



ESC ETSI API

5.5.0.86 OAS3

[/esc-etsi-api](#)

Documentation :

ETSI-MANO REST Northbound API

This REST API is another programmatic interface to ESC that uses a REST architecture. The API accepts and returns HTTP or HTTPS messages that contain JavaScript Object Notation (JSON).

It is the payloads for these request/responses that are defined by the European Telecommunications Standards Institute (ETSI), specifically around Management and Orchestration (MANO). It contains its own data model, designed around the ETSI-MANO specification (ETSI GS NFV-SOL 003 V2.4.1), that abstracts away from the ESC core data model.

This initial implementation of the ETSI-MANO standards for NFV is to address the Or-Vnfm reference point, i.e. the interface between the Network Function Virtualisation Orchestrator (NFVO) and the Virtual Network Function Manager (VNFM).

The Or-Vnfm reference point details the interactions to onboard ETSI-compliant VNF packages, manage resources, and VNF lifecycle management (LCM) operations.

During the lifespan of a VNF Instance, it moves between INSTANTIATED and NOT_INSTANTIATED states, whereas operations that perform LCM operations have a more complex state machine, as per the diagram below.

The ETSI-MANO specification considers provisioning of many components of a network service outside the remit of the VNFM, namely:

- Tenants
- Images
- Flavours
- External Networks/Virtual Link
- Externally Managed Internal Virtual Link
- Subnets

This means that LCM operations on an instance of a VNF submitted to the ETSI-MANO REST API expect these resources to be created out-of-band (OOB) as far as the VNFM is concerned. It is likely that these resources are created via the NFVO, either at the time of onboarding the VNF package or onboarding the tenant, and will be represented by VIM (Virtual Infrastructure Manager) identifiers in the request to ESC.

Managing Resources

Managing Resources via the ETSI-MANO API The ETSI-MANO API communicates with NFVO for lifecycle management. A configuration template, the Virtual Network Function Descriptor (VNFD) file describes the deployment parameters and operational behaviors of a VNF type. The VNFD is used in the process

of deploying a VNF and managing the lifecycle of a VNF instance. The flow of operations to deploy a VNF instance is:

1. Create VNF Identifier
2. Instantiate VNF The flow of operations to fully undeploy (and release resources used by a VNF instance) is:
3. Terminate VNF
4. Delete VNF Identifier

The other LCM operations are applicable once the VNF has been instantiated, except from Query which is applicable at any time since it does not modify the VNF.

LCM Operations

Here is an overview of the operations that can affect a VNF instance.

- **Create VNF Identifier:** Generate a new VNF Instance Id (a universally unique identifier) that is subsequently used as a handle to reference the instance upon which to execute further operations.
- **Instantiate VNF:** Deploy a new VNF instance in the VIM. The Instantiate request will contain instance-specific values and this, coupled with the VNFD and the Grant information will provide all the information required by the VIM to deploy the VNF. The VNFD is retrieved from the NFVO as part of this call flow which provides the resource requirements for the VNF to be instantiated. This data set is then further supplemented by requesting permission from the NFVO to continue with the request which returns Grant information that converts some of these resource requirements to actual resources that are reserved in the VIM.
- **Operate VNF:** Allow a VNF instance to be started or stopped. The resources are not released or changed, but the VNF instance in the VIM is toggled between these two states.
- **Query VNF:** Query one or more VNF instances known to ESC. This is a specific REST endpoint that can be filtered to find specific instances. In this initial release, the instances can be filtered by the VNF Instance Id.
- **Scale VNF:** Scale VNF instance incrementally.
- **Scale VNF to Level:** Scale VNF instance to target level.
- **Terminate VNF:** Undeploy the VNF instance in the VIM. The resources themselves remain reserved for the VNF instance, however the VNF itself is undeployed.
- **Delete VNF Identifier:** The resources are fully released in the VIM and in ESC and the associated VNF instance identifier is also released.
- **Heal VNF:** Recover a VNF.
- **Modify VNF:** Modify a VNF resource.
- **Change External VNF Connectivity:** Change the deployment flavour of a VNF instance.
- **Change VNF Flavour:** Change the deployment flavour of a VNF instance.

Authentication: At the time of publication, only Basic Authentication is supported using the ETSI Swagger API. Cisco ESC does support OAUTH 2.0 authentication. Please see the user guide for details.

Attribute Selectors: REST endpoints which are used to query multiple results support attribute selectors (see the ETSI-MANO specification for more details).

- **all_fields:** This URI query parameter requests that all complex attributes are included in the response, including those suppressed by `exclude_default`. It is inverse to the "exclude_default" parameter.
- **fields:** This URI query parameter requests that only the listed complex attributes are included in the response.
- **exclude_fields:** This URI query parameter requests that the listed complex attributes are excluded from the response.

- **exclude_default:** Presence of this URI query parameter requests that a default set of complex attributes shall be excluded from the response.

If no attribute selector is supplied then the default behaviour is the same as `exclude_default` (this can be changed to `all_fields` by setting the property `attribute.selector.default.all_fields` to true).

Server

Or-Vnfm vnf_instances

This resource represents VNF instances for the Or-Vnfm Reference Point. The client can use this resource to create individual VNF instance resources, and to query VNF instances.



POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/instantiate` Instantiate a VNF

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_ext_conn` Change the external VNF connectivity

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/terminate` Terminate a VNF Instance

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/heal` Heal a VNF Instance

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale` Scale a VNF Instance

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale_to_level` Scale a VNF Instance to Level

PATCH `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Modify an individual VNF Instance

GET `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Read an individual VNF resource

DELETE `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Delete a VNF instance resource

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/operate` Operate a VNF Instance

POST `/or_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_flavour` Change the VNF Flavour

POST `/or_vnfm/vnflcm/v1/vnf_instances` Create a VNF Instance resource

GET `/or_vnfm/vnflcm/v1/vnf_instances` Query multiple VNF instances

Ve-Vnfm vnf_instances

This resource represents VNF instances for the Ve-Vnfm Reference Point. The client can use this resource to create individual VNF instance resources, and to query VNF instances.



POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale` Scale a VNF Instance

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/terminate` Terminate a VNF Instance

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_ext_conn` Change the external VNF connectivity

POST `/ve_vnfm/vnflcm/v1/vnf_instances` Create a VNF Instance resource

GET `/ve_vnfm/vnflcm/v1/vnf_instances` Query multiple VNF instances

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/instantiate` Instantiate a VNF

PATCH `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Modify an individual VNF Instance

GET `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Read an individual VNF resource

DELETE `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}` Delete a VNF instance resource

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/heal` Heal a VNF Instance

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale_to_level` Scale a VNF Instance to Level

POST `/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/operate` Operate a VNF Instance

POST

`/ve_vnfm/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_flavour`

Change the VNF Flavour

vnf_instances

This resource represents VNF instances. The client can use this resource to create individual VNF instance resources, and to query VNF instances.



POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_ext_conn`

Change the external VNF connectivity

POST

`/vnflcm/v1/vnf_instances` Create a VNF Instance resource

GET

`/vnflcm/v1/vnf_instances` Query multiple VNF instances

PATCH

`/vnflcm/v1/vnf_instances/{vnfInstanceId}` Modify an individual VNF Instance

GET

`/vnflcm/v1/vnf_instances/{vnfInstanceId}` Read an individual VNF resource

DELETE

`/vnflcm/v1/vnf_instances/{vnfInstanceId}` Delete a VNF instance resource

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale_to_level`

Scale a VNF Instance to Level

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/operate` Operate a VNF Instance

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/change_flavour` Change the VNF Flavour

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/scale` Scale a VNF Instance

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/terminate` Terminate a VNF Instance

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/heal` Heal a VNF Instance

POST

`/vnflcm/v1/vnf_instances/{vnfInstanceId}/instantiate` Instantiate a VNF

vnf_instances extensions

This resource represents extensions to VNF instances.



