



## **Deliverable 3.4: Prototype deliverable**

Dissemination level	PU
Version	0.1
Due date	31.10.2015
Version date	16.11.2015

This project is co-funded  
by the European Union



---

## Document information

---

### Editors

Balázs Sonkoly (BME)

---

### Contributors

Balázs Sonkoly (BME), János Czentye (BME), Balázs Németh (BME), Róbert Szabó (ETH), Raphael Vicente Rosa (ETH), Dávid Jocha (ETH), Sahel Saghaf (iMinds)

---

### Coordinator

Dr. András Császár  
Ericsson Magyarország Kommunikációs Rendszerek Kft. (ETH)  
Könyves Kálmán körút 11/B épület  
1097 Budapest, Hungary  
Fax: +36 (1) 437-7467  
Email: [andras.csaszar@ericsson.com](mailto:andras.csaszar@ericsson.com)

---

### Project funding

7th Framework Programme  
FP7-ICT-2013-11  
Collaborative project  
Grant Agreement No. 619609

---

### Legal Disclaimer

The information in this document is provided 'as is', and no guarantee or warranty is given that the information is fit for any particular purpose. The above referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law.

© 2013 - 2015 by UNIFY Consortium

---

## Summary

This document provides high-level and low-level overviews and programming documentations on the accompanying prototypes delivered by WP3. The current version of the main prototype of WP3 called SPOOPro is publicly released and also delivered to WP2 for integration into the project-wide UNIFY integrated prototype (IntPro). The final report on SPOOPro will be given in D3.5 including potential modifications required during the integration.

Our prototype has been organized into two separated software packages, namely into ESCAPE and into the Virtualizer3 library. Both components are publicly released under the Apache License, Version 2.0. The main part of the document provides a higher level summary and two annex sections are devoted to giving a deeper insight into these packages, respectively.

## Contents

<b>1 ESCAPE: description of the prototype</b>	<b>1</b>
1.1 High-level overview . . . . .	2
1.2 System architecture of ESCAPE . . . . .	2
1.2.1 Service Layer . . . . .	2
1.2.2 Orchestration Layer . . . . .	4
1.2.3 An online mapping algorithm . . . . .	5
<b>2 Virtualizer3 library</b>	<b>8</b>
2.1 Main features . . . . .	8
2.2 Usage examples . . . . .	9
2.2.1 Parse from file . . . . .	9
2.2.2 Get-config . . . . .	10
2.2.3 Edit-config . . . . .	10
2.2.4 Edit-config: multiple requests . . . . .	11
2.2.5 Edit-config and copy-config: diff and patch . . . . .	11
<b>3 Supported infrastructure domains</b>	<b>12</b>
<b>4 Illustrating the operation of ESCAPE</b>	<b>14</b>
4.1 Bottom-up process flow . . . . .	14
4.2 Top-down process flow . . . . .	15
<b>5 ESCAPE: implementation view</b>	<b>17</b>
5.1 Service module . . . . .	18
5.2 Orchestration module . . . . .	20
5.3 Adaptation module . . . . .	22
5.4 Infrastructure module . . . . .	24
5.5 Network Function Forwarding Graph (NF-FG) module . . . . .	25
5.6 Implemented interfaces . . . . .	25
<b>References</b>	<b>33</b>

<b>1</b>	<b>Overview</b>	<b>35</b>
<b>2</b>	<b>Installation</b>	<b>36</b>
2.1	The preferred way . . . . .	36
2.2	The hard way . . . . .	37
2.3	Setup a Virtual environment (optional) . . . . .	38
<b>3</b>	<b>ESCAPEv2 example commands</b>	<b>40</b>
3.1	The simplest use-case . . . . .	40
3.2	More advanced commands (mostly advisable for testing purposes) . . . . .	41
<b>4</b>	<b>REST APIs</b>	<b>43</b>
4.1	Common API functions . . . . .	43
4.2	Service API specific functions . . . . .	43
4.3	ROS API specific functions . . . . .	43
4.4	Cf-Or API specific functions . . . . .	44
<b>5</b>	<b>Configuration</b>	<b>45</b>
5.1	Configuration structure . . . . .	45
5.2	Default configuration . . . . .	47
<b>6</b>	<b>Development</b>	<b>50</b>
<b>7</b>	<b>Debugging</b>	<b>51</b>
<b>8</b>	<b>API documentation</b>	<b>52</b>
8.1	ESCAPEv2 class structure . . . . .	52
8.2	Topmost POX modules for UNIFY's layers/sublayers . . . . .	155
<b>9</b>	<b>Contacts</b>	<b>205</b>
<b>10</b>	<b>Indices and tables</b>	<b>206</b>

# Contents of Annex II

<b>1 Namespace Index</b>	<b>229</b>
1.1 Packages . . . . .	229
<b>2 Hierarchical Index</b>	<b>230</b>
2.1 Class Hierarchy . . . . .	230
<b>3 Class Index</b>	<b>232</b>
3.1 Class List . . . . .	232
<b>4 File Index</b>	<b>234</b>
4.1 File List . . . . .	234
<b>5 Namespace Documentation</b>	<b>235</b>
5.1 gen Namespace Reference . . . . .	235
5.2 gen.baseclasses Namespace Reference . . . . .	235
5.3 gen.virtualizer3 Namespace Reference . . . . .	240
<b>6 Class Documentation</b>	<b>242</b>
6.1 gen.baseclasses.BooleanLeaf Class Reference . . . . .	242
6.2 gen.baseclasses.Decimal64Leaf Class Reference . . . . .	244
6.3 gen.baseclasses.FilterYang Class Reference . . . . .	248
6.4 gen.virtualizer3.Flowentry Class Reference . . . . .	250
6.5 gen.virtualizer3.FlowtableFlowtable Class Reference . . . . .	252
6.6 gen.virtualizer3.GroupingFlowentry Class Reference . . . . .	254
6.7 gen.virtualizer3.GroupingFlowtable Class Reference . . . . .	257
6.8 gen.virtualizer3.GroupingId_name Class Reference . . . . .	258
6.9 gen.virtualizer3.GroupingId_name_type Class Reference . . . . .	260
6.10 gen.virtualizer3.GroupingInfra_node Class Reference . . . . .	262
6.11 gen.virtualizer3.GroupingLink Class Reference . . . . .	263
6.12 gen.virtualizer3.GroupingLink_resource Class Reference . . . . .	265
6.13 gen.virtualizer3.GroupingLinks Class Reference . . . . .	267
6.14 gen.virtualizer3.GroupingNode Class Reference . . . . .	268
6.15 gen.virtualizer3.GroupingNodes Class Reference . . . . .	271

**Contents of Annex II**

6.16 gen.virtualizer3.GroupingPort Class Reference . . . . .	273
6.17 gen.virtualizer3.GroupingSoftware_resource Class Reference . . . . .	275
6.18 gen.virtualizer3.Infra_node Class Reference . . . . .	277
6.19 gen.virtualizer3.Infra_capabilities Class Reference . . . . .	278
6.20 gen.baseclasses.IntLeaf Class Reference . . . . .	279
6.21 gen.baseclasses.Leaf Class Reference . . . . .	282
6.22 gen.baseclasses.Leafref Class Reference . . . . .	285
6.23 gen.virtualizer3.Link Class Reference . . . . .	288
6.24 gen.virtualizer3.Link_resource Class Reference . . . . .	290
6.25 gen.virtualizer3.LinksLinks Class Reference . . . . .	291
6.26 gen.baseclasses.ListedYang Class Reference . . . . .	293
6.27 gen.baseclasses.ListYang Class Reference . . . . .	295
6.28 gen.virtualizer3.Node Class Reference . . . . .	297
6.29 gen.virtualizer3.NodePorts Class Reference . . . . .	298
6.30 gen.virtualizer3.Nodes Class Reference . . . . .	300
6.31 object Class Reference . . . . .	301
6.32 gen.virtualizer3.Port Class Reference . . . . .	302
6.33 gen.virtualizer3.Port_ref Class Reference . . . . .	304
6.34 gen.virtualizer3.Software_resource Class Reference . . . . .	306
6.35 gen.baseclasses.StringLeaf Class Reference . . . . .	307
6.36 gen.virtualizer3.Virtualizer Class Reference . . . . .	309
6.37 gen.virtualizer3.VirtualizerNodes Class Reference . . . . .	311
6.38 gen.baseclasses.Yang Class Reference . . . . .	313
<b>7 File Documentation . . . . .</b>	<b>319</b>
7.1 baseclasses.py File Reference . . . . .	319
7.2 virtualizer3.py File Reference . . . . .	321
<b>Index . . . . .</b>	<b>322</b>

## 1 ESCAPE: description of the prototype

This section is devoted to give an overview on ESCAPEv2 (in the rest of the document, we refer to it as ESCAPE). This is the main proof of concept prototype of WP3 implementing all relevant components of our service programming and orchestration framework. It is a realization of the UNIFY programmability framework which enables the joint programming and virtualization of cloud and networking resources. An essential requirement from our framework is the support of efficient integration of different modules implemented by different partners and the easy re-use and integration of available tools. Moreover, ESCAPE is designed to be a platform for inter-workpackage integration. On the one hand, it supports the orchestration on top of Universal Node (UN) which is the main outcome of WP5. On the other hand, verification, monitoring and troubleshooting tools implemented by WP4 will also be integrated with this framework. The current state of the prototype and further implementation details can be found in Section 5.

The high-level overview on ESCAPE, the system architecture with the main components, illustrative examples and some lower level implementation details are given in the following subsections. Additionally, Annex I provides the missing pieces, such as installation instructions, examples on the usage, configuration choices, and detailed API documentation.

