**3GPP TSG-SA5 Meeting #156 *S5-244877***

**Maastricht, Netherlands, 19th Aug 2024 - 23rd Aug 2024**

|  |
| --- |
| *CR-Form-v12.3* |
| **CHANGE REQUEST** |
|  |
|  | **28.105** | **CR** | **0155** | **rev** | **1** | **Current version:** | **18.4.0** |  |
|  |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form: comprehensive instructions can be found at* [*http://www.3gpp.org/Change-Requests*](http://www.3gpp.org/Change-Requests)*.* |
|  |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***Proposed change affects:*** | UICC apps |  | ME |  | Radio Access Network | **X** | Core Network | **X** |

|  |
| --- |
|  |
| ***Title:***  | Rel-18 CR TS 28.105 correct the description of ML model lifecycle and ML entity |
|  |  |
| ***Source to WG:*** | Huawei |
| ***Source to TSG:*** | S5 |
|  |  |
| ***Work item code:*** | AIML\_MGT |  | ***Date:*** | 2024-07-16 |
|  |  |  |  |  |
| ***Category:*** | **F** |  | ***Release:*** | Rel-18 |
|  | *Use one of the following categories:****F*** *(correction)****A*** *(mirror corresponding to a change in an earlier release)****B*** *(addition of feature),* ***C*** *(functional modification of feature)****D*** *(editorial modification)*Detailed explanations of the above categories canbe found in 3GPP [TR 21.900](http://www.3gpp.org/ftp/Specs/html-info/21900.htm). | *Use one of the following releases:Rel-8 (Release 8)Rel-9 (Release 9)Rel-10 (Release 10)Rel-11 (Release 11)…Rel-15 (Release 15)Rel-16 (Release 16)Rel-17 (Release 17)Rel-18 (Release 18)* |
|  |  |
| ***Reason for change:*** | According to the conclusions of the previous meetings, the word "workflow" needs to be changed to ML model lifecycle and the word "ML entity" needs to be changed to ML model. |
|  |  |
| ***Summary of change:*** | Correct the description of ML model lifecycle.Correct the “ML entity” to “ML model” |
|  |  |
| ***Consequences if not approved:*** | The description will make confusion. |
|  |  |
| ***Clauses affected:*** | 4.a.0, 6.1, 6.5.5.1, 7.3a.1.2.2.1, 7.3a.3.2.2.3, A.7, A.10, A.11 |
|  |  |
|  | **Y** | **N** |  |  |
| ***Other specs*** |  | **X** |  Other core specifications  | TS/TR ... CR ...  |
| ***affected:*** |  | **X** |  Test specifications | TS/TR ... CR ...  |
| ***(show related CRs)*** |  | **X** |  O&M Specifications | TS/TR ... CR ...  |
|  |  |
| ***Other comments:*** |  |
|  |  |
| ***This CR's revision history:*** |  |

|  |
| --- |
| **1st modified section** |

# 4a AI/ML management functionality and service framework

## 4a.0 ML model lifecycle

AI/ML techniques are widely used in 5GS (including 5GC, NG-RAN, and management system), the generic AI/ML operational workflow shown in Figure 4a.0-1, highlights the main steps of an ML model lifecycle .



Figure 4a.0-1: ML model lifecycle

The ML model lifecycle includes training, emulation, deployment, and inference. These steps are briefly described below:

**- ML model training:** training, including initial training and re-training, of an ML model or a group of ML models. It also includes validation of the ML model to evaluate the performance when the ML model performs on the training data and validation data. If the validation result does not meet the expectation (e.g., the variance is not acceptable), the ML model needs to be re-trained.

**- ML model testing:** testing of a validated ML model to evaluate the performance of the trained ML model when it performs on testing data. If the testing result meets the expectations, the ML model may proceed to the next step If the testing result does not meet the expectations, the ML model needs to be re-trained.

**- AI/ML inference emulation:** running an ML model for inference in an emulation environment. The purpose is to evaluate the inference performance of the ML model in the emulation environment prior to applying it to the target network or system.

NOTE: The AI/ML inference emulation is considered optional and can be skipped in the ML model lifecycle.

**- ML model deployment:** ML model deployment includes the ML model loading process (a.k.a. a sequence of atomic actions) to make a trained ML model available for use at the target AI/ML inference function.

ML model deployment may not be needed in some cases, for example when the training function and inference function are co-located.