POST /or_vnfm/vnflcm/v1/ext/vnf_instances/{vnfInstanceId}/monitoring/migrate Migrate monitoring for a VNF

POST /ve_vnfm/vnflcm/v1/ext/vnf_instances/{vnfInstanceId}/monitoring/operations Enable/disable monitoring VNF/particular VMs

GET /vnflcm/v1/ext/vnf_instances/{vnfInstanceId}/deployment Extension endpoint to get deployment descriptor

POST /or_vnfm/vnflcm/v1/ext/vnf_instances/{vnfInstanceId}/monitoring/operations Enable/disable monitoring for VNF/particular VMs

POST /ve_vnfm/vnflcm/v1/ext/vnf_instances/{vnfInstanceId}/monitoring/migrate Migrate monitoring for a VNF

Or-Vnfm vnf_lcm_op_occs

This resource represents VNF lifecycle management operation occurrences for the Or-Vnfm Reference Point. The client can use this resource to query status information about multiple VNF lifecycle management operation occurrences.



POST /or_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOpOccId}/rollback Rollback a VNF lifecycle management operation occurrence

POST /or_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOpOccId}/retry Retry a VNF lifecycle management operation occurrence

POST /or_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOpOccId}/cancel Cancel a VNF lifecycle management operation occurrence

GET /or_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOpOccId} Read an individual VNF lifecycle management operation occurrence

GET /or_vnfm/vnflcm/v1/vnf_lcm_op_occs Query multiple VNF lifecycle management operation occurrences

POST /or_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOpOccId}/fail Mark a VNF lifecycle management operation

Ve-Vnfm vnf_lcm_op_occs

This resource represents VNF lifecycle management operation occurrences for the Ve-Vnfm Rreference Point. The client can use this resource to query status information about multiple VNF lifecycle management operation occurrences.



POST /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/cancel Cancel a VNF lifecycle management operation occurrence

POST /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/rollback Rollback a VNF lifecycle management operation occurrence

GET /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs Query multiple VNF lifecycle management operation occurrences

POST /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/fail Mark a VNF lifecycle management operation occurrence as failed

GET /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId} Read an individual VNF lifecycle management operation occurrence

POST /ve_vnfm/vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/retry Retry a VNF lifecycle management operation occurrence

vnf_lcm_op_occs

This resource represents VNF lifecycle management operation occurrences. The client can use this resource to query status information about multiple VNF lifecycle management operation occurrences.



POST /vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/cancel Cancel a VNF lifecycle management operation occurrence

POST /vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/fail Mark a VNF lifecycle management operation occurrence as failed


POST /vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId}/retry Retry a VNF lifecycle management operation occurrence

GET /vnflcm/v1/vnf_lcm_op_occs/{vnfLcmOp0ccId} Read an individual VNF lifecycle management operation occurrence

POST `/vnflcm/v1/vnf_lcm_op_occs/{vnflcmOpOccId}/rollback` Rollback a VNF lifecycle management operation occurrence

GET `/vnflcm/v1/vnf_lcm_op_occs` Query multiple VNF lifecycle management operation occurrences

Or-Vnfm lcn_subscriptions

This resource represents VNF lifecycle management notification subscriptions for the Or-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF lifecycle management, and to query its subscriptions. 


GET `/or_vnfm/vnflcm/v1/subscriptions/{subscriptionId}` Read an individual VNF lifecycle management subscription resource

DELETE `/or_vnfm/vnflcm/v1/subscriptions/{subscriptionId}` Terminate an individual VNF lifecycle management subscription

POST `/or_vnfm/vnflcm/v1/subscriptions` Create a new subscription

GET `/or_vnfm/vnflcm/v1/subscriptions` Queries the list of active VNF lifecycle management subscriptions

Ve-Vnfm lcn_subscriptions

This resource represents VNF lifecycle management notification subscriptions for the Ve-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF lifecycle management, and to query its subscriptions. 

POST `/ve_vnfm/vnflcm/v1/subscriptions` Create a new subscription

GET `/ve_vnfm/vnflcm/v1/subscriptions` Queries the list of active VNF lifecycle management subscriptions

GET `/ve_vnfm/vnflcm/v1/subscriptions/{subscriptionId}` Read an individual VNF lifecycle management subscription resource

DELETE `/ve_vnfm/vnflcm/v1/subscriptions/{subscriptionId}` Terminate an individual VNF lifecycle management subscription

lccn_subscriptions

This resource represents VNF lifecycle management notification subscriptions. The client can use this resource to subscribe to notifications related to VNF lifecycle management, and to query its subscriptions.



GET `/vnflcm/v1/subscriptions/{subscriptionId}` Read an individual VNF lifecycle management subscription resource

DELETE `/vnflcm/v1/subscriptions/{subscriptionId}` Terminate an individual VNF lifecycle management subscription

POST `/vnflcm/v1/subscriptions` Create a new subscription

GET `/vnflcm/v1/subscriptions` Queries the list of active VNF lifecycle management subscriptions

Or-Vnfm fm_subscriptions

This resource represents VNF alarm subscriptions for the Or-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF alarms and to query its subscriptions.



POST `/or_vnfm/vnffm/v1/subscriptions` Create a new VNF alarm subscription

GET `/or_vnfm/vnffm/v1/subscriptions` Queries the list of active VNF alarm subscriptions

GET `/or_vnfm/vnffm/v1/subscriptions/{subscriptionId}` Read an individual VNF alarm subscription resource

DELETE `/or_vnfm/vnffm/v1/subscriptions/{subscriptionId}` Terminate an individual VNF alarm subscription

Ve-Vnfm fm_subscriptions

This resource represents VNF alarm subscriptions for the Ve-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF alarms and to query its subscriptions.



POST `/ve_vnfm/vnffm/v1/subscriptions` Create a new VNF alarm subscription

GET `/ve_vnfm/vnffm/v1/subscriptions` Queries the list of active VNF alarm subscriptions

GET /ve_vnfm/vnffm/v1/subscriptions/{subscriptionId} Read an individual VNF alarm subscription resource

DELETE /ve_vnfm/vnffm/v1/subscriptions/{subscriptionId} Terminate an individual VNF alarm subscription

fm_subscriptions

This resource represents VNF alarm subscriptions. The client can use this resource to subscribe to notifications related to VNF alarms and to query its subscriptions.



GET /vnffm/v1/subscriptions/{subscriptionId} Read an individual VNF alarm subscription resource

DELETE /vnffm/v1/subscriptions/{subscriptionId} Terminate an individual VNF alarm subscription

POST /vnffm/v1/subscriptions Create a new VNF alarm subscription

GET /vnffm/v1/subscriptions Queries the list of active VNF alarm subscriptions

Or-Vnfm pm_subscriptions

This resource represents VNF performance subscriptions for the Or-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF performance and to query its subscriptions.



POST /or_vnfm/vnfp/v1/subscriptions Create a new VNF performance subscription

GET /or_vnfm/vnfp/v1/subscriptions Queries the list of active VNF performance subscriptions

GET /or_vnfm/vnfp/v1/subscriptions/{subscriptionId} Read an individual VNF performance subscription resource

DELETE /or_vnfm/vnfp/v1/subscriptions/{subscriptionId} Terminate an individual VNF performance subscription

Ve-Vnfm pm_subscriptions

This resource represents VNF performance subscriptions for the Ve-Vnfm Reference Point. The client can use this resource to subscribe to notifications related to VNF performance and to query its subscriptions.



GET /ve_vnfm/vnfp/v1/subscriptions/{subscriptionId} Read an individual VNF performance subscription resource

DELETE /ve_vnfm/vnfp/v1/subscriptions/{subscriptionId} Terminate an individual VNF performance subscription

POST /ve_vnfm/vnfp/v1/subscriptions Create a new VNF performance subscription

GET /ve_vnfm/vnfp/v1/subscriptions Queries the list of active VNF performance subscriptions

pm_subscriptions

This resource represents VNF performance subscriptions. The client can use this resource to subscribe to notifications related to VNF performance and to query its subscriptions.



