**3GPP TSG- Meeting #**

**, , -**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *CR-Form-v12.2* | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | |
|  | | | | | | | | |
|  |  | **CR** |  | **rev** |  | **Current version:** |  |  |
|  | | | | | | | | |
| *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:*** |  | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Source to WG:*** |  | | | | | | | | | |
| ***Source to TSG:*** | S5 | | | | | | | | | |
|  |  | | | | | | | | | |
| ***Work item code:*** |  | | | | |  | ***Date:*** | | |  |
|  |  | | | |  | |  | | |  |
| ***Category:*** | F |  | | | | | ***Release:*** | | |  |
|  | *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 can be 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-16 (Release 16) Rel-17 (Release 17) Rel-18 (Release 18) Rel-19 (Release 19)* | |
|  |  | | | | | | | | | |
| ***Reason for change:*** | | There is a lot of discussion about how to handle asynchonous and multi-step operations. We want to provide guidelines for any future operations assuring that such operations will be handled in a uniform mannner.  The contribution doesn't describe new functionality, just how existing functionality should be used. It provides hitherto missing descriptions about existing functionality.  All operations shall receive a response in a reasonably short time. This is needed in order for the MnS consumer to be able to distinguish between a slow MnS provider and a failed operation.  If an operation involves multiple steps or potentially takes a longer time the producer needs to send an initial response and provide information about the progress and final result of the operation. In this case all we need is a processMonitor and potentially some additional state attributes to monitor/indicate the change and the end-result of the running operation.  As an issue ***independent from the above***, a CRUD operation may have a side effect of creating additional NRM changes. In some cases it is not known a priori how many MOIs or attributes will be updated during the operation. As the normal CRUD operations are expected to create/delete/modify what was requested (or modify nothing if they fail) it is best to use an auxiliary xxxJob IOC to handle uncertain NRM updates as a side effect of the creation/modification of the xxxJob MOI. An example would be allocating a new network slice.This may or may not result in creating a new NetworkSlice MOI. It is not know a priori which will happen.  This way the direct CRUD udate of the xxxJob MOI will follow normal CRUD operation patterns, and the uncertain NRM updates can be handled as side-effects that can be monitored separately using either additional state attributes on the xxxJob IOC or by data change notifications on the additional NRM updates.  While information about these uncertain updates could be added to some network resource IOC (like NetworkSlice) it is better to place this set of administrative information into a separate xxxJob IOC.  It is better to separate IOCs that model network resources and IOCs that model some administrative/OAM task. 3GPP already uses such administrative IOCs e.g. PerfmetricJob, ThresholdMonitor.  Separating such administrative tasks makes it easier to clean-up the administrative information once its no longer needed. E.g. instead of removing some NetworkSlice MOIs based on internal parameters just checking the list of xxxJob MOIs to find out which is not needed. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Summary of change:*** | | Add descripiton on how to handle simple, multi-step and non-trivial operations. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Consequences if not approved:*** | | Misunderstandings about how to handle such operations; different handling patterns for different operations.. | | | | | | | | |
|  | |  | | | | | | | | |
| ***Clauses affected:*** | | 11.a, 11.a.1, 11.a.2, Annex C | | | | | | | | |
|  | |  | | | | | | | | |
|  | | **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:*** | |  | | | | | | | | |

***First change***

11.a General handling of provisioning operations

11.a.1 Rules for all operations

All operations shall receive a response in a reasonably short time. This is needed in order for the MnS consumer to be able to distinguish between a slow responding MnS provider and a connectivity or processing error.

While the MnS producer cannot control delays due to the connecting network, if the processing of a request is even potentially time consuming, some response should be provided before the final results are available.

Even though the exact length of "Short time" cannot be defined (as it is dependent on many factors), the concept and proper handling of slow requests is still important. It is the responsibility of designers/implementers of each use-case to decide whether it is potentially slow or fast.

11.a.2 Rules for CRUD iniated procedures

Provisioning (CRUD) operations may (or may not) initate updates to background resources, thus the complete procedure (actions taken) may consist of two parts: first updating the NRM datastore ( e.g. creating an MOI) then executing a background task (e.g. initalizing a new virtual machine for a new network function).

In some cases it is not a priori known what NRM data updates will result from a single CRUD operation.

Due to the above CRUD initiated procedures should be handled based on two independent properties. While a CRUD operation (updating the NRM datastore) itself is always synchronous and not dependent on these properties; associated activities may be more complex.

A procedure initiated by a CRUD operation may be seen as single step or multi-step from a consumer point of view. While its invocation is a single step, its execution and responses may involve multiple steps.

- **Single step procedure**: The consumer sees these as a single step even if there are multiple background resources affected. The operation is initiated by a single request and sends back a single response message. There are no intermediate stage results although the end-result might indicate which parts of the operation succeeded or failed. Once the operation succeeded both the NRM changes and any background resource updates (that need a change due to the management operation) are done. These operations are assumed to succeed or fail in a "short time"; see clause 11.a.1.

- **Multi-step procedure**: The consumer sees multiple separate steps of the execution. While the NRM updates should be excuted in a single first step and fast, needed background resource updates might take multiple steps and a longer time.