**- AI/ML inference:** performing inference using a trained ML model by the AI/ML inference function. The AI/ML inference may also trigger model re-training or update based on e.g., performance monitoring and evaluation.

|  |
| --- |
| **Next modified sections** |

# 6 AI/ML management use cases and requirements

## 6.1 ML model lifecycle management capabilities

Each operational step in the ML model lifecycle (see clause 4a.0.1) is supported by one or more AI/ML management capabilities as listed below.

**Management capabilities for ML model training**

**- ML model training management**: allowing the MnS consumer to request the ML model training, consume and control the producer-initiated training, and manage the ML model training/re-training process. The training management capability may include training performance management and setting a policy for the producer-initiated ML model training.

**-** ML model training capability also includes validation to evaluate the performance of the ML model when performing on the validation data, and to identify the variance of the performance on the training and validation data. If the variance is not acceptable, the ML model would need to be re-trained before being made available for the next step in the ML model lifecycle (e.g., ML model testing).

**Management capabilities for ML testing**

**- ML model testing management**: allowing the MnS consumer to request the ML model testing, and to receive the testing results for a trained ML model. It may also include capabilities for selecting the specific performance metrics to be used or reported by the ML testing function. MnS consumer may also be allowed to trigger ML model re-training based on the ML model testing performance results.

**Management capabilities for AI/ML inference emulation:**

* **AI/ML inference emulation:** a capability allowing an MnS consumer to request an ML inference emulation for a specific ML model or models (after the training, validation, and testing) to evaluate the inference performance in an emulation environment prior to applying it to the target network or system.

**Management capabilities for ML model** **deployment:**

**- ML model loading management**: allowing the MnS consumer to trigger, control and/or monitor the ML model loading process.

**Management capabilities for AI/ML inference:**

**- AI/ML inference management:** allowing an MnS consumer to control the inference, i.e., activate/deactivate the inference function and/or ML model/models, configure the allowed ranges of the inference output parameters. The capabilities also allow the MnS consumer to monitor and evaluate the inference performance and when needed trigger an update of an ML model or an AI/ML inference function.

The use cases and corresponding requirements for AI/ML management capabilities are specified in the following clauses.

|  |
| --- |
| **Next modified sections** |

### 6.5.5 Executing AI/ML Inference

#### 6.5.5.1 Description

Different functionalities in the network or management domains may utilize AI/ML inference techniques to conduct their tasks under different contexts. Depending on the contexts, the outcome of the ML model at inference might be different. The history of such inference outcome and the corresponding context within which they were taken may be of interest to different consumers.

|  |
| --- |
| **Next modified sections** |

##### 7.3a.1.2.2 MLTrainingRequest

###### 7.3a.1.2.2.1 Definition

The IOC MLTrainingRequest represents the ML model training request that is trigered by the ML training MnS consumer.

To trigger the ML model training process, ML training MnS consumer needs create MLTrainingRequest object instances on the ML training MnS producer. The MLTrainingRequest MOI is contained under one MLTrainingFunction MOI.

The MLTrainingRequest MOI may represent the request for initial ML model training or re-training. For ML model re-training, the MLTrainingRequest is associated to one MLModel for re-training a single ML model, or associated to one MLModelCoordinationGroup.

The MLTrainingRequest may have a source to identify its origin, which may be used to prioritize the training resources for different sources. The sources may be for example the network functions, operator roles, or other functional differentiations.

Each MLTrainingRequest indicates the expectedRunTimeContext that describes the specific conditions for which the MLModel should be trained.

In case the request is accepted, the ML training MnS producer decides when to start the ML model training based on consumer requirements. Once the MnS producer decides to start the training based on the request, the ML training MnS producer instantiates one or more MLTrainingProcess MOI(s) that are responsible to perform the followings:

- collects (more) data for training, if the training data are not available or the data are available but not sufficient for the training;

- prepares and selects the required training data, with consideration of the consumer’s request provided candidate training data if any. The ML training MnS producer may examine the consumer's provided candidate training data and select none, some or all of them for training. In addition, the ML training MnS producer may select some other training data that are available in order to meet the consumer’s requirements for the ML model training;

- trains the MLModel using the selected and prepared training data.

The MLTrainingRequest may have a requestStatus field to represent the status of the specific MLTrainingRequest:

- The attribute values are "NOT\_STARTED", " IN\_PROGRESS", "SUSPENDED", "FINISHED", and "CANCELLED".

- When value turns to " IN\_PROGRESS", the ML training MnS producer instantiates one or more MLTrainingProcess MOI(s) representing the training process(es) being performed per the request and notifies the MLT MnS consumer(s) who subscribed to the notification.