POST /vnfp/v1/subscriptions Create a new VNF performance subscription

GET /vnfp/v1/subscriptions Queries the list of active VNF performance subscriptions

GET /vnfp/v1/subscriptions/{subscriptionId} Read an individual VNF performance subscription resource

DELETE /vnfp/v1/subscriptions/{subscriptionId} Terminate an individual VNF performance subscription

Or-Vnfm alarms

These are all the resources and methods provided for the VNF fault management interface for the Or-Vnfm Reference Point.



GET /or_vnfm/vnff/v1/alarms Get all alarm resource

PATCH /or_vnfm/vnff/v1/alarms/{alarmId} This can be used to change the acknowledgement status of an alarm

GET /or_vnfm/vnff/v1/alarms/{alarmId} Get an individual alarm resource

Ve-Vnfm alarms

These are all the resources and methods provided for the VNF fault management interface for the Ve-Vnfm Reference Point.



GET /ve_vnfm/vnff/v1/alarms Get all alarm resource

PATCH /ve_vnfm/vnff/v1/alarms/{alarmId} This can be used to change the acknowledgement status of an alarm

GET /ve_vnfm/vnffm/v1/alarms/{alarmId} Get an individual alarm resource

alarms

These are all the resources and methods provided for the VNF fault management interface. ▼

PATCH /vnffm/v1/alarms/{alarmId} This can be used to change the acknowledgement status of an alarm

GET /vnffm/v1/alarms/{alarmId} Get an individual alarm resource

GET /vnffm/v1/alarms Get all alarm resource

pm_jobs

These are all the resources and methods provided for the VNF Performance Management interface. ▼

POST /vnfpm/v1/pm_jobs Create a PM Job

GET /vnfpm/v1/pm_jobs Query multiple PM Jobs

GET /vnfpm/v1/pm_jobs/{pmJobId} Read an individual PM Job

DELETE /vnfpm/v1/pm_jobs/{pmJobId} Delete a PM Job

POST /vnfpm/v1/ext/pm_jobs/{pmJobId}/reports Extension endpoint to create a Performance Report

GET /vnfpm/v1/pm_jobs/{pmJobId}/reports/{reportId} Read an individual Performance Report

thresholds

These are all the resources and methods provided for the VNF thresholds interface. ▼

POST /vnfpm/v1/thresholds Create a new threshold

GET /vnfpm/v1/thresholds Query the list of thresholds

GET /vnfpm/v1/thresholds/{thresholdId} Read an individual threshold resource

DELETE /vnfpm/v1/thresholds/{thresholdId} Delete an individual threshold

Maintenance Operations

This resource represents ETSI Maintenance Operations



POST /etsi/operationmode/{operationMode} Sets the Operation Mode of ETSI

GET /etsi/operationmode Returns the ETSI Operation Mode

Models



```

FmNotificationsFilter {
  description: This type represents a subscription filter related
                to notifications about VNF faults.

  perceivedSeverities      [...]
  faultyResourceTypes      [...]
  probableCauses           [...]
  vnfInstanceSubscriptionFilter VnfInstanceSubscriptionFilter {...}
  notificationTypes        [...]
  eventTypes                [...]
}

OperateVnfRequestSol2 {
  description: This type represents request parameters for the "Operate VNF"
                operation.

  vnfInstanceId string($uuid)
                Identifier of VNFC instances. Cardinality can be "0" to
                denote that the request applies to the whole VNF and not a
                specific VNFC instance.

  additionalParams KeyValuePairs {...}
  stopType string
                It signals whether forceful or graceful stop is requested.
                Ignored if changeStateTo=STARTED.

                Enum:
                    Array [ 1 ]
  changeStateTo* VnfOperationalStateType string
                Enum:
                    Array [ 2 ]
}

```

```

HealVnfRequestSol2 {
  description:      This type represents request parameters for the "Heal VNF" operation.

  healScript         string
                    Provides link to a script that should be executed as part of the healing action or a set of rules for healing procedure.

  vnfcInstanceId     string($uuid)
                    List of VNFC instances requiring a healing action.

  additionalParams   KeyValuePairs {...}

  cause              string
                    Indicates the reason why a healing procedure is required.
}

```

```

AffectedVirtualStorage {
  description:      This type provides information about added, deleted, modified and temporary virtual storage resources

  changeType*        string
                    Signals the type of change.

                    Enum:
                        Array [ 4 ]

  virtualLinkDescId* string($uuid)
                    Identifier of the related VirtualStorage descriptor in the VNFD.

  id*                 string($uuid)
                    Identifier of the storage instance, identifying the applicable "virtualStorageResourceInfo" entry in the "VnfInstance" data type

  storageResource*   ResourceHandle {...}
}

```

```

EventType string
Enum:
  Array [ 5 ]

```

```

ScaleInfo {
  description:      This type represents the scale level of a VNF instance related to a scaling aspect.

  scaleLevel*         integer($int32)
                    Indicates the scale level. The minimum value shall be 0 and the maximum value shall be <= maxScaleLevel as described in the VNFD.

  aspectId*           string($uuid)
                    Identifier of the scaling aspect
}

```

```

InstantiateVnfRequest {
  description:      This type represents request parameters for the

```

"Instantiate VNF" operation.

```

extManagedVirtualLinks    [...]
flavourId*                 string($uuid)
                           Identifier of the VNF deployment flavour to be
                           instantiated.

instantiationLevelId      string($uuid)
                           Identifier of the instantiation level of the deployment
                           flavour to be instantiated. If not present, the default
                           instantiation level as declared in the VNFD is
                           instantiated.

vimConnectionInfo         [...]
additionalParams           KeyValuePairs  {...}
extVirtualLinks           [...]
localizationLanguage      string
                           Localization language of the VNF to be instantiated.
}

```

```

VnfInfoModificationRequest {
  description:             This type represents attribute modifications for an
                           "Individual VNF instance" resource, i.e. modifications
                           to a resource representation based on the "VnfInstance"
                           data type.

  metadata                KeyValuePairs  {...}
  extensions              KeyValuePairs  {...}
  vimConnectionInfo      [...]
  vnfPkgId               string($uuid)
                           New value of the "vnfPkgId" attribute in "VnfInstance".
                           The value "null" is not permitted.

  vnfConfigurableProperties  KeyValuePairs  {...}
  vnfInstanceName        string
                           New value of the "vnfInstanceName" attribute in
                           "VnfInstance", or "null" to remove the attribute.

  vnfInstanceDescription string
                           New value of the "vnfInstanceDescription" attribute in
                           "VnfInstance", or "null" to remove the attribute.
}

```

```

ChangeExtVnfConnectivityRequestSol2 {
  description:             This type represents request parameters for the "Change
                           external VNF connectivity" operation to modify the external
                           connectivity of a VNF instance.

  additionalParams       KeyValuePairs  {...}
  extVirtualLinks*      [...]
}

```

```

VnfInstanceSol2 {
  description:             This type represents a VNF instance as per Ve-Vnfm
                           Reference Point.
}

```

```

vnfProductName*      string
                    Name to identify the VNF Product. The value is copied
                    from the VNFD.

vnfdVersion*        string
                    Identifies the version of the VNFD. The value is copied
                    from the VNFD.

metadata            KeyValuePairs  {...}

vnfProvider*       string
                    Provider of the VNF and the VNFD. The value is copied
                    from the VNFD.

_links*            {...}

vnfPkgId*          string($uuid)
                    Identifier of information held by the NFVO about the
                    specific VNF package on which the VNF is based. This
                    identifier was allocated by the NFVO.

vnfConfigurableProperties  KeyValuePairs  {...}

vnfdId*            string($uuid)
                    Identifier of the VNFD on which the VNF instance is
                    based.

instantiationState* string
                    The instantiation state of the VNF.

                    Enum:

                    Array [ 2 ]

vnfInstanceDescription  string
                    Human-readable description of the VNF instance.

extensions          KeyValuePairs  {...}

instantiatedVnfInfo    {...}

vnfInstanceName      string
                    Name of the VNF instance.

id*                 string($uuid)
                    Identifier of the VNF instance.

vnfSoftwareVersion*  string
                    Software version of the VNF. The value is copied from
                    the VNFD.
}