The online version of the documentation including all aforementioned parts can be found here:

- main wiki page: <https://sb.tmit.bme.hu/mediawiki/index.php/ESCAPE>
- API documentation: <https://sb.tmit.bme.hu/escape>

The repository of the source codes is currently hosted on our internal project server<sup>1</sup>:

- <https://gitlab.fp7-unify.eu/Balazs.Sonkoly/escape-shared>

A detailed description on the previous state of the prototype was given in [D3.2]. Here, we summarize the **main changes and progress** from that version:

- integration of the Virtualizer library (Virtualizer3, see Section 2), implemented by ETH, which supports several operations with NF-FGs following the Virtualizer Yang model (described in [D3.2a], updated in [D3.3])
- OpenStack (OS) domain can be controlled via the SI-Or interface based on the Virtualizer library (in the previous version, two dedicated control channels were used for compute and networking resources, respectively)
- supporting recursive orchestration, multiple ESCAPE instances or other orchestrators supporting the Virtualizer library can be stacked on top of each other
- significant extension of the Controller Adaptation Sublayer (CAS), clarification of the roles of domain managers and adapters
- full implementation of the top-down and bottom-up process flows
- supporting multi-domain orchestration (merging and splitting NF-FGs according to the available domains)
- initial implementation of the Cf-Or interface
- integrating our online mapping algorithm

<sup>1</sup>This repository is publicly available. However, it will be moved to github later.

## 1.1 High-level overview

ESCAPE is a multi-domain orchestrator, thus strictly speaking, it implements the Orchestration Layer (OL) of the UNIFY architecture. However, we have added a simple Service Layer (SL) interacting with clients, and implemented an Infrastructure Layer (IL) based on Mininet. This combination of the components enables to easily setup a standalone development framework and supports agile prototyping of different elements of our SFC control plane. The high-level components and their relations are shown in Figure 1a.

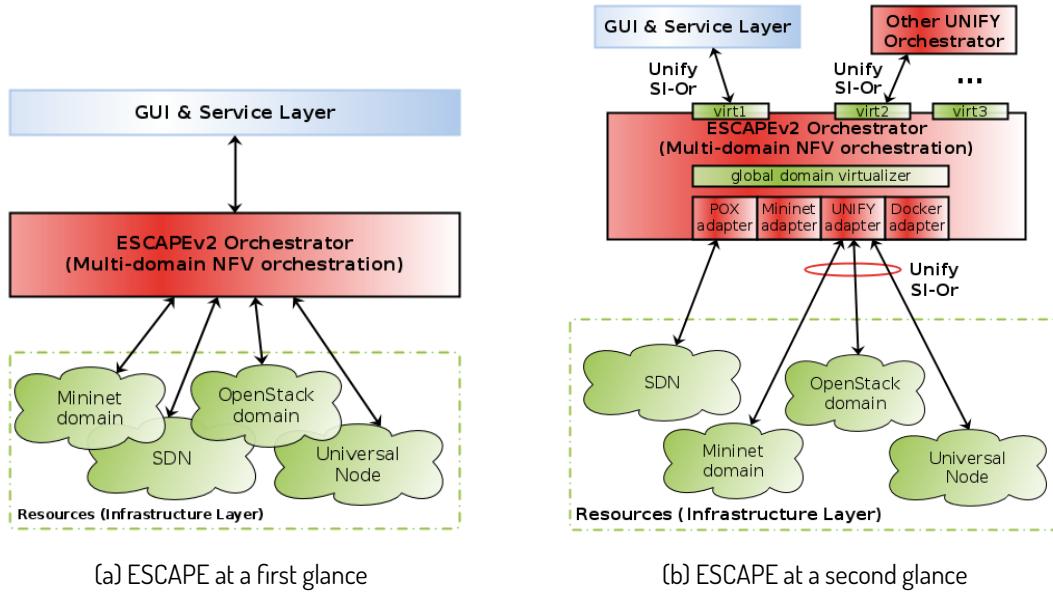


Figure 1: ESCAPE: a multi-domain NFV orchestrator

A more detailed architecture view is given in Figure 1b. ESCAPE implements the main interface of UNIFY, namely the SI-Or, both at north and south. This enables multiple higher-level orchestrators on top of ESCAPE with corresponding virtual infrastructure views provided by virtualizers. ESCAPE itself constructs and works on a global domain view. The higher-level virtualizer configurations and Virtualized Network Function (VNF) deployments are multiplexed in this element. The connection towards different infrastructure domains based on legacy or novel technologies are realized via dedicated adapter modules of ESCAPE. The most important one called UNIFY adapter implements the SI-Or interface. By this means, the same interface can be used for different technological domains.

## 1.2 System architecture of ESCAPE

The detailed system architecture of the OL and the optional SL is shown in Figure 2.

ESCAPE is (mainly) implemented in Python on top of POX platform [POX]. The modular approach and loosely coupled components make it easy to change several parts and evaluate custom algorithms. In this section, we introduce the main components, interfaces and features of the framework. Further details on the implementation are given in Section 5.

### 1.2.1 Service Layer

The SL contains an API and a GUI at the top level where users can request and manage services and VNFs. The GUI has been significantly redesigned and now we are working on the integration of Juju [Juju] and ESCAPE. This will make a more professional, Juju-based user interface available by the end of the next phase. Currently, a REST API

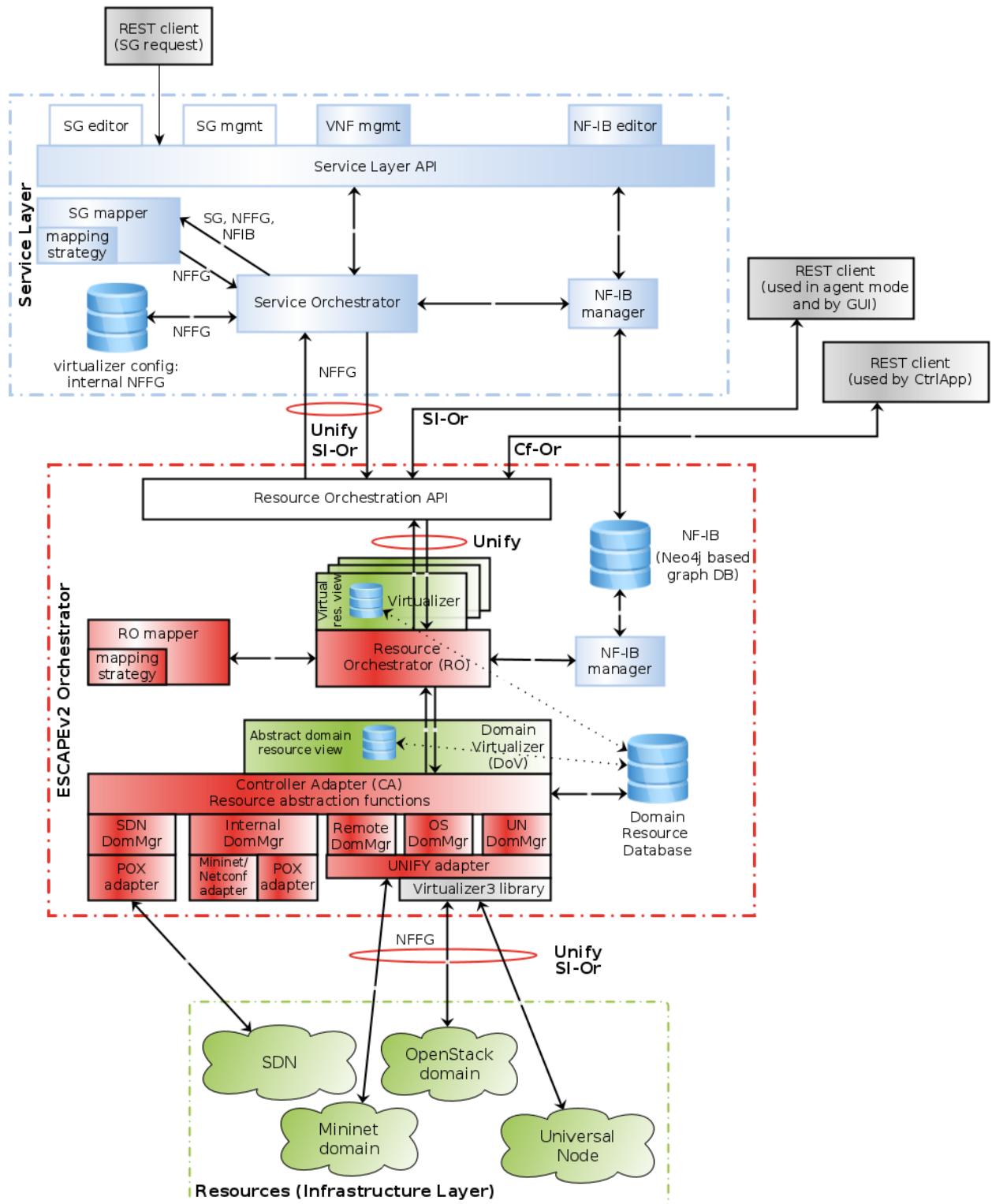


Figure 2: System architecture of ESCAPE

can be used for interacting with this layer. For example, REST client plugins are available for popular web browsers, hence the requests can be sent simply from browsers. The API is capable of formulating a Service Graph (SG) from the request (which should follow a given json format describing the SG) and passes that to a dedicated service orchestrator. This element is responsible for gathering resource information provided by the Virtualizer of the lower layer (e.g., BiS-

---

BiS view). This information on the virtual resources is stored in our internal NF-FG format. Additionally, the service orchestrator can retrieve information from the Network Function Information Base (NF-IB) (see later). Mapping of SG to available resources is delegated to the SG mapper module which constructs an NF-FG storing the request, the virtual resources and the mapping between Network Functions (NFs) and infrastructure nodes in a common data structure.

