indication of a potential conflict which the CCL should notify to the CoordinationEntity.

Step 17. In case of correlated oscillations, the CoordinationEntity informs the actor CCLs of the correlated oscillations indicating a potential conflict. For this, the coordinationEntity updates the correlatedOscillation attribute in the metricValueConflict added to the list of observedMetricValueConflicts. It then notifies the CCL of the metricValueConflict

Step 18. In case of flipflop, the CCL informs the CoordinationEntity of the flipflop indicating a potential conflict. For this the CCL adds the metric that is flipflopping as an entry in the flipflopMetrics attribute of the coordination entity.

Step 19. The CoordinationEntity derives recommendations to the CCL including whether the CCL should change the prioritisation of its desired control metrics

Step 20. The CoordinationEntity notifies the proposed changes to the CCL including setting control metrics or their priorities.

\* \* \* Fourth Change \* \* \* \*

# 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 a 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 affetced 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**

## B.F CCL metric-value conflicts avoidance and detection (Figure 7.F-1)

@startuml CCL CCL actions conflicts, detection and resolution

skinparam Shadowing false

autonumber

skinparam monochrome true

!pragma teoz 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 \nMnS producer \n (CCL metric-value conflicts coordination)" as xCL

Note over CL1, xCL: Actor-CCL, other-CCLs and CoordinationEntity’s capability for CCL actions coordination \nare instantiated and configured as needed

CL1 -> xCL:

& CL2 -> xCL: Register scopes (incl. metrics)of interest

xCL -> xCL: Evaluate desired metrics \n& outcomes for conflict

xCL -> xCL: If potential conflict, revise \n metrics & planned outcomes

xCL -> CL1:

& xCL -> CL2: send new/ revised \nmetrics and outcomes.

Alt other-CCLs have executed

CL2 -> xCL: Register executedAction and \ntolerance on parameters & metrics.

End

CL1 -> CL1: Derive desired actions plan

CL1 -> xCL: Register desired actions plan

xCL -> xCL: evaluate performance \nimprovement & reliability/\nconfidence of actions plans

xCL -> CL1: notify action accepted or not (& failed criteria if check has failed)

alt

xCL -> CL1: Notify if CCL is trusted

end

CL1 -> CL1: if action accepted, execute onto network

CL1 -> xCL: Register executedAction, planned impact and desired outcomes

xCL -> xCL: compare impact scopes \nto detect metric correlated \noscillations

CL1 -> CL1: evaluate own desired metrics for ping-pong/ flipflops

xCL -> CL1: If correlated oscillations, inform actor CCLs of the correlated oscillations (i.e. potential conflict)

CL1 -> xCL: If flipflop, inform of flipflop \n(i.e, a potential conflict)

xCL -> xCL: derives new CCL config \n(e.g., desired metrics priorities)

xCL -> CL1: notifies new config to CCL inclufing new metrics or outcomes.

@enduml

**PlantUML source code for Figure 7.D-1 CCL coordination to avoid and detect CCL metric-value conflicts**