```

```

VimConnectionInfo  {
  description:      This type represents parameters needed to connect to a VIM
                    for managing the resources of a VNF instance.

  vimType*         string
                    Discriminator for the different types of the VIM information.

  vimId            string($uuid)
                    The identifier of the VIM instance. This identifier is
                    managed by the NFVO.

  extra            KeyValuePairs  {...}

  interfaceInfo   KeyValuePairs  {...}

  id*              string($uuid)
                    The identifier of the VIM Connection. This identifier is
                    managed by the NFVO.

  accessInfo

```



```

    KeyValuePairs    {...}
}

CancelModeType    string
Enum:
    Array [ 2 ]

TerminateVnfRequest {
    description:          This type represents request parameters for the
                           "Terminate VNF" operation.

    gracefulTerminationTimeout integer($int32)
                           This attribute is only applicable in case of graceful
                           termination. It defines the time to wait for the VNF to
                           be taken out of service before shutting down the VNF
                           and releasing the resources. The unit is seconds.

    additionalParams    KeyValuePairs    {...}
    terminationType*    string
                           Indicates whether forceful or graceful termination is
                           requested.

                           Enum:
                               Array [ 2 ]
}

CreatePmJobRequest {
    description:          This type represents a request to create a PM job

    objectInstanceIds*   [...]
    criteria*            PmJobCriteria    {...}
}

LifecycleChangeNotificationsFilter {
    description:          This type represents a subscription filter related
                           to notifications about VNF lifecycle changes

    operationStates      [...]
    vnfInstanceSubscriptionFilter VnfInstanceSubscriptionFilter    {...}
    notificationTypes    [...]
    operationTypes       [...]
}

LccnLinks {
    description:          This type represents the links to resources that a
                           notification can contain

    subscription*        Link    {...}
    vnfLcmOpOcc          Link    {...}
    vnfInstance*         Link    {...}
}

```

}

```

VnfExtCpConfig  {
  description:   This type represents an externally provided link port or
                 network address information per instance of an external
                 connection point.

  linkPortId    string($uuid)
                 Identifier of a pre-configured link port to which the
                 external CP will be associated.

  cpProtocolData [...]
  cpInstanceId  string($uuid)
                 Identifier of the external CP instance to which this set of
                 configuration parameters is requested to be applied.
}

```

```

ScaleVnfToLevelRequest  {
  description:   This type represents request parameters for the "Scale VNF to
                 Level" operation.

  instantiationLevelId string($uuid)
                 Identifier of the target instantiation level of the current
                 deployment flavour to which the VNF is requested to be
                 scaled.

  scaleInfo          [...]
  additionalParams  KeyValuePairs  {...}
}

```

```

HealVnfRequest  {
  description:   This type represents request parameters for the "Heal VNF"
                 operation.

  additionalParams  KeyValuePairs  {...}
  cause             string
                 Indicates the reason why a healing procedure is required.
}

```

LcmOperationType string

The enumeration LcmOperationType represents those lifecycle operations that trigger a VNF lifecycle management operation occurrence notification.

Enum:

Array [9]

```

ChangeVnfFlavourRequest  {
  description:   This type represents request parameters for the "Change VNF
                 flavour" operation.

  extManagedVirtualLinks [...]
  instantiationLevelId  string($uuid)
}

```

```

Identifier of the instantiation level of the deployment
flavour to be instantiated. If not present, the default
instantiation level as declared in the VNFD is
instantiated.

vimConnectionInfo      [...]
additionalParams       KeyValuePairs  {...}
extVirtualLinks        [...]
newFlavourId*          string($uuid)
                        Identifier of the VNF deployment flavour to be
                        instantiated.
}

```

```

ThresholdCriteria  {
  description:      This type represents criteria that define a threshold.

  simpleThresholdDetails  {...}
  performanceMetric*      string
                          Defines the performance metric associated with the
                          threshold, as specified in an external measurement
                          specification.

  thresholdType*          string
                          Type of threshold. This attribute determines which other
                          attributes are present in the data structure.

                          Enum:
                              Array [ 1 ]
}

```

```

PerceivedSeverityType  string
Enum:
  Array [ 6 ]

```

```

OperationMode  string
This type includes the Operation Mode of ETSI

```

```

MonitoringMigrateRequest  {
  description:      This type represents request parameters for the operate
                        operation available on ext API.

  key*                  string
                          This is the key in which the value for the monitoring agent
                          should be stored.

  monitoringAgent*      string
                          Deployment identifier of the monitoring agent. In the event
                          the agent is local to ESC, the string should be set to
                          "dmonaName://local_mona".
}

```

```

Threshold  {
  description:      This type represents a threshold

```

```

objectInstanceId*  string($uuid)
                   Identifier of the VNF instance associated with the threshold.

_links*           {...}
criteria*         ThresholdCriteria  {...}
id*               string($uuid)
                   Identifier of this threshold resource.
}

```

```

PmSubscriptionRequest  {
  description:       This type represents a subscription request related to
                     notifications about VNF performance.

  filter             PmNotificationsFilter  {...}
  callbackUri*      string($uri)
                     The URI of the endpoint to send the notification to.

  authentication    SubscriptionAuthentication  {...}
}

```

```

VnfcInfoModifications  {
  description:       This type represents modifications of an entry in an
                     array of "VnfcInfo" objects.

  vnfcConfigurableProperties*  KeyValuePairs  {...}
  id*                string($uuid)
                     Identifier of the VNFC instance of which the
                     information is to be modified.
}

```

```

LccnSubscription      {
  description:       This type represents a subscription related to notifications
                     about VNF lifecycle changes.

  filter             LifecycleChangeNotificationsFilter  {...}
  _links*           {...}
  callbackUri*      string($uri)
                     The URI of the endpoint to send the notification to.

  id*                string($uuid)
                     Identifier of this subscription resource.
}

```

```

VnfcInfoModificationRequestSol2  {
  description:       This type represents attribute modifications for an
                     "Individual VNF instance" resource, i.e.
                     modifications to a resource representation based on
                     the "VnfInstance" data type.

  vnfcInfoModifications  [...]
  metadata                KeyValuePairs  {...}
  extensions
}

```

```

        KeyValuePairs    {...}
vnfcInfoModificationsDeleteIds string($uuid)
    List of identifiers entries to be deleted from the
    "vnfcInfoModifications" attribute array to be used
    as "deleteIdList".

vnfPkgId                string($uuid)
    New value of the "vnfPkgId" attribute in
    "VnfInstance". The value "null" is not permitted.

vnfConfigurableProperties
vnfInstanceName        string
    New value of the "vnfInstanceName" attribute in
    "VnfInstance", or "null" to remove the attribute.

vnfInstanceDescription string
    New value of the "vnfInstanceDescription" attribute
    in "VnfInstance", or "null" to remove the
    attribute.
}

```

```

ExtLinkPort    {
    description:    This type represents a link port of an external VL, i.e. a
                    port providing connectivity for the VNF to an NS VL.

    resourceHandle* ResourceHandle    {...}
    id*             string($uuid)
                    Identifier of this link port as provided by the entity that
                    has created the link port.

    cpInstanceId   string($uuid)
                    Identifier of the external CP of the VNF to be connected to
                    this link port.
}

```

```

AffectedVirtualLink    {
    description:    This type provides information about added, deleted, modified
                    and temporary VLS

    networkResource* ResourceHandle    {...}
    changeType*     string
                    Signals the type of change.

                    Enum:
                    Array [ 6 ]
    virtualLinkDescId* string($uuid)
                    Identifier of the related VLD in the VNFD.

    id*             string($uuid)
                    Identifier of the virtual link instance, identifying the
                    applicable "vnfVirtualLinkResourceInfo" entry in the
                    "VnfInstance" data type
}

```

```

LcmOperationStateType string
Enum:
    Array [ 7 ]

```

VnfOperationalStateType string

Enum:

Array [2]

CreateVnfRequest {

description: This type represents request parameters for the "Create VNF identifier" operation.

vnfdId* string(\$uuid)
Identifier that identifies the VNFD which defines the VNF instance to be created.

vnfInstanceName string
Human-readable name of the VNF instance to be created.

vnfInstanceDescription string
Human-readable description of the VNF instance to be created.