### 1.2.2 Orchestration Layer

OL encompasses the most important components of the resource orchestration process which replaces ETSI's VIM. An API is set up on the top centralizing the interaction with the upper layer (SL or other orchestrators) and realizing the (*i*) full functionality of the SI-Or interface, and (*ii*) an initial version of the Cf-Or interface. According to our concept, the virtualizer is the managed entity while the other peer of the relation is the manager. The manager can configure given deployments based on the requests coming from upper levels, and it can also retrieve the running configuration. This is a NETCONF like configuration approach, however, currently we are using a simple HTTP based transport to convey our own XML structures following the virtualizer YANG data model. In long term, we will integrate available NETCONF clients and servers with the framework and the standard NETCONF protocol will be used for the communication. The main (NETCONF compliant) functions supported by the API are the following:

- get-config
- edit-config

On the one hand, the request coming as an NF-FG in an edit-config call is forwarded to the Resource Orchestrator (RO) via the corresponding Virtualizer (which is responsible for policy enforcement as well). On the other hand, the virtual view and the current configuration (containing deployment information as well) of the Virtualizer is provided as another NF-FG to the upper layer via get-config calls. RO is the key entity managing the components involved in the orchestration process. The input is an NF-FG which should be deployed on top of the abstract domain resource view provided by the Domain Virtualizer. RO collects and forwards all required data to RO mapper which multiplexes the requests coming from different virtualizers. More specifically, the NF-FG, the domain view and the NF-IB are passed to the RO mapper which invokes the configured mapping strategy and optionally interacts with the Neo4j-based graph database containing information on NFs and decomposition rules. NF-IB corresponds to “VNF Catalogue” in NFV MANO with the difference of supporting service decomposition. The outcome is a new NF-FG which is sent to the Controller Adapter (CA) component.

The role of the CA is twofold. First, it gathers technology specific information on resources of different domains then builds an abstract domain view (bottom-up process flow). The RO works with this abstract model. Second, the result of the mapping process, which is an NF-FG describing the deployment on the Domain Virtualizer, is decomposed according to the lower level domains and delegated to the corresponding lower level orchestrators or controllers (top-down process flow). The interaction with different types of technology domains are handled by domain managers and adapter modules. (A given domain manager can use multiple adapters for different control channels.) Currently, we have the following domain managers exploiting the given adapters:

- **SDN Domain manager:** handle OpenFlow domains, convert abstract flow rules to OpenFlow flow entries
  - **POX adapter:** use POX OpenFlow controller to configure switches
- **Internal Domain manager:** enable internal Mininet domains mainly for rapid prototyping

- **Mininet adapter:** build/start/stop local Mininet infrastructure, the topology can be given as input in a config file
- **POX adapter:** use POX OpenFlow controller to configure switches
- **VNFStarter / Netconf adapter:** initiate/start/stop NF, connect/disconnect NF to/from switch, get info on running NF via the Netconf protocol
- **Remote Domain manager:** control remote ESCAPE orchestrating e.g., a local Mininet domain (recursive orchestration)
  - **UNIFY adapter:** implement the SI-Or interface based on our internal NF-FG data model
- **OpenStack Domain manager:** control an OS domain
  - **UNIFY adapter:** implement the SI-Or interface based on the Virtualizer3 library and NF-FG data model
- **Universal Node Domain manager:** control an UN domain
  - **UNIFY adapter:** implement the SI-Or interface based on the Virtualizer3 library and NF-FG data model

### 1.2.3 An online mapping algorithm

In [D3.3], we have presented our recent achievements with Divine embedding algorithm which is based on Integer Programming. On the one hand, the results are promising and during the next period we will add it as a novel mapping strategy to ESCAPE. On the other hand, ESCAPE is a modular framework providing default implementations for all relevant components of UNIFY service programming and orchestration framework. These default modules can easily be changed. We have already integrated a simple, greedy mapping strategy with ESCAPE in order to support the full top-down process flow. Here, we present this component realizing the online mapping of requests to available resources. This algorithm can be used both in the Service and Orchestration layers.

We have designed and implemented a fast and efficient resource orchestration algorithm in order to handle the full mapping of enormous amount of Service Graphs arriving within few seconds to provider networks in the scale of several tens-to-hundreds of thousands of substrate nodes. It is a preference value based greedy backtrack algorithm based on graph pattern matching and bounded graph simulation. We have added this algorithm as a new mapping strategy to ESCAPE.

Besides the physical nodes and NFs, the substrate and the service graph have Service Access Points (SAPs), where the user connects to the network. These endpoints can be mapped unambiguously between the two graphs. Service chains are expressed as end-to-end paths in the Service Graph (see Service Layer of Figure 3), defining which NFs should a specific subset of network traffic traverse between the appropriate SAPs, meanwhile the given requirements should hold on the path.

A simplified pseudo code of the algorithm can be found in Alg. 1. Line 7 in the pseudo code basically says that the Service Graph is partitioned into small paths derived from the end-to-end service chain requirements, so that, the subchains are disjoint on the edges of the Service Graph. This aids the greedy mapping process to find an appropriate host for NFs which are used by multiple service chains. In line 13, the variable  $k$  indicates the branching factor of the backtrack process. More specifically, we select and store the  $k$  best candidates and try the best one first. If backtrack is required later (because the NF cannot be mapped), the second candidate will be selected, and so forth.

The substrate nodes are ordered by a composite preference value function, which takes notice of (i) the weighted sum of the substrate node's utilizations of the resource components, (ii) the path length measured in latency and (iii)

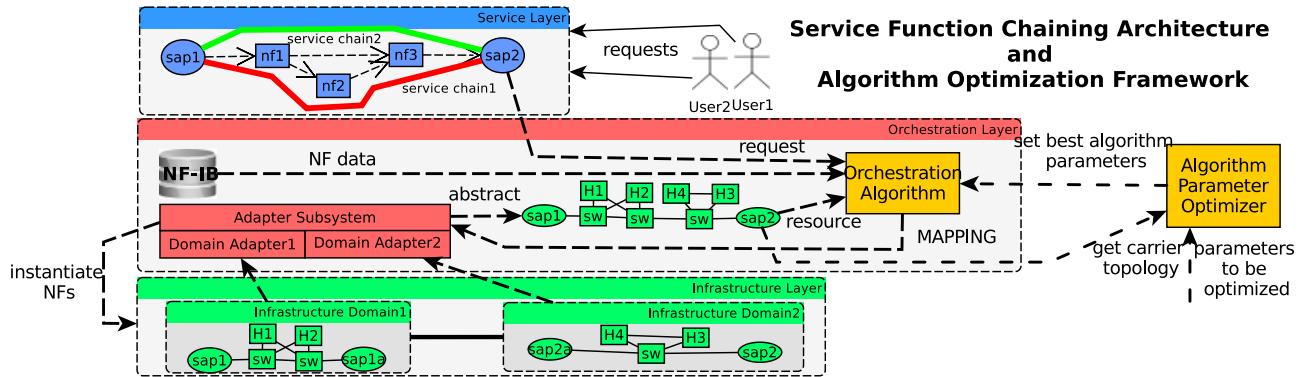


Figure 3: A possible layered SFC architecture extended with the orchestration Algorithm Optimization Framework.

#### Algorithm 1: Online Mapping Algorithm

---

**Input:** substrate graph  $(V, E)$  with available resources, bandwidths, latencies; Service Graph  $(V_p, E_p)$  with end-to-end service chains  $(SC)$ , and its quality of service requirements  
**Output:** mapping of NFs to substrate nodes, mapping of logical NF connections to substrate paths

- 1 **Procedure**  $MAP((V, E), (V_p, E_p), (SC))$
- 2   Basic preprocessing on request and substrate graph
- 3   **forall**  $SAP_p \in V_p$  **do**
- 4     | Find  $SAP$  from  $V$  for  $SAP_p$
- 5   **end**
- 6   Find helper subgraphs for each end-to-end chain in the substrate graph
- 7   **forall**  $c \in SC$  **do**
- 8     | Divide  $c$  into subchains ( $subc$ ) so every edge in  $E_p$  is exactly in one subchain
- 9     | Add  $subc$  to  $sSC$
- 10   **end**
- 11   **forall**  $subc \in sSC$  **do**
- 12     **forall**  $NF \in subc$  **do**
- 13       | Push  $k$  best substrate nodes and paths to  $backtrack(NF)$  stack
- 14       | **if**  $NF \in$  other subchains **then**
- 15         | Set placement criteria for  $NF$
- 16       | **end**
- 17       | Map  $NF$  to best substrate node and path leading to it
- 18       | Update graph resources
- 19     **end**
- 20     | **if**  $NF$  could not be mapped anywhere **then**
- 21       | Pop substrate node and path from  $backtrack(NF)$  stack
- 22       | Redo graph and path resources
- 23       | Try allocating  $NF$  again
- 24     **end**
- 25     | Map last unmapped edge of  $subc$
- 26   **end**
- 27   **return** Complete mapping

---

the average link utilization on the path leading to the node. Furthermore, the importance among the three components is also an adjustable parameter. This preference value defines the best candidates among the substrate nodes, where  $NF$  could be mapped.

The placement criteria in line 15 is determined by the NF sharing between end-to-end chains. In line 17, the algo-

---

rithm subtracts the resources from the substrate graph reserved for the allocation of  $NF$ . If  $NF$  could not be mapped and backtrack is required, the resource reservation is undone in line 22.

### 1.2.3.1 Parameter fine-tuning survey

The previously presented algorithm has an abundance of adjustable parameters, which cannot be set without correct examination of the quality of service requirements of the Service Graphs, and their correlation to the resources of the substrate graph. So a right parameter setting is highly dependent on the environment where the algorithm is to be applied.

How should the utilization of a substrate node's resource component (e.g. memory) be valued? What weighting should be used between the resource components? How should a substrate path leading to a candidate host node be selected? What should be the relative importance of the components of the substrate node preference value? What branching factor and depth should the backtrack process have? What kind of Service Graphs are expected?