The procedure is initiated by a single CRUD request. The MnS producer shall update the NRM (e.g. create a requested MOI) then send back a prompt response message. This response indicates that the requested NRM changes have been executed, and that any needed background resource updates were initiated and the input receive in the CRUD request is acceptable. At this point the created network resources are not ensured to be ready for use. (Note, the operation may fail completely e.g. if the input data is unacceptable.)

The multiple steps may simply indicate the percentage of completion of the procedure (0-100 %) or they may indicate distinct, named execution stages e.g., virtual-machine-created, virtual-machine-configured, virtual-machine-ready.

A multi-step procedure allows better visibility of the progress of background resource updates and may provide the consumer the possibility to cancel the operation.

The MnS provider shall provide attributes/MOIs that allow the MnS consumer to monitor the progress of the background resource updates. The dataType "processMonitor" (see [11]) shall be used to facilitate this. It is allowed to specify that some optional-to-support subparts of the processMonitor are not used in a specific use-case. If needed additional state attributes shall be added to indicate the progress and result state of updates to the background resources. When adding additional attributes the known state handling patterns shall be preferred (operationalState, availabilityStatus,usageState, etc.) It is the responsibility of the MnS consumer to monitor the background resource updates via notifications or repeated read operations.

Procedures that may take a longer time should be defined as multi-step operations"; see clause 11.a.1.

An operation can be a simple or a non-trivial operation.

- **Simple operations** request a fixed, a-priori know set of NRM changes (create/update/delete) e.g. setting some MOIs and attributes.

- **Non-trivial operations** are initiated by a CRUD operation, but may or may not result in changes to additional NRM elements (MOIs or attributes). These additional NRM updates are not possible to know before the operation is invoked.

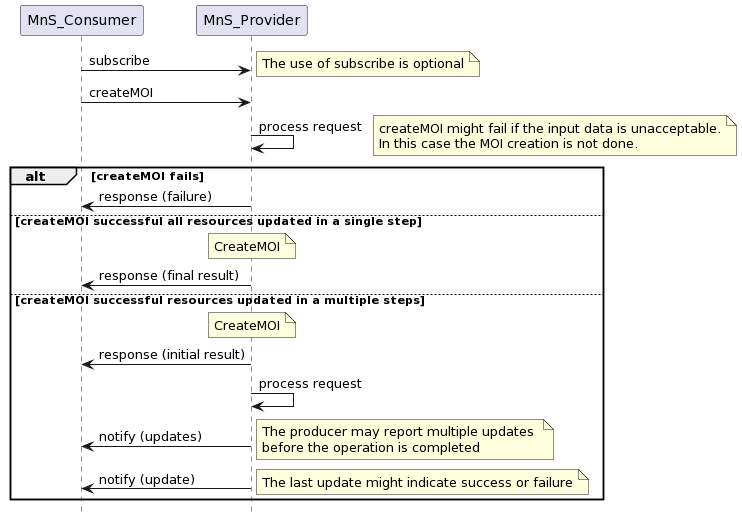
If an operation is non-trivial an auxiliary "xxxJob" IOC should be defined whose sole purpose is to facilitate invoking the operation. (xxxJob is the recommended naming pattern, but other names can also be used.)

If the operation also initiates a multi-step procedure, processMonitor should be added to the "xxxJob" IOC. Additional state attributes representing the state of the process should be added to the "xxxJob" IOC. If an attribute represents the state of a resource that is modeled by its own IOC the state attribute should be added to this "resource-IOC".

**Table 11.a.2-1: IOCs, attributes for non-trvial and/or multi-step operations**

|  |  |  |
| --- | --- | --- |
|  | Simple operation | Non-trivial operation |
| Single-step | - no special handling | - Define an "xxxJob" IOC |
| Multi-step | Define in the IOC created/updated  - processMonitor attribute  - Additional state attributes may be defined  Might allow cancelation | - Define an "xxxJob" IOC  - Use processMonitor attribute on "xxxJob"  - Additional state variables may be added to "xxxJob" or a "resourecIOC"  Might allow cancelation |

The following diagram is an example of a multi-step procedure initiated by a createMOI request.



**Figure 11.a.2-1. Multi-step procedure**

***Next change***

# Annex C (informative): PlantUML source for diagrams

@startuml

'Created using https://www.planttext.com/

'Figure 11.a.2-1. Multi-step procedure

hide footbox

MnS\_Consumer -> MnS\_Provider : subscribe

note right: The use of subscribe is optional

MnS\_Consumer -> MnS\_Provider : createMOI

MnS\_Provider -> MnS\_Provider : process request

note right

createMOI might fail if the input data is unacceptable.

In this case the MOI creation is not done.

end note

alt createMOI fails

MnS\_Consumer <- MnS\_Provider : response (failure)

else createMOI successful all resources updated in a single step

note over MnS\_Provider: CreateMOI

MnS\_Consumer <- MnS\_Provider : response (final result)

else createMOI successful resources updated in a multiple steps

note over MnS\_Provider: CreateMOI

MnS\_Consumer <- MnS\_Provider : response (initial result)

MnS\_Provider -> MnS\_Provider : process request

MnS\_Consumer <- MnS\_Provider : notify (updates)

note right

The producer may report multiple updates

before the operation is completed

end note

MnS\_Consumer <- MnS\_Provider : notify (update)

note right: The last update might indicate success or failure

end

@enduml

***End of changes***