}

VnfIdentifierCreationNotification {

description: This type represents a VNF identifier creation notification, which informs the receiver of the creation of a new VNF instance resource and the associated VNF instance identifier

timeStamp* string(\$date-time)
Date-time of the generation of the notification.

vnfInstanceId* string(\$uuid)
The created VNF instance identifier

_links* LccnLinks {...}

id* string(\$uuid)
Identifier of this notification

notificationType* string
Discriminator for the different notification types.

subscriptionId string(\$uuid)
Identifier of the subscription that this notification relates to.

}

MacAddress string

ChangeExtVnfConnectivityRequest {

description: This type represents request parameters for the "Change external VNF connectivity" operation to modify the external connectivity of a VNF instance.

vimConnectionInfo [...]

additionalParams KeyValuePairs {...}

extVirtualLinks* [...]

}

```

PmNotificationsFilter {
  description:           This type represents a filter that can be used to
                           subscribe for notifications related to performance
                           management events.

  vnfInstanceSubscriptionFilter VnfInstanceSubscriptionFilter {...}
  notificationTypes             [...]
}

```

```

PmJobCriteria {
  description:           This type represents collection criteria for PM jobs

  collectionPeriod*         integer($int32)
                           Specifies the periodicity at which the producer will
                           collect performance information.

  reportingPeriod*          integer($int32)
                           Specifies the periodicity at which the producer will report
                           to the consumer about performance information.

  reportingBoundary         string($date-time)
                           Identifies a time boundary after which the reporting will
                           stop. The boundary shall allow a single reporting as well
                           as periodic reporting up to the boundary.

  performanceMetricGroup   [...]
  performanceMetric         [...]
}

```

```

PerformanceReport {
  description:           This type defines the format of a performance report provided
                           by the VNFM to the NFVO as a result of collecting performance
                           information as part of a PM job.

  entries*                  [...]
}

```

```

CpProtocolData {
  description:           This type represents network protocol data.

  ipOverEthernet           [...]
  layerProtocol            string
                           Identifier of layer(s) and protocol(s). Permitted values:
                           IP_OVER_ETHERNET

  Enum:
      Array [ 1 ]
}

```

```

VirtualStorageResourceInfo {
  description:           This type represents the information that allows addressing
                           a virtualised resource that is used by a VNF instance

  metadata                 KeyValuePairs {...}
  reservationId           string($uuid)
}

```

The reservation identifier applicable to the resource. It shall be present when an applicable reservation exists.

```

virtualStorageDescId* string($uuid)
    Identifier of the VirtualStorageDesc in the VNFD.

id* string($uuid)
    Identifier of this VirtualStorageResourceInfo instance.

storageResource ResourceHandle {...}
}

FmSubscription {
    description: This type represents a subscription related to notifications
    about VNF faults.

    filter FmNotificationsFilter {...}

    _links* {...}

    callbackUri* string($uri)
        The URI of the endpoint to send the notification to.

    id* string($uuid)
        Identifier of this subscription resource.
}

FmSubscriptionRequest {
    description: This type represents a subscription request related to
    notifications about VNF faults.

    filter FmNotificationsFilter {...}

    callbackUri* string($uri)
        The URI of the endpoint to send the notification to.

    authentication SubscriptionAuthentication {...}
}

AlarmSol2 {
    description: The alarm data type encapsulates information about an
    alarm.

    isRootCause* boolean
        Attribute indicating if this fault is the root for other
        correlated alarms. If TRUE, then the alarms listed in the
        attribute CorrelatedAlarmId are caused by this fault.

    rootCauseFaultyResource* FaultyResourceInfo {...}

    alarmRaisedTime* string($date-time)
        Time stamp indicating when the alarm is raised by the
        managed object.

    alarmClearedTime string($date-time)
        Time stamp indicating when the alarm was cleared. It
        shall be present if the alarm has been cleared

    eventType* EventType string
        Enum:

        Array [ 5 ]

    alarmChangedTime string($date-time)

```



```

Time stamp indicating when the alarm was last changed. It
shall be present if the alarm has been updated.

ackState*      string
                Acknowledgement state of the alarm.

                Enum:
                    Array [ 2 ]
managedObjectId* string($uuid)
                Identifier of the affected VNF instance.

perceivedSeverity* PerceivedSeverityType string
                Enum:
                    Array [ 6 ]
probableCause*  string
                Information about the probable cause of the fault.

eventTime*     string($date-time)
                Time stamp indicating when the fault was observed.

faultType      string
                Additional information to clarify the type of the fault.

correlatedAlarmIds [...]
faultDetails   [...]
id*            string($uuid)
                Identifier of this Alarm information element.

vnfcInstanceIds* [...]
}

```

```

VnfInfoModificationsSol2 {
  description:      This type represents attribute modifications that were
                        performed on an "Individual VNF instance" resource. The
                        attributes that can be included consist of those
                        requested to be modified explicitly in the
                        "VnfInfoModificationRequest" data structure, and
                        additional attributes of the "VnfInstance" data
                        structure that were modified implicitly e.g. when
                        modifying the referenced VNF package.

  vnfProductName      string
                        If present, this attribute signals modifications of the
                        "vnfProductName" attribute in "VnfInstance".

  metadata             KeyValuePairs  {...}
  extensions           KeyValuePairs  {...}
  vnfdVersion         string
                        If present, this attribute signals modifications of the
                        "vnfdVersion" attribute in "VnfInstance".

  vnfProvider         string
                        If present, this attribute signals modifications of the
                        "vnfProvider" attribute in "VnfInstance".

  vnfConfigurableProperties KeyValuePairs  {...}
  vnfPkgId            string($uuid)
                        If present, this attribute signals modifications of the
                        "vnfPkgId" attribute in "VnfInstance".

  vnfdId              string($uuid)
                        If present, this attribute signals modifications of the
                        "vnfdId" attribute in "VnfInstance".
}

```

```

vnfInstanceName      string
                    If present, this attribute signals modifications of the
                    "vnfInstanceName" attribute in "VnfInstance".

vnfInstanceDescription string
                    If present, this attribute signals modifications of the
                    "vnfInstanceDescription" attribute in "VnfInstance".

vnfSoftwareVersion   string
                    If present, this attribute signals modifications of the
                    "vnfSoftwareVersion" attribute in "VnfInstance".

}

```

```

CancelMode {
  description: This type represents a parameter to select the mode of
               cancelling an ongoing VNF LCM operation occurrence.

  cancelMode* CancelModeType string
               Enum:
               Array [ 2 ]
}

```

```

VnfIdentifierDeletionNotification {
  description: This type represents a VNF identifier deletion notification,
               which informs the receiver of the deletion of a new VNF
               instance resource and the associated VNF instance identifier.

  timeStamp* string($date-time)
             Date-time of the generation of the notification.

  vnfInstanceId* string($uuid)
                The deleted VNF instance identifier

  _links* LccnLinks {...}

  id* string($uuid)
     Identifier of this notification

  notificationType* string
                   Discriminator for the different notification types.

  subscriptionId string($uuid)
                 Identifier of the subscription that this notification relates
                 to.

}

```

```

Link {
  description: This type represents a link to a resource.

  href* string($uri)
       URI of the referenced resource.

}