These questions must be answered so that the acceptance ratio and possibly the quality of orchestration of the algorithm would be maximized (see Algorithm Parameter Optimizer (APO) in Fig. 3). The problem with the assessment of the mapping quality is that the optimal solution for an input is unknown, so we can only give estimates based on the reservation state of the network. Acceptance ratio is tested with benchmark Service Graph requests (increasing in size and strictness), which is much easier to define. The type of SG benchmark sequence, which roughly describes the nature of the expected SG requests from the users, can be defined as an input to the APO.

The feasibility of the survey highly depends on the desired quality of the parameter optimization and the available computation capacity besides the size of the substrate network. For example, the survey for a substrate network around the scale of twelve thousands nodes takes several days to complete on our university's supercomputer. When the substrate network topology is changed slightly, i.e., a few network paths or servers are affected, the resulting parameter setting of the survey still holds, because it is valid for the overall resource capabilities of the network. The parameter fine-tuning approach can be applied in both single-domain and multi-domain environments depending on the current role of ESCAPE.

As a part of the Algorithm Optimization Framework, an arbitrary, large scale service provider network can be defined with all of its resource parameters, where the fine-tuned algorithm will be applied. Using this target substrate network, a combinatorial optimization process<sup>2</sup> in the parameter space defined by a set of the mentioned tunable algorithm parameters is conducted to achieve the best or desired acceptance ratio and quality of orchestration of the algorithm. This method is summarized by the inputs and output of APO in Figure 3.

In [Nem+15], we showcase two example setups of the algorithm: (i) Deployable Service Graph requests in the ESCAPE framework, where the substrate network, sized several tens of nodes, is simulated by Mininet. Arbitrary service requests can be given to ESCAPE via a graphical interface and any parameter of the algorithm can be adjusted to demonstrate their effect on the mapping results. (ii) An all-round abstract description of a carrier grade network with several tens-to-hundreds of thousands of substrate nodes connected into a real world topology, where the already fine-tuned<sup>3</sup> resource orchestration algorithm can be executed with arbitrary service request. The results of the mapping can be examined on visualizations in both cases.

---

<sup>2</sup>MIP cannot be applied because the objective function which maps vectors from the algorithm parameter space to the scalar numbers (e.g. acceptance ratio) is nor linear.

<sup>3</sup>The fine-tuning process takes too much time to demonstrate its operation at the time and place of the demo.

## 2 Virtualizer3 library

The Virtualizer3 library is the key enabler of a common SI-Or (UNIFY) interface. As SI-Or is the most important interface of the UNIFY architecture appearing at different layers, we propose a common library to be used by all partners for implementing related interfaces. The Virtualizer3 library is an implementation of our Virtualizer YANG model defined in [D3.2a] and further elaborated in [D3.3]. On the one hand, from a YANG data model, it can construct the Python representation containing all necessary elements. On the other hand, the library implements basic operations with Virtualizer objects, such as parse/dump Virtualizer configurations from/to XML structure, get/edit/diff config NETCONF operations, iteration on the structure, and access to given elements.

The most recent stable version of the library is always available from github:

- <https://github.com/Ericsson/unify-virtualizer>

Additionally, we have a project internal repository for development which is available at:

- <https://gitlab.fp7-unify.eu/Robert.Szabo/virtualizer>

### 2.1 Main features

The main goal of the library is to provide a common interface between components from various partners, by realizing the SI-Or UNIFY interface according to the Virtualizer YANG model. Further expectations on the library were:

- Realize the elements of the YANG model by Python Objects
- Parse an entire Virtualizer instance from XML
- Dump an entire Virtualizer instance to XML
- Support main NETCONF abstractions
- Support workflows, like: get config, edit config, diff config
- Support working with full configurations as well as partial updates to existing configurations
- Support easy access to components of an object (e.g. flow entries of a flow table)
- Support walking in the structure (e.g., from a port of a flow entry to the NF to which the port belongs to)
- API to be extensible

The library consists of two files and the object tree is shown in Fig. 4:

**virtualizer3.py** contains the Virtualizer object structure. This is autogenerated from the virtualizer3.yang module

**baseclasses.py** contains the common components for the library, like the base Yang class for generic parsing, comparison, etc.

The main functions of the library are:

**Virtualizer.parse(input):** parse an instance from an XML input (file or string)

**Virtualizer.xmlIO:** dump an instance to an XML string

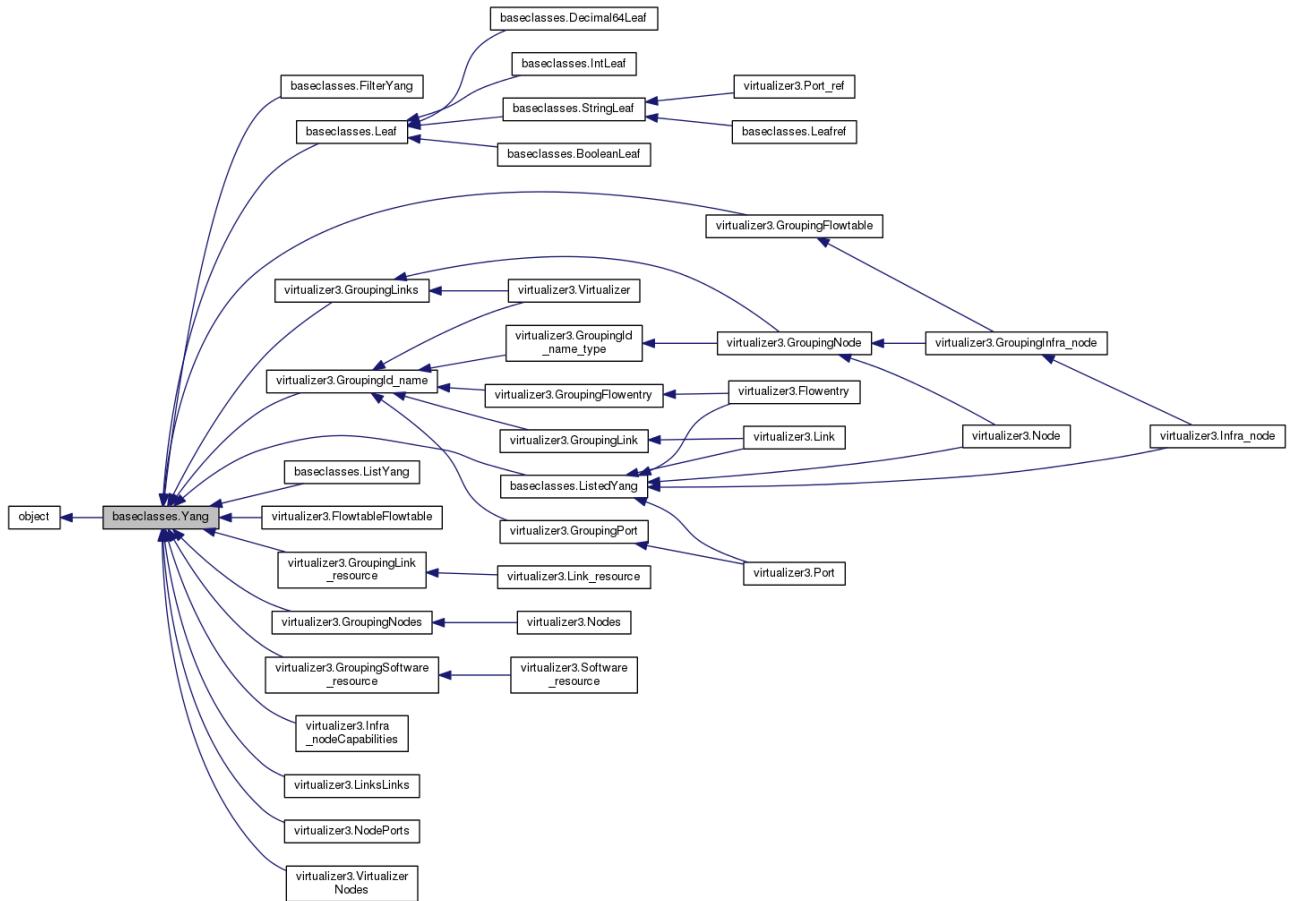


Figure 4: Virtualizer3's object tree

**Virtualizer.reduce(other)**: remove all components which are redundant and the same compared to the other instance

**Virtualizer.bind(other)**: resolve cross references between components, fill in missing parts from the other instance

**Virtualizer.merge(other)**: merge configurations (i.e. merge configuration updates)

The created library has been used in [Son+15] as the interface between the domains of ESCAPE, OpenStack and the Universal Node.

## 2.2 Usage examples

### 2.2.1 Parse from file

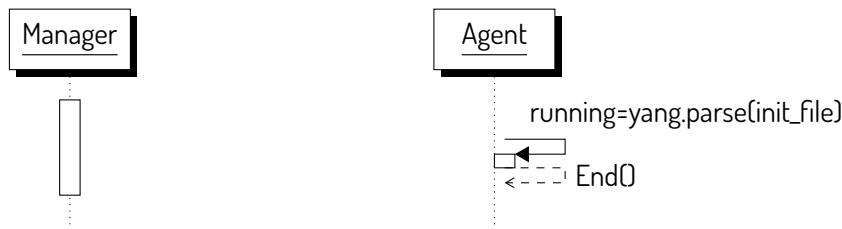


Figure 5: Parse from file

## 2.2.2 Get-config

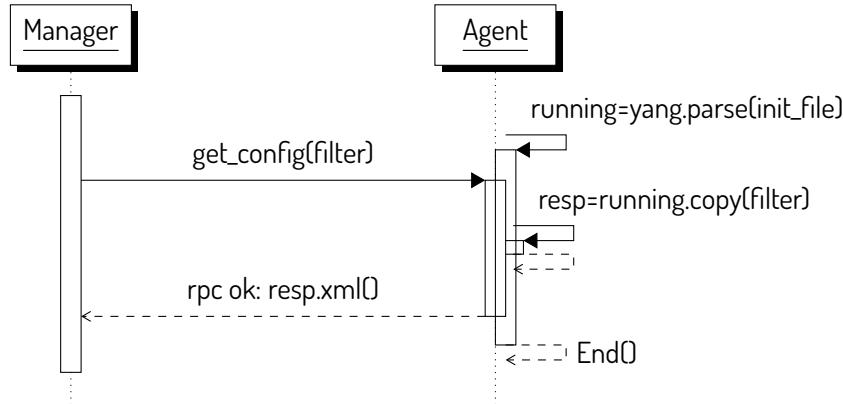


Figure 6: Get-config: filter and return configuration data

## 2.2.3 Edit-config

Workflow:

- `reduce()`: reduce change set to canonical representation;
- `bind()`: /optional/ copy missing references (`leafref`) to the change set.