When all of the training process associated to this request are completed, the value turns to "FINISHED".

The ML training MnS prodcuer shall delete the corresponding MLTrainingRequest instance in case of the status value turns to "FINISHED" or "CANCELLED". The MnS producer may notify the status of the request to MnS consumer after deleting MLTrainingRequest instance.

|  |
| --- |
| **Next modified sections** |

##### 7.3a.3.2.2 MLModelLoadingPolicy

###### 7.3a.3.2.2.1 Definition

This IOC represents the ML model loading policy set by the MnS consumer to the producer for loading an ML model to the target inference function(s).

To specify ML model loading policy for one or muiltiply ML models, MnS consumer needs to create MLModelLoadingPolicy object instances.

To remove ML model loading policy for one or muiltiply ML models, MnS consumer needs to delete MLModelLoadingPolicy object instances.

This IOC is used for the MnS consumer to set the conditions for the producer-initated ML model loading. The MnS producer is only allowed to load the ML model when all of the conditions are met.

###### 7.3a.3.2.2.2 Attributes

Table 7.3a.3.2.2.2-1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Attribute name | Support Qualifier | isReadable  | isWritable | isInvariant | isNotifyable |
| aIMLInferenceName | CM | T | T | F | T |
| policyForLoading | M | T | T | F | T |
| **Attribute related to role** |  |  |  |  |  |
| mLModelRef | CM | T | F | F | F |

###### 7.3a.3.2.2.3 Attribute constraints

Table 7.3a.3.2.2.3-1

|  |  |
| --- | --- |
| Name | Definition |
| aIMLInferenceName Support Qualifier | Condition: The ML model loading policy is related to an initially trained ML model. |
| mLModelRef Support Qualifier | Condition: The ML model loading policy is related to a re-trained ML model. |

|  |
| --- |
| **Next modified sections** |

# A.7 PlantUML code for Figure 7.3a.1.1.2-2: Inheritance Hierarchy for ML model testing related NRMs

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

'skinparam maxMessageSize 250

class Top <<InformationObjectClass>>

class ManagedFunction <<InformationObjectClass>>

class MLTestingFunction <<InformationObjectClass>>

class MLTestingRequest <<InformationObjectClass>>

class MLTestingReport <<InformationObjectClass>>

ManagedFunction <|-- MLTestingFunction

Top <|-- MLTestingRequest

Top <|-- MLTestingReport

@enduml

|  |
| --- |
| **Next modified sections** |

# A.10 PlantUML code for Figure 7.3a.3.1.1-1: NRM fragment for ML model loading

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

'skinparam maxMessageSize 250

class AIMLInferenceFunction <<InformationObjectClass>>

class MLModel <<InformationObjectClass>>

class MLModelLoadingRequest <<InformationObjectClass>>

class MLModelLoadingPolicy <<InformationObjectClass>>

class MLModelLoadingProcess <<InformationObjectClass>>

AIMLInferenceFunction"1" \*-- "\*" MLModelLoadingRequest : <<names>>

AIMLInferenceFunction "1" \*-- "\*" MLModelLoadingPolicy : <<names>>

AIMLInferenceFunction "1" \*-- "\*" MLModelLoadingProcess : <<names>>

MLModelLoadingRequest "1" <-r- "\*" MLModelLoadingProcess

MLModelLoadingProcess "\*" -r-> "1" MLModelLoadingPolicy

MLModelLoadingRequest "1" --> "1" MLModel

MLModelLoadingProcess "1" --> "1" MLModel

AIMLInferenceFunction "1" \*-- "\*" MLModel : <<names>>

(MLModelLoadingProcess, MLModelLoadingRequest) ... (MLModelLoadingProcess, MLModelLoadingPolicy) : {xor}

@enduml

# A.11 PlantUML code for Figure 7.3a.3.1.2-1: Inheritance Hierarchy for ML model loading related NRMs

@startuml

skinparam ClassStereotypeFontStyle normal

skinparam ClassBackgroundColor White

skinparam shadowing false

skinparam monochrome true

hide members

hide circle

'skinparam maxMessageSize 250

class Top <<InformationObjectClass>>

class MLModelLoadingRequest <<InformationObjectClass>>

class MLModelLoadingPolicy <<InformationObjectClass>>

class MLModelLoadingProcess <<InformationObjectClass>>

Top <|-- MLModelLoadingRequest

Top <|-- MLModelLoadingPolicy

Top <|-- MLModelLoadingProcess

@enduml

|  |
| --- |
| **End of modified sections** |