```

```

VnfcResourceInfo {
  description: This type represents the information on virtualised compute
               and storage resources used by a VNFC in a VNF instance

  metadata KeyValuePairs {...}
}

```

```

storageResourceIds      [...]
reservationId           string($uuid)
                        The reservation identifier applicable to the resource. It
                        shall be present when an applicable reservation exists.

vnfcCpInfo              {...}
id*                     string($uuid)
                        Identifier of this VnfcResourceInfo instance

vduId*                  string($uuid)
                        Reference to the applicable VDU in the VNFD.

computeResource        ResourceHandle  {...}
}

```

```

ExtManagedVirtualLinkData  {
  description:           This type represents an externally-managed internal VL.

  resourceId*           string($uuid)
                        The identifier of the resource in the scope of the VIM or the
                        resource provider.

  vimConnectionId       string($uuid)
                        Identifier of the VIM connection to manage this resource.
                        This attribute shall only be supported and present if VNF-
                        related resource management in direct mode is applicable.

  virtualLinkDescId*    string($uuid)
                        The identifier of the VLD in the VNFD for this VL.

  id*                   string($uuid)
                        The identifier of the externally-managed internal VL
                        instance.

  resourceProviderId    string($uuid)
                        Identifies the entity responsible for the management of this
                        resource. This attribute shall only be supported and present
                        if VNF-related resource management in indirect mode is
                        applicable.
}

```

```

MonitoringParameter        {
  description:           This type represents a monitoring parameter that is tracked
                        by the VNFM

  timeStamp*            string($date-time)
                        Represents the point in time when the measurement has been
                        performed, as known to the VNFM.

  name                  string
                        Human readable name of the monitoring parameter, as defined
                        in the VNFD.

  id*                   string($uuid)
                        Identifier of the monitoring parameter defined in the VNFD.

  value*                {...}
}

```

```

InstantiateVnfRequestSol2  {
  description:           This type represents request parameters for the

```

"Instantiate VNF" operation.

```

extManagedVirtualLinks    [...]
flavourId*                 string($uuid)
                           Identifier of the VNF deployment flavour to be
                           instantiated.

instantiationLevelId      string($uuid)
                           Identifier of the instantiation level of the deployment
                           flavour to be instantiated. If not present, the default
                           instantiation level as declared in the VNFD is
                           instantiated.

additionalParams           KeyValuePairs  {...}
extVirtualLinks           [...]
localizationLanguage      string
                           Localization language of the VNF to be instantiated.
}

```

```

ScaleVnfRequest  {
  description:      This type represents request parameters for the "Scale VNF"
                    operation.

  numberOfSteps    integer($int32)
                  Number of scaling steps to be executed as part of this Scale
                  VNF operation. It shall be a positive number and the default
                  value shall be 1.

  additionalParams KeyValuePairs  {...}
  aspectId*        string($uuid)
                  Identifier of the scaling aspect.

  type*            string
                  Indicates the type of the scale operation requested.

                  Enum:
                    Array [ 2 ]
}

```

```

OperateVnfRequest {
  description:      This type represents request parameters for the "Operate VNF"
                    operation.

  additionalParams KeyValuePairs  {...}
  stopType         StopType string
                  Enum:
                    Array [ 2 ]
  gracefulStopTimeout integer($int32)
                  The time interval (in seconds) to wait for the VNF to be
                  taken out of service during graceful stop, before stopping
                  the VNF. Ignored if changeStateTo=STARTED.

  changeStateTo*  VnfOperationalStateType string
                  Enum:
                    Array [ 2 ]
}

```

```

VnfLcmOpOccGeneric {

```

```

description:          This type represents a VNF lifecycle management operation
                        occurrence.

grantId                string($uuid)
                        Identifier of the grant related to this VNF LCM operation
                        occurrence, if such grant exists.

_links*                {...}

operationState*       LcmOperationStateType string
                        Enum:
                            Array [ 7 ]

error                  ProblemDetails  {...}

vnfInstanceId*        string($uuid)
                        Identifier of the VNF instance to which the operation
                        applies.

resourceChanges        {...}

cancelMode             CancelModeType string
                        Enum:
                            Array [ 2 ]

operationParams*      {...}

stateEnteredTime*     string($date-time)
                        Date-time when the current state was entered.

changedExtConnectivity
startTime*            string($date-time)
                        Date-time of the start of the operation.

id*                   string($uuid)
                        Identifier of this VNF lifecycle management operation
                        occurrence.

isAutomaticInvocation* boolean
                        Set to true if this VNF LCM operation occurrence has been
                        triggered by an automated procedure inside the VNFM (i.e.
                        ScaleVnf / ScaleVnfToLevel triggered by auto-scale, or
                        HealVnf triggered by auto-heal). Set to false otherwise.

operation*            LcmOperationType string
                        The enumeration LcmOperationType represents those lifecycle
                        operations that trigger a VNF lifecycle management
                        operation occurrence notification.
                        Enum:
                            Array [ 9 ]

isCancelPending*      boolean
                        If the VNF LCM operation occurrence is in "STARTING",
                        "PROCESSING" or "ROLLING_BACK" state and the operation is
                        being cancelled, this attribute shall be set to true.
                        Otherwise, it shall be set to false.

}

```

ExtManagedVirtualLinkInfo {

```

description:          This type provides information about an externally-managed
                        virtual link.

networkResource*      ResourceHandle  {...}

id*                   string($uuid)
                        Identifier of the externally-managed internal VL and the
                        related externally-managed VL information instance.

vnfVirtualLinkDescId* string($uuid)

```

Identifier of the VNF Virtual Link Descriptor (VLD) in the VNFD.

```
vnfLinkPorts      [...]
}

```

VnfLcmOperationOccurrenceNotification {

description: This type represents a VNF lifecycle management operation occurrence notification, which informs the receiver of changes in the VNF lifecycle caused by a VNF LCM operation occurrence.

notificationStatus* string
Indicates whether this notification reports about the start of a lifecycle operation or the result of a lifecycle operation.

Enum:

Array [2]

affectedVirtualLinks [...]

affectedVirtualStorages [...]

affectedVnfcs [...]

_links* LccnLinks {...}

operationState* LcmOperationStateType string
Enum:

Array [7]

notificationType* string
Discriminator for the different notification types.

error [...]

timeStamp* string(\$date-time)
Date-time of the generation of the notification.

vnfInstanceId* string(\$uuid)
The identifier of the VNF instance affected

vnfLcmOpOccId* string(\$uuid)
The identifier of the VNF lifecycle management operation occurrence associated to the notification.

changedInfo VnfInfoModifications {...}

changedExtConnectivity [...]

id* string(\$uuid)
Identifier of this notification

subscriptionId string(\$uuid)
Identifier of the subscription that this notification relates to.

isAutomaticInvocation* string(\$boolean)
Set to true if this VNF LCM operation occurrence has been triggered by an automated procedure inside the VNFM (i.e. ScaleVnf / ScaleVnfToLevel triggered by auto-scale, or HealVnf triggered by auto-heal).

operation* LcmOperationType string
The enumeration LcmOperationType represents those lifecycle operations that trigger a VNF lifecycle management operation occurrence notification.