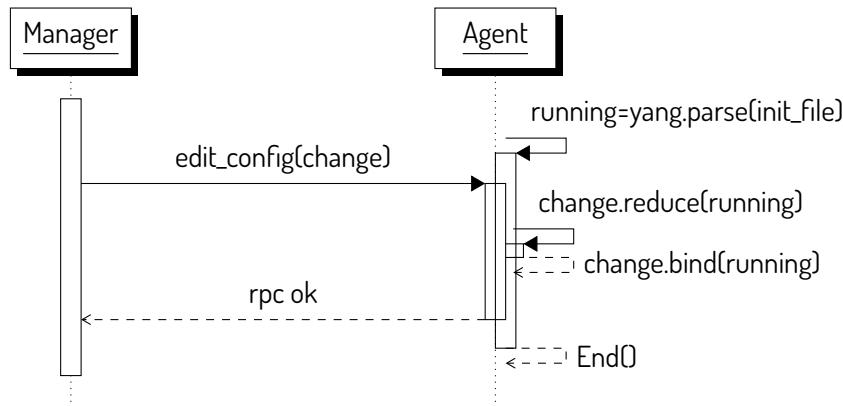


Figure 7: Edit-config: reduce and bind

#### 2.2.4 Edit-config: multiple requests

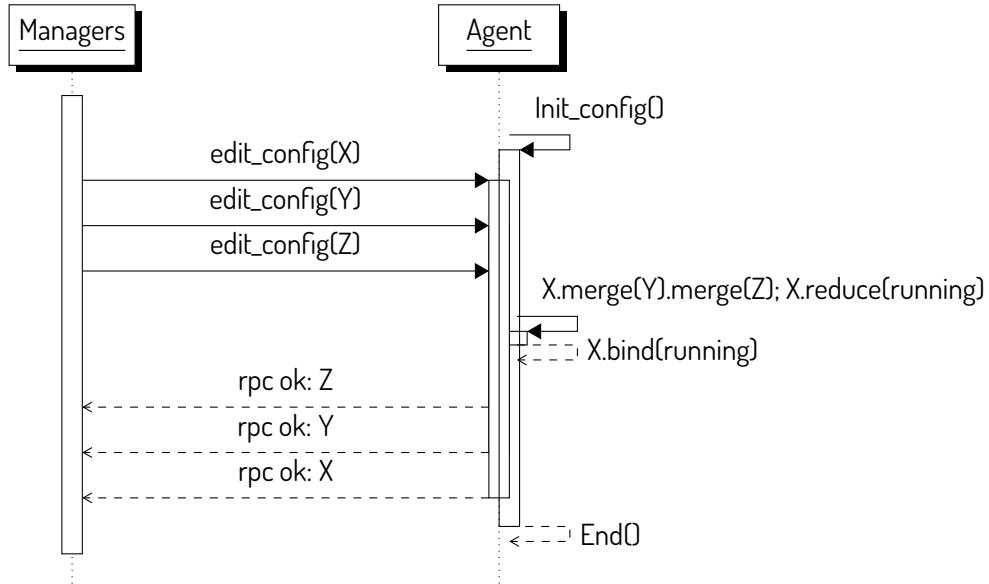


Figure 8: Edit-config: merge change requests

#### 2.2.5 Edit-config and copy-config: diff and patch

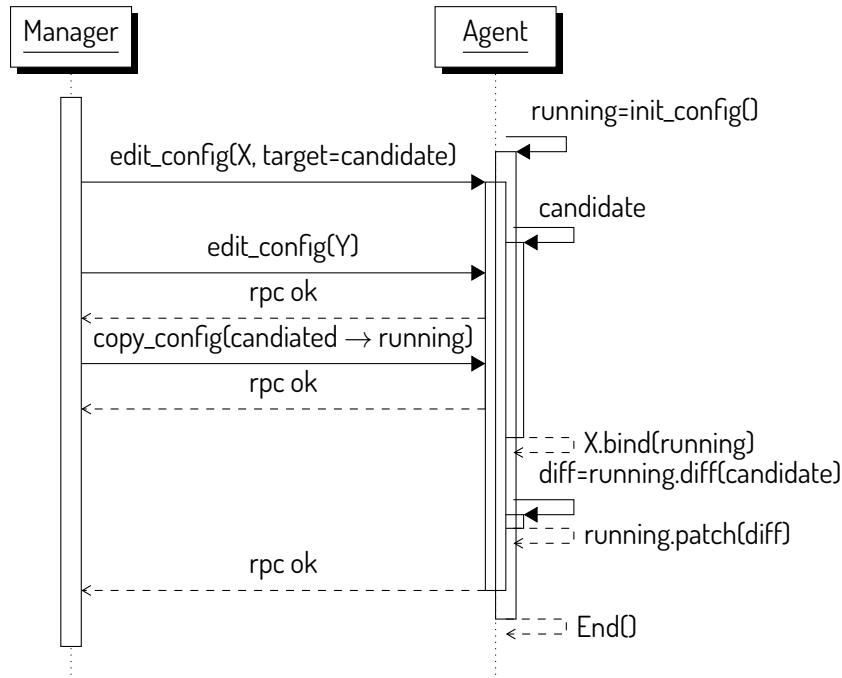


Figure 9: Copy-config: calculating diff and applying diff as patch

### 3 Supported infrastructure domains

ESCAPE supports several infrastructure domains based on the previously presented domain managers and adapters. If we use our novel SI-Or interface, on top of the infrastructure domains, additional components have to be implemented. To make this integration with legacy domains easier, a dedicated library called Virtualizer3 has been implemented. As it is shown in Figure 10, this library is integrated with ESCAPE and we also made it use in the local orchestrator of the OpenStack domain and the Universal Node domain, respectively.

The multi-domain setup shown in Figure 10 has been demonstrated at Sigcomm 2015 [Son+15]. At the infrastructure level, different technologies are supported and integrated with the framework. We kept our previous Mininet based domain orchestrated by a dedicated ESCAPE entity via NETCONF and OpenFlow control channels. Here, the NFs are run as isolated Click processes. As a legacy data center solution, we support clouds managed by OpenStack and OpenDaylight. This requires a UNIFY conform local orchestrator to be implemented on top of an OpenStack domain. The control of legacy OpenFlow networks is realized by a POX controller and the corresponding domain manager with the adapter module. And finally, a proof of concept implementation of our Universal Node concept is also provided. UN local orchestrator is responsible for controlling logical switch instances (of e.g., xDPd) and for managing NFs run as Docker containers. The high performance is achieved by Intel's DPDK based datapath acceleration.

During the demo, we have showcased *i*) how to create joint domain abstraction for networks and clouds; *ii*) how to orchestrate and optimize resource allocation and deploy service chains over these unified resources; *iii*) how we can take advantage of recursive orchestration and NF decomposition.

A detailed illustration of the bottom-up and top-down process flows implemented by ESCAPE is shown in Section 4.

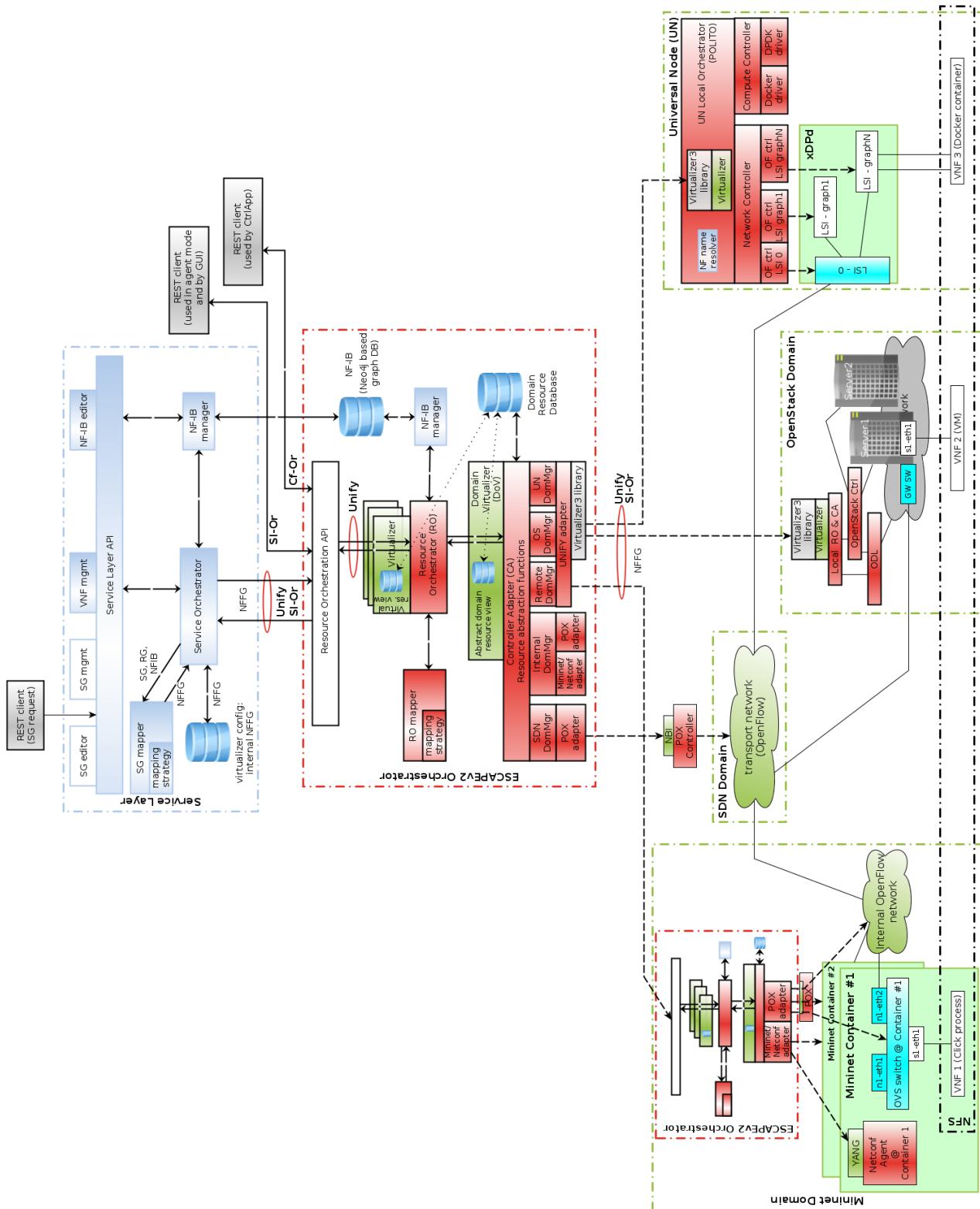


Figure 10: Integrating ESCAPE with different infrastructure domains

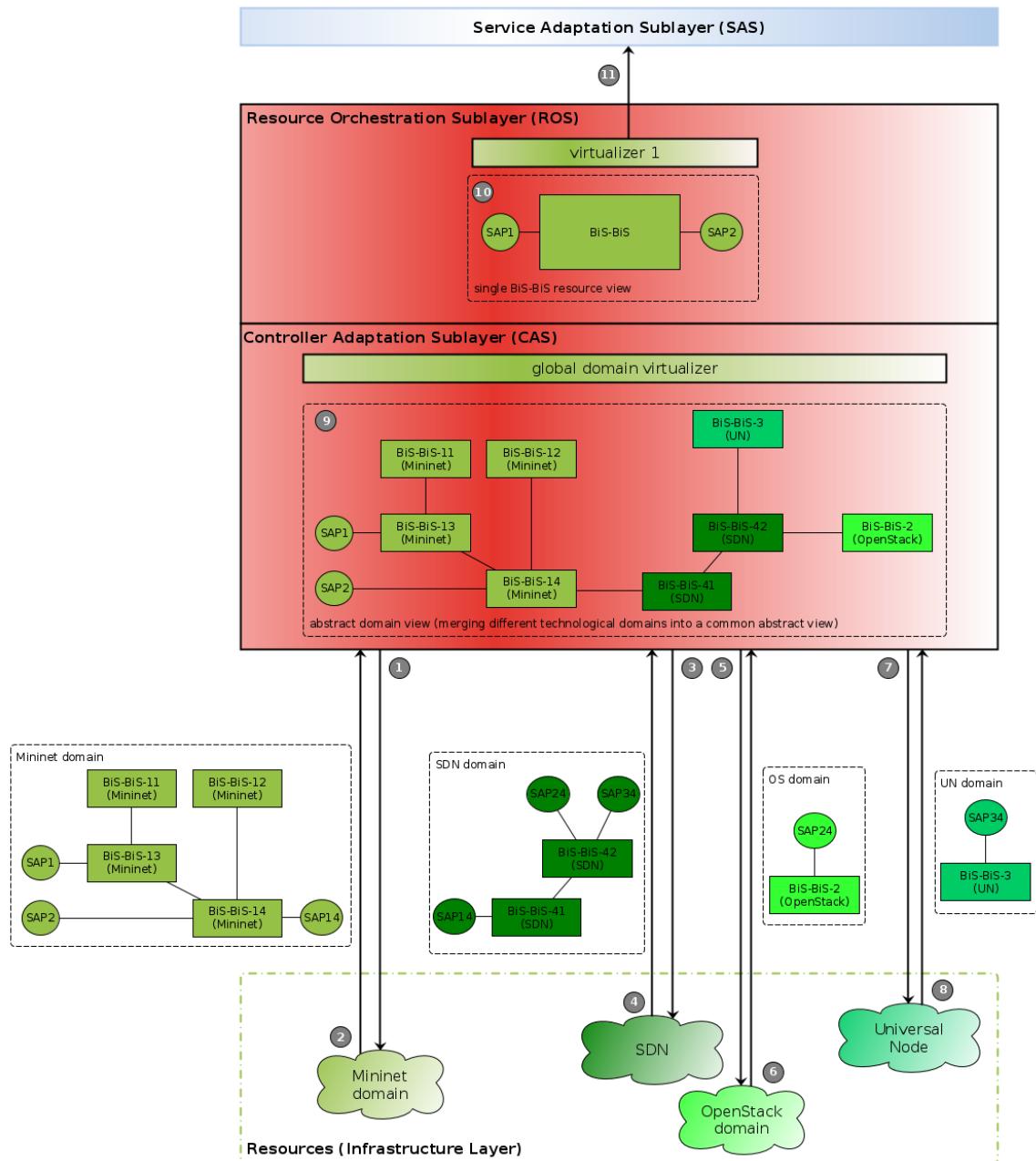


Figure 11: Illustration of the bottom-up process flow in ESCAPE

## 4 Illustrating the operation of ESCAPE

This section is devoted to illustrate the main steps of the bottom-up and top-down process flows supported by ESCAPE, respectively.

### 4.1 Bottom-up process flow

In Figure 11, the current implementation of the bottom-up process flow in ESCAPE is illustrated by a simple example. During the bootstrap phase, ESCAPE gathers the resource information from the available domains and constructs the abstract domain view by merging individual domains. The connections between different domains are identified by inter-domain SAPs with the same id.

---

The steps of the bottom-up process flow are the following:

1. get resource info from Mininet domain
2. send resource info
3. get resource info from SDN domain
4. send resource info
5. get resource info from OpenStack domain
6. send resource info
7. get resource info from UniversalNode domain
8. send resource info
9. construct global abstract domain view
10. construct single BiS-BiS virtualizer view
11. send status info

## 4.2 Top-down process flow

In Figure 12, a simple example is given illustrating the top-down process flow of ESCAPE. A request is formulated as an SG and sent to the REST API of the Service Layer. The Service Orchestrator in this layer maps the request to a single BiS-BiS then sends the configuration to the Orchestration Layer. The Resource Orchestrator maps the service components to its global resource view and updates the configuration of the global domain virtualizer. As a final step, the result given as an NF-FG is splitted according to the domains and sent to corresponding resource agents or controllers or local orchestrators.

The steps of the top-down process flow are the following:

1. send SG request via REST API
2. map the request to the single BiS-BiS virtualizer
3. send edit-config request to the Orchestrator
4. map the service to the global domain virtualizer
5. split the NF-FG according to the domains
6. send given part of the NF-FG to the Mininet domain
7. send given part of the NF-FG to the SDN domain
8. send given part of the NF-FG to the OpenStack domain
9. send given part of the NF-FG to the UniversalNode domain

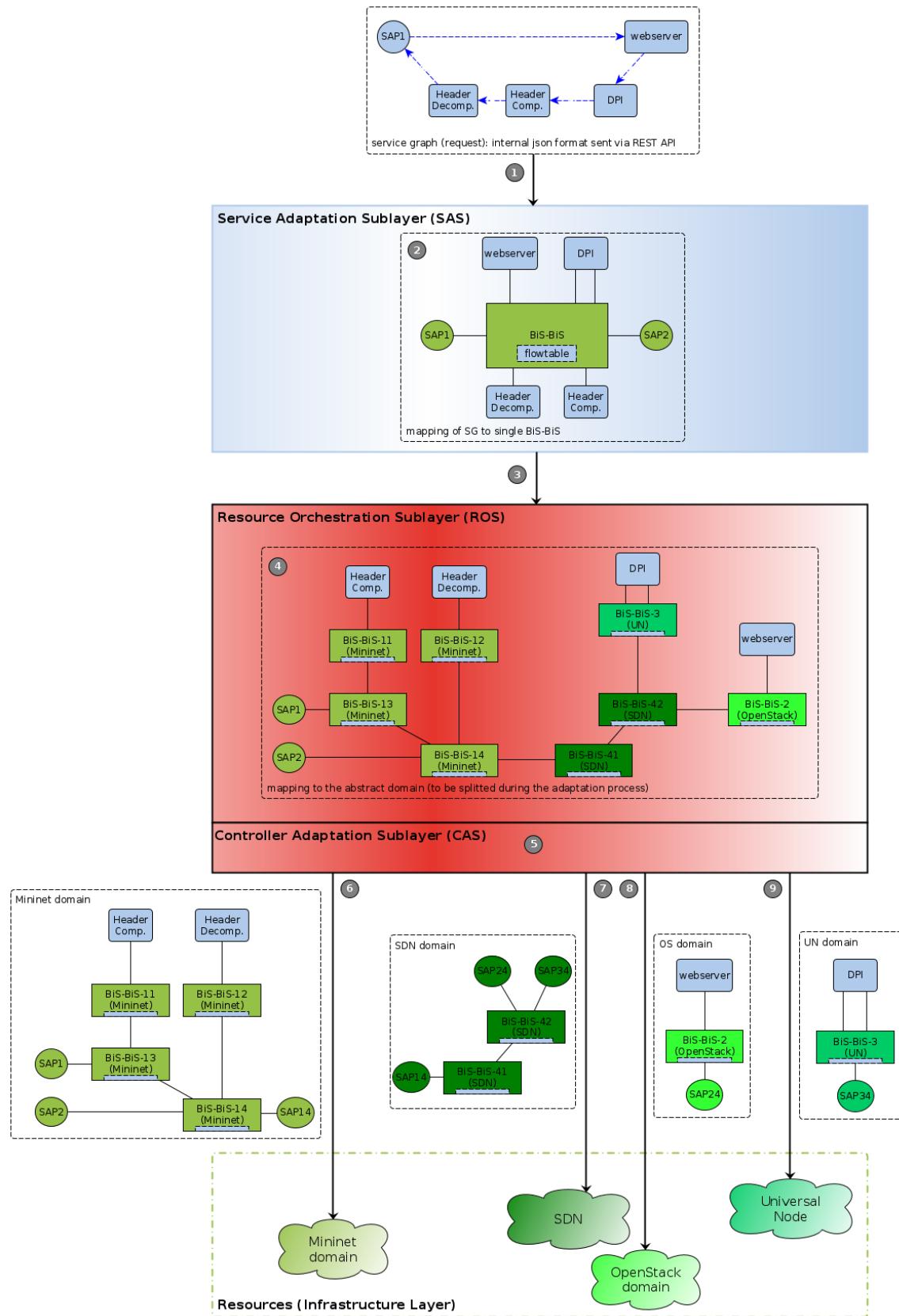


Figure 12: Illustration of the top-down process flow in ESCAPE

## 5 ESCAPE: implementation view

This section provides implementation details on the latest version of ESCAPE. The software framework is organized into the following packages<sup>4</sup>:

- escape.service: implements the Service Adaptation Sublayer (SAS) of the Service Layer (SL) of the UNIFY architecture.
- escape.orchest: implements the components of the Resource Orchestration Sublayer (ROS) which encompasses the resource orchestration related functions of the UNIFY Orchestration Layer (OL).
- escape.adapt: implements the components of the Controller Adaptation Sublayer (CAS) which encompasses the controller adaptation related functions of the UNIFY OL.
- escape.infr: implements a Mininet based infrastructure domain (Infrastructure Layer (IL) of the UNIFY architecture).
- escape.util: includes utilities and abstract classes used by other modules.

The main changes from the previous version reported in [D3.2] are the following:

- A graph-based and internally-used NF-FG library was developed to store and forward the topology resource information polled from different domains and service requests coming from UNIFY users in a transparent and abstract way. This NF-FG implementation exploits the structural benefits of graphs-based representation to offer elemental operations, basic graph algorithms and complementary functions for the main orchestration algorithm.
- The implementation of the Service and Resource Orchestration Layer was refined and upgraded with the Virtualizer-based approach to achieve scalable information hiding between layers and accomplish the multi-purpose role of both local and global orchestrator.
- A general orchestration algorithm was integrated with ESCAPE which is used in the Service Adaptation and Resource Orchestration Sublayers to realize the multi-layer orchestration process flow.
- The most significant changes were made in the Controller Adaptation Sublayer mainly focusing on multi-domain operations. This includes the interaction with different technological domains, polling resource information from available domains, merging multiple domains into the abstract global view, splitting the mapped, global NF-FG according to the domains and sending those parts to the given domains.
- A new package implementing a Mininet based Infrastructure Layer was added to ESCAPE. Some components of this part were available in the previous version of ESCAPE. Now, it is fully integrated with the framework and the corresponding adapters (for controlling NFs via NETCONF) were also added. With this element, we have a standalone development framework supporting the full top-down and bottom-up process flows.

The rest of this section gives low level implementation details of these main points following the structure of ESCAPE packages. First, the static model of a given module is described. Second, the class structure of the implemented

<sup>4</sup>A package contains several modules. But sometimes we use the term module for the whole package as well.

---

layer and the relations between the internal components are illustrated by an UML class diagram. Finally, the cooperation of the introduced architectural parts and the interaction steps between them is highlighted through a specific case study. This example shows the main steps of the UNIFY top-down process flow.

The final subsection presents the main elements of our internally used NF-FG library.

## 5.1 Service module

The Service module represents the Service (Graph) Adaptation Sub-layer (SAS) of the UNIFY SL described in [D2.2]. Additionally, it contains a REST API on top of the layer for unified communication with upper components such as UNIFY actors via a GUI or other standalone applications. The static class structure of this main module is shown in Figure 13.

One of the main logical parts of Service module is the REST API. The REST API provides a unified interface based on the HTTP protocol and the REST design approach. The RESTServer class encompasses this functionality. It runs a custom-implemented HTTP server in a different thread and initializes the ServiceRequestHandler class which defines the interface of the relevant UNIFY reference point (namely the U-SI interface). The class consists of purely the abstract UNIFY interface functions therefore it can be replaced without the need to replace or modify other components. Via this API, a service request can be initiated (with sg function) or the resource information provided by a Virtualizer can be queried (with topology function). The Virtualizer object of the Service module is created and initiated during the bootstrap process when it gathers the resource information from the lower layer (OL). The Virtualizer in the SL offers a single BiS-BiS virtual view by default. This is generated from the global resource view of the OL (DoV component). The RESTServer uses a RequestCache instance to register every service in order to follow the status of the initiated services.

In order to separate the UNIFY API from the REST behaviour, the general functionality of HTTP request handling is defined in an abstract class called AbstractRequestHandler. This class contains the basic common functions which

- parse the HTTP requests,
- split and interpret the URLs according to the REST approach to determine the UNIFY API function need to be called,
- parse the optional HTTP body as the parameter with respect to security requirements,
- and delegate the request process to the actual module-level API function with the processed parameters in a common form (as an NF-FG).

The other main part of Service module represents the Service Adaptation Sub-layer. The main entry and exit point is the ServiceLayerAPI class. This element realizes the actual interface of the SAS sub-layer and proceeds the calls comes from external source (e.g. REST API, file, other modules) to the appropriate subcomponents. The general behaviour for each top-layer module of ESCAPE is defined in the AbstractAPI class. This class contains the

- basic and general steps related to the control of module's life cycle,
- definition of dependencies on other components,
- initiation and tear down of internal elements,
- general interface for interaction with other modules and external actors,
- and the management of communication between internal elements.

---

According to these functions the role of the actual API classes derived from `AbstractAPI` is threefold. First, it hides the implementation and behaviour of POX to make the modules' implementation more portable and easily changeable. Second, it handles the module dependencies to grant a consistent initialization process. Third, it handles the event-driven communication between modules so internal elements only have to know and use the common functions of the derived `AbstractAPI` class defined in each top-level module. Furthermore, with these functionalities provided by the `AbstractAPI` base class the main modules of ESCAPE can achieve a loosely coupled, transparent communication and easily adjustable module structure.

The central point of the Service Layer is the `ServiceOrchestrator` class derived from the common `AbstractOrchestrator` base class. The base class initializes, contains and handles the internal elements which are involved in the highest level of the service chaining process. The derived orchestrator class also supervises the supplementary functions related to the service orchestration such as managing, storing and handling service requests, handling virtual resource information and choosing the algorithm for service-level mapping.

The service request managing functionality is realized by the `SGManager` wrapper class which offers a common interface for handling and storing service requests in a platform and technology independent way. The format in which the service requests are stored is the same internal NFFG representation class (called NFFG) which is used to store the polled resource information.

The `VirtualResourceManager` class handles the virtual resource information assigned to the service module in the same way as the `SGManager` for Service resources. In the background the resource information is not stored explicitly in a standalone NFFG instance. Instead the manager class have a reference to a dedicated `Virtualizer` element, which can generate the resource information on the fly. Due to the wrapper classes the storing format can be modified easily to use only NFFG representation and a fully separated module design can be achieved. This manager class as all Manager classes in ESCAPE hides the actual format of the stored resources and provides the opportunity to change its implementation transparently. The assigned `Virtualizer` of the SL is the default `SingleBiSBiSVirtualizer` inherited from the common `AbstractVirtualizer` class. This `Virtualizer` offers the trivial one BiS-BiS virtualization which is generated from the collected global resource information (DoV) and consists of only one infrastructure node with the aggregated resource values and the detected SAPs. The generation of the resource information and the requesting of the global resource view are executed on the fly, during the Service Layer orchestration process.

The orchestration steps are encompassed by the `ServiceGraphMapper` class, which pre-processes and verifies the given information and provides it in the appropriate format for the mapping algorithm.

The mapping algorithm is defined in a separate element for simplicity and clarity. The trivial service level orchestration which is carried out by the same mapping algorithm used in the Orchestration module is executed by the `DefaultServiceMappingStrategy` class. The general interfaces for the mapper and strategy classes are defined in the `AbstractMapper` and `AbstractStrategy` classes.

The communication between the elements inside the modules is based on events. The Event classes in the layer components represent the different stages during the ESCAPE processes. The event-driven communication relies on POX's own event handling mechanism, but every communication primitive is attached to well-defined functions for the purpose of supporting other asynchronous communication forms, e.g., different implementations of event-driven communication based on Observer design pattern, Asynchronous Queuing or Message Bus architecture based on ZeroMQ.

---

### Steps of a service request in the Service module

1. On the top of the UNIFY hierarchy a UNIFY actor can request the virtual view / resource topology of the Service Layer to collect available resource information or to show the topology on a GUI application.
2. The service request can be given via the REST API in a HTTP message. The function is defined in the URL (the general sg function along with POST HTTP verb) with a formatted body as an NF-FG in JSON format.
3. The message is processed; the optional parameters are parsed and converted concerning the HTTP verb and delegated to the sg() function which is part of the UNIFY U-S1 API representation in the ServiceRequestHandler class. The REST API also caches the service request.
4. The main ServiceLayerAPI class receives the UNIFY API call and forwards to the central ServiceOrchestrator element.
5. The orchestrator saves the generated Service Graph in the SGManager with internal NF-FG format, obtains the resource information via the assigned OneBiSBiSVirtualizer with the help of the VirtualResourceManager and invokes the SGMapper in order to start the mapping process.
6. The SGMapper requests the resource information from the given OneBiSBiSVirtualizer in the NF-FG format, validates the service request against the resource info in respect of sanity and syntax, perform the configured pre- and post-processing tasks and finally invokes the actual mapping algorithm of the DefaultServiceMappingStrategy.
7. The DefaultServiceMappingStrategy calls directly the configured orchestration algorithm and handles any mapping errors.
8. After the mapping process is finished, the actual Strategy element returns the outcome in an SGMappingFinished event, which is processed by the module API class and proceeds the given NF-FG to the lower layer for instantiation via a general function. The instantiation notification is realized via the InstantiateNFFGEvent class.