Enum:

Array [9]

}

```

VnfLcmOpOccSol2  {
  description:      This type represents a VNF lifecycle management operation occurrence.

  grantId          string($uuid)
                  Identifier of the grant related to this VNF LCM operation occurrence, if such grant exists.

  _links*          {...}

  operationState*  LcmOperationStateType string
                  Enum:
                    Array [ 7 ]

  error            ProblemDetails  {...}

  vnfInstanceId*   string($uuid)
                  Identifier of the VNF instance to which the operation applies.

  resourceChanges  {...}

  cancelMode       CancelModeType string
                  Enum:
                    Array [ 2 ]

  operationParams* {...}

  stateEnteredTime* string($date-time)
                  Date-time when the current state was entered.

  changedExtConnectivity  [...]

  startTime*       string($date-time)
                  Date-time of the start of the operation.

  id*              string($uuid)
                  Identifier of this VNF lifecycle management operation occurrence.

  isAutomaticInvocation* boolean
                  Set to true if this VNF LCM operation occurrence has been triggered by an automated procedure inside the VNFM (i.e. ScaleVnf / ScaleVnfToLevel triggered by auto-scale, or HealVnf triggered by auto-heal). Set to false otherwise.

  operation*       LcmOperationType string
                  The enumeration LcmOperationType represents those lifecycle operations that trigger a VNF lifecycle management operation occurrence notification.
                  Enum:
                    Array [ 9 ]

  isCancelPending* boolean
                  If the VNF LCM operation occurrence is in "STARTING", "PROCESSING" or "ROLLING_BACK" state and the operation is being cancelled, this attribute shall be set to true. Otherwise, it shall be set to false.

  changedInfo      VnfInfoModificationsSol2  {...}
}

```

```

NetworkAddressInfo  {
  description:      This type represents information about a network address that has been assigned

```

```

    macAddress*      MacAddress string
    subnetIpRanges  [...]
    ipAddress       IPAddress string
  }

```

```

VnfInfoModifications {
  description:      This type represents attribute modifications that were performed on an "Individual VNF instance" resource. The attributes that can be included consist of those requested to be modified explicitly in the "VnfInfoModificationRequest" data structure, and additional attributes of the "VnfInstance" data structure that were modified implicitly e.g. when modifying the referenced VNF package.

  vnfProductName     string
                    If present, this attribute signals modifications of the "vnfProductName" attribute in "VnfInstance".

  metadata           KeyValuePairs  {...}
  extensions         KeyValuePairs  {...}
  vimConnectionInfo [...]
  vnfdVersion        string
                    If present, this attribute signals modifications of the "vnfdVersion" attribute in "VnfInstance".

  vnfProvider        string
                    If present, this attribute signals modifications of the "vnfProvider" attribute in "VnfInstance".

  vnfConfigurableProperties KeyValuePairs  {...}
  vnfPkgId           string($uuid)
                    If present, this attribute signals modifications of the "vnfPkgId" attribute in "VnfInstance".

  vnfdId             string($uuid)
                    If present, this attribute signals modifications of the "vnfdId" attribute in "VnfInstance".

  vnfInstanceName   string
                    If present, this attribute signals modifications of the "vnfInstanceName" attribute in "VnfInstance".

  vnfInstanceDescription string
                    If present, this attribute signals modifications of the "vnfInstanceDescription" attribute in "VnfInstance".

  vnfSoftwareVersion string
                    If present, this attribute signals modifications of the "vnfSoftwareVersion" attribute in "VnfInstance".
}

```

```

AffectedVnfc {
  description:      This type provides information about added, deleted, modified and temporary VNFCs.

  addedStorageResourceIds [...]
  changeType*        string
                    Signals the type of change
}

```



```

Enum:
    Array [ 4 ]
id*          string($uuid)
             Identifier of the Vnfc instance, identifying the
             applicable "vnfcResourceInfo" entry in the "VnfInstance"
             data type
vduId*       string($uuid)
             Identifier of the related VDU in the VNFD.
computeResource* ResourceHandle {...}
removedStorageResourceIds [...]
}

```

```

IpOverEthernetAddressData {
  description: This type represents network address data for IP over
               Ethernet.
  macAddress   string($mac)
               MAC address.
  ipAddresses  [...]
}

```

```

Entry {
  description: Performance information entry
  objectInstanceId* string
               The object instance (i.e. VNF instance) for which the
               performance metric is reported.
  performanceMetric* string
               Name of the metric collected.
  performanceValues* [...]
  objectType*       string
               Defines the object type for which performance information is
               reported
}

```

```

VnfVirtualLinkResourceInfo {
  description: This type represents the information that allows addressing
               a virtualised resource that is used by an internal VL
               instance in a VNF instance.
  metadata     KeyValuePairs {...}
  reservationId string($uuid)
               The reservation identifier applicable to the resource. It
               shall be present when an applicable reservation exists.
  networkResource* ResourceHandle {...}
  id*          string($uuid)
               Identifier of this VnfVirtualLinkResourceInfo instance.
  vnfVirtualLinkDescId* string($uuid)
               Identifier of the VNF Virtual Link Descriptor (VLD) in the
               VNFD.
  vnfLinkPorts
}

```

```

    [...]
}

VnfInstance {
  description:           This type represents a VNF instance.

  vnfProductName*       string
                        Name to identify the VNF Product. The value is copied
                        from the VNFD.

  vnfVersion*           string
                        Identifies the version of the VNFD. The value is copied
                        from the VNFD.

  vimConnectionInfo    [...]

  metadata              KeyValuePairs  {...}

  vnfProvider*         string
                        Provider of the VNF and the VNFD. The value is copied
                        from the VNFD.

  _links*              {...}

  vnfPkgId*            string($uuid)
                        Identifier of information held by the NFVO about the
                        specific VNF package on which the VNF is based. This
                        identifier was allocated by the NFVO.

  vnfConfigurableProperties  KeyValuePairs  {...}

  vnfId*               string($uuid)
                        Identifier of the VNFD on which the VNF instance is
                        based.

  instantiationState*  string
                        The instantiation state of the VNF.

                        Enum:
                        Array [ 2 ]

  vnfInstanceDescription  string
                        Human-readable description of the VNF instance.

  extensions            KeyValuePairs  {...}

  instantiatedVnfInfo    {...}

  vnfInstanceName       string
                        Name of the VNF instance.

  id*                   string($uuid)
                        Identifier of the VNF instance.

  vnfSoftwareVersion*   string
                        Software version of the VNF. The value is copied from
                        the VNFD.
}

KeyValuePairs {
  description:           This type represents a list of key-value pairs. The order of
                        the pairs in the list is not significant.
}

```

```

VnfcInfo {
  description: This type represents the information about a VNFC instance that is part of a VNF instance

  vnfcState* string
  State of the VNFC instance.

  Enum:
    Array [ 2 ]
  vnfcConfigurableProperties KeyValuePairs {...}
  id* string($uuid)
  Identifier of the VNFC instance.

  vduId* string($uuid)
  Reference to the applicable VDU information element in the VNFD.
}

```

```

TerminateVnfRequestSol2 {
  description: This type represents request parameters for the "Terminate VNF" operation.

  additionalParams KeyValuePairs {...}
  terminationType* string
  Indicates whether forceful or graceful termination is requested.

  Enum:
    Array [ 1 ]
}

```

```

VnfInstanceSubscriptionFilter {
  description: This type represents subscription filter criteria to match VNF instances.

  vnfdIds [...]
  vnfProductsFromProviders [...]
  vnfInstanceNames [...]
  vnfInstanceIds [...]
}

```

```

PmSubscription {
  description: This type represents a subscription related to notifications about VNF performance.

  filter PmNotificationsFilter {...}
  _links* {...}
  callbackUri* string($uri)
  The URI of the endpoint to send the notification to.

  id* string($uuid)
  Identifier that identifies the subscription.
}

```

```

AlarmModifications  {
  description:      This type represents attribute modifications for an
                    "Individual alarm" resource

  ackState*        string
                  New value of the "ackState" attribute in "Alarm".

                  Enum:
                    Array [ 1 ]
}

VnfLinkPort  {
  description:      This type represents a link port of an internal VL of a VNF

  resourceHandle*  ResourceHandle  {...}

  id*              string($uuid)
                  Identifier of this link port as provided by the entity that
                  has created the link port.

  cpInstanceId     string($uuid)
                  Identifier of the external CP of the VNF to be connected to
                  this link port.
}

CreateThresholdRequest  {
  description:      This type represents a request to create a threshold

  objectInstanceId*  string($uuid)
                  Identifier of the VNF instance associated with this
                  threshold.

  criteria*          ThresholdCriteria  {...}
}

Report  {
  description:      Information about available reports collected by this PM job.

  readyTime*        string($date-time)
                  The time when the report was made available.

  fileSize          integer($int32)
                  The size of the report file in bytes, if known.

  expiryTime        string($date-time)
                  The time when the report will expire.

  href*             string($uri)
                  The Uri where the report can be obtained.
}

ExtVirtualLinkData  {
  description:      This type represents an external VL.

  resourceId*       string($uuid)
                  The identifier of the resource in the scope of the VIM or the
                  resource provider.
}