## 5.2 Orchestration module

The Orchestration module represents the Resource Orchestration Sub-layer (ROS) of the UNIFY OL. The communication with the upper and lower layer is managed by the POX event mechanism as it is implemented in the Service module. The static class structure of this main module is shown in Figure 14. The structure of this module, the separation of internal components and their relations are designed in compliance with the Service module as precisely as possible in order to support the transparency and consistency of the ESCAPE architecture.

The Orchestration layer has its own REST API. With the ROSAgentRequestHandler class, it can provide resource information for third-party applications or a GUI and implements the UNIFY interface for the local orchestration mode (edit-config function). It can also be used for requesting the resource information of the ROS layer explicitly (get-config function). Here, we use a non-filtering Virtualizer, namely the GlobalViewVirtualizer which skips the resource virtualization and offers the global domain resource completely to the upper layers or external entities.

Additionally the Orchestrator module can be initiated with a special REST API to implement the Cf-Or interface. With this extension, ESCAPE will support elastic NFs and the special elastic router/firewall use-case proposed by UNIFY project. The interface functions are defined in the CfOrRequestHandler class.

---

The main interface of the Orchestration module is realized by the `ResourceOrchestrationAPI` class. It has a similar role than the `ServiceLayerAPI` in the Service module. It manages the module's life cycle, handles internal and external communications and translates calls from events to class functions. Based on the external event-driven communication, the `ResourceOrchestrationAPI` realizes the relevant `SI-Or` and `Cf-Or` interfaces.

The central component of this module is the `ResourceOrchestrator` which is responsible for the orchestration at the level of global domain view (DoV). It initializes, contains and controls internal module elements and gathers needed information similarly to the `ServiceOrchestrator` class.

The management of requested and already installed NF-FG instances is performed by the `NFFGManager` class. The manager class uses the internal `NFFG` representation as the storage format.

The orchestration steps are encompassed by the `ResourceOrchestrationMapper` class similarly to the mapper used in the Service module. The mapping algorithm of this layer is defined in a derived `AbstractStrategy` class, namely in the `ESCAPEMappingStrategy` class. It uses the request stored in an NF-FG and the actual resource information. The Network Function descriptions can be requested via a wrapper class, i.e., the `NFIBManager`, which hides the implementation details of the NF-IB and offers a platform-independent interface. The current version of the NF-IB is implemented in a neo4j database. This manager class is provided for the orchestration-level mapping algorithm by default.

An important task of this module is the proper handling of the virtualizers and virtual resource views. The orchestration module is responsible for the creation, assignment and storing of virtual resource views. The functionality of these virtual views is encompassed by the `AbstractVirtualizer` base class. This class offers a general interface for the interaction with the Virtualizers and contains the common functions, such as generating the virtual resource information in our internal NF-FG format. The derived classes of the `AbstractVirtualizer` represents the different kind of Virtualizers defined in the UNIFY architecture and contains the metadata for the resource information filtering. The derived classes such as `SingleBiSBiSVirtualizer` and `GlobalViewVirtualizer` represent the virtual views and offer the virtualized resource information for the assigned upper layer(s). The `DomainVirtualizer` class is a special kind of Virtualizer which represents the abstract global resource view, namely the Domain Virtualizer (DoV) created by and requested from the lower layer. The Virtualizer instances are managed and created by the `VirtualizerManager` class. This manager class also stores the `DomainVirtualizer` instance which is used for the creation of the virtual views.

The policy enforcement functions which are closely related to the Virtualizers are defined in the `PolicyEnforcement` class. This class implements the enforcement and checking functions. In every case when a derived `AbstractVirtualizer` instance is created the `PolicyEnforcement` class is attached to that Virtualizer in order to set up the policy related functionality automatically. The attachment is performed by the `PolicyEnforcementMetaClass`. The policy enforcement functionality which realized by the previous classes follows the Filter Chain approach associated with the functions of the Virtualizers. That design allows defining and attaching a checking or enforcing function before and/or after the involved function of a Virtualizer is invoked by other internal module components.

### Steps of a service request in the Orchestration module

The input parameter is an `InstantiateNFFGEvent` event which is raised by the `ServiceLayerAPI` at the end of the process flow presented for the Service module. Beside this internal event, service requests can be received on the REST APIs as well, but the process flow is similar.

1. The NF-FG request can be originated from an upper layer in a form of an internal event or from a separated entity interacting via the REST API.

- (a) If the NF-FG configuration comes from the Service module, the triggering event called `InstantiateNFFGEvent` is handled by the `ResourceOrchestrationAPI` class which is the communication point between the internal components and other top modules. The event contains the mapped service description in the format of the internal NF-FG. Based on the type of the event, a dedicated event handler is invoked. These handlers in the actual top module class represent SI-Or API.
  - (b) The requests received from other external entities are parsed and processed by `ROSAgentRequestHandler` or `CfOrRequestHandler` and then forwarded to the `ResourceOrchestrationAPI` directly.
2. The request is delegated to the central `ResourceOrchestrator` via the corresponding API function. The process is similar to the service request process of the Service module described in the previous section.
  3. The orchestrator saves the generated NF-FG using the `NFFGManager` wrapper (using the internal NF-FG format); obtains the global resource view as a `DomainVirtualizer` by invoking the `VirtualizerManager` class.
  4. `ResourceOrchestrator` invokes the `orchestration()` function of the `ResourceOrchestrationMapper` class in order to initiate the NF-FG mapping process.
  5. The orchestration process requests the global resource information via the `DomainVirtualizer` and invokes the actual mapping algorithm of the `ESCAPEMappingStrategy`. The validation of the inputs of the mapping algorithm can be performed by the `ResourceOrchestrator`, as well.
  6. The `ESCAPEMappingStrategy` uses the `NFIBManager` to run the algorithm and returns with the mapped NF-FG in the common NF-FG format in an asynchronous way with the help of the `NFFGMappingFinishedEvent`.
  7. The event is handled by the `ResourceOrchestrationAPI` class and it proceeds the on-going service request by invoking a general communication function.
  8. The mapped NF-FG is forwarded to the lower layer via the `InstallNFFGEvent`.

### 5.3 Adaptation module

The Adaptation module represents the Controller Adaptation Sublayer (CAS) of UNIFY OL. The communication with the upper layer is managed by the POX event mechanism similarly to other modules. The static class structure of the Adaptation module is shown in Figure 15. The structure is in line with the previously introduced top API modules.

The main interface of the Adaptation module is realized by the `ControllerAdaptationAPI`. Its functions and responsibilities are identical to the other top API classes derived from `AbstractAPI`. This API realizes the corresponding UNIFY reference point, more exactly, the Or-Ca interface.

The central component of this module is the `ControllerAdapter`. This component initializes the domain handler component with the help of the `ComponentConfigurator` class. It initializes, contains and handles the internal `DomainManager` elements based on the global configuration of ESCAPE. Here, we follow the Component Configurator design pattern. The main tasks of the `ControllerAdapter` class can be split into two major parts.

First, it handles the incoming NF-FG instances coming from the upper Orchestration module. For this task, the `ControllerAdapter` contains the functions for processing the mapped NF-FG instances, splitting into subsets of NF-FG descriptions based on the initiated domain managers. Control of the installation of the sub-parts is carried out by the `ComponentConfigurator` which forwards the relevant NF-FG part to the appropriate domain managers. Here, we use the same internal NF-FG format.

---

Second, it handles the domain changes originated from the lower layer (IL). For this task, the ControllerAdapter initiates and manages the domain managers derived from the `AbstractDomainManager` base class. This base class contains the main functionality for the domain management, such as polling the registered domain agents, parsing and converting the collected domain resource information into the internal NF-FG format and handling the topology changes. Every domain has its own domain manager. We have implemented the `InternalDomainManager` for our internal, Mininet based infrastructure, the `UniversalNodeDomainManager` for the remote management of Universal Nodes, the `OpenStackDomainManager` for the remote management of an OpenStack domain, the `SDNDomainManager` for OpenFlow domains, and the `RemoteESCAPEDomainManager` for a remote domain controlled by ESCAPE (recursive orchestration via the SI-Or interface).

For the protocol specific communication with the domain agents (such as NETCONF-based RPCs or REST based requests in XML format), the domain managers initiate and use adapter classes inherited from the `AbstractESCAPEAdapter` class. This base class defines the common management functionality and also offers a general interface for the ControllerAdapter. The adapters wrap and hide the different protocol specific details and give a general and reusable way for the interactions with domain agents, e.g., adapters inherited from the `AbstractRESTAdapter` class for using RESTful communications or adapters inherited from `AbstractOFControllerAdapter` class for managing SDN / OpenFlow controllers. These adapters are also responsible for the process and conversion of the received raw data into the internal NF-FG representation using the `NFFG` class.

The domain / topology information from the lower layers is stored via the `DomainResourceManager` wrapper class which is managed by the ControllerAdapter, too. The ControllerAdapter implements the connection between the domain adapters and the domain resource database. This manager class manages the global resource information and contains the functionality of setting, merging and updating different domain resource information (in the internal NF-FG format) provided by the domain managers.

The `DomainResourceManager` class offers an abstract global view of the provisioned elements hiding the physical characteristics via the `DomainVirtualizer` class. The `DomainVirtualizer` is inherited from the `AbstractVirtualizer`, therefore the common