```

```

vimConnectionId    string($uuid)
                   Identifier of the VIM connection to manage this resource.
                   This attribute shall only be supported and present if VNF-
                   related resource management in direct mode is applicable.

id*                string($uuid)
                   The identifier of the external VL instance.

resourceProviderId string($uuid)
                   Identifies the entity responsible for the management of this
                   resource. This attribute shall only be supported and present
                   if VNF-related resource management in indirect mode is
                   applicable.

extCps              [...]
}

```

```

ChangeVnfFlavourRequestSol2 {
  description:      This type represents request parameters for the "Change VNF
                       flavour" operation.

  extManagedVirtualLinks [...]
  instantiationLevelId string($uuid)
                       Identifier of the instantiation level of the deployment
                       flavour to be instantiated. If not present, the default
                       instantiation level as declared in the VNFD is
                       instantiated.

  additionalParams    KeyValuePairs {...}
  extVirtualLinks     [...]
  newFlavourId*       string($uuid)
                       Identifier of the VNF deployment flavour to be
                       instantiated.
}

```

```

ProblemDetails {
  description:      A JSON representation of a "ProblemDetails" data structure
                       according to IETF RFC 7807 that provides additional details
                       of the error

  instance            string($uri)
                       A URI reference that identifies the specific occurrence of
                       the problem.

  detail*             string
                       A human-readable explanation specific to this occurrence of
                       the problem.

  type                string($uri)
                       A URI reference according to IETF RFC 3986 [5] that
                       identifies the problem type.

  title               string
                       A short, human-readable summary of the problem type.

  status*             integer($int32)
                       The HTTP status code for this occurrence of the problem

  additionalAttributes [...]
}

```

```

VnfExtCpData  {
  description:      This type represents an external CP.

  cpdId*           string($uuid)
                  The identifier of the CPD in the VNFD.

  cpConfig*       [...]
}

```

```

StopType  string
Enum:
  Array [ 2 ]

```

```

OperateRequest  {
  description:      This type represents request parameters for the operate
                  operation available on ext API.

  additionalParams  KeyValuePairs  {...}

  vnfcInstanceIds  [...]

  operation*        {...}
}

```

```

VnfLcmOpOcc  {
  description:      This type represents a VNF lifecycle management operation
                  occurrence.

  grantId          string($uuid)
                  Identifier of the grant related to this VNF LCM operation
                  occurrence, if such grant exists.

  _links*         {...}

  operationState* LcmOperationStateType string
                  Enum:
                  Array [ 7 ]

  error           ProblemDetails  {...}

  vnfInstanceId* string($uuid)
                  Identifier of the VNF instance to which the operation
                  applies.

  resourceChanges  {...}

  cancelMode       CancelModeType string
                  Enum:
                  Array [ 2 ]

  operationParams* {...}

  stateEnteredTime* string($date-time)
                  Date-time when the current state was entered.

  changedExtConnectivity  [...]

  startTime*        string($date-time)
                  Date-time of the start of the operation.

  id*              string($uuid)
                  Identifier of this VNF lifecycle management operation
                  occurrence.

  isAutomaticInvocation* boolean

```

```

        Set to true if this VNF LCM operation occurrence has been
        triggered by an automated procedure inside the VNFM (i.e.
        ScaleVnf / ScaleVnfToLevel triggered by auto-scale, or
        HealVnf triggered by auto-heal). Set to false otherwise.

        operation*          LcmOperationType string
                            The enumeration LcmOperationType represents those lifecycle
                            operations that trigger a VNF lifecycle management
                            operation occurrence notification.

                            Enum:
                                Array [ 9 ]

        isCancelPending*    boolean
                            If the VNF LCM operation occurrence is in "STARTING",
                            "PROCESSING" or "ROLLING_BACK" state and the operation is
                            being cancelled, this attribute shall be set to true.
                            Otherwise, it shall be set to false.

        changedInfo         VnfInfoModifications {...}
    }

```

```

SubscriptionAuthentication {
    description:          A data structure that defines the authorization
                            requirements.

    params0auth2ClientCredentials {...}
    paramsBasic             {...}
    authType*              [...]
}

```

```

LccnSubscriptionRequest {
    description:          This type represents a subscription request related to
                            notifications about VNF lifecycle changes.

    filter                  LifecycleChangeNotificationsFilter {...}
    callbackUri*           string($uri)
                            The URI of the endpoint to send the notification to.

    authentication         SubscriptionAuthentication {...}
}

```

```

FaultyResourceType string
Enum:
    Array [ 3 ]

```

```

FaultyResourceInfo {
    description:          This type represents the faulty virtual resources that have a
                            negative impact on a VNF

    faultyResource*        ResourceHandle {...}
    faultyResourceType*    FaultyResourceType string
                            Enum:
                                Array [ 3 ]

    id*                    string($uuid)
                            Unique identifier of the Faulty Resource Info object
}

```

}

Alarm {

<i>description:</i>	<i>The alarm data type encapsulates information about an alarm.</i>
isRootCause*	boolean Attribute indicating if this fault is the root for other correlated alarms. If TRUE, then the alarms listed in the attribute CorrelatedAlarmId are caused by this fault.
rootCauseFaultyResource*	FaultyResourceInfo {...}
alarmRaisedTime*	string(\$date-time) Time stamp indicating when the alarm is raised by the managed object.
alarmClearedTime	string(\$date-time) Time stamp indicating when the alarm was cleared. It shall be present if the alarm has been cleared
eventType*	EventType string Enum: Array [5]
alarmChangedTime	string(\$date-time) Time stamp indicating when the alarm was last changed. It shall be present if the alarm has been updated.
ackState*	string Acknowledgement state of the alarm. Enum: Array [2]
managedObjectId*	string(\$uuid) Identifier of the affected VNF instance.
perceivedSeverity*	PerceivedSeverityType string Enum: Array [6]
probableCause*	string Information about the probable cause of the fault.
eventTime*	string(\$date-time) Time stamp indicating when the fault was observed.
faultType	string Additional information to clarify the type of the fault.
correlatedAlarmIds	[...]
faultDetails	[...]
id*	string(\$uuid) Identifier of this Alarm information element.

}

ResourceHandle {

<i>description:</i>	<i>This type represents the information that allows addressing a virtualised resource that is used by a VNF instance. Information about the resource is available from the VIM.</i>
resourceId*	string(\$uuid) Identifier of the resource in the scope of the VIM or the resource provider.


```

vimConnectionId    string($uuid)
                   Identifier of the VIM connection to manage the resource.

vimLevelResourceType string
                   Type of the resource in the scope of the VIM or the resource
                   provider.

resourceProviderId string($uuid)
                   Identifier of the entity responsible for the management of
                   the resource.
}

```

```

ExtVirtualLinkInfo {
  description:      This type represents information about an external VL.
  resourceHandle*  ResourceHandle  {...}
  linkPorts        [...]
  id*              string($uuid)
                   Identifier of the external VL and the related external VL
                   information instance
}

```

```

IpAddress string

```

```

PmJob {
  description:      This type represents a PM job
  reports           [...]
  objectInstanceIds* [...]
  criteria*        PmJobCriteria  {...}
  id*              string($uuid)
                   Identifier of this PM job.
}

```

```

PerformanceValue {
  description:      Performance value with associated timestamp
  value*           {...}
  timestamp*       string($date-time)
                   Time stamp indicating when the data was collected.
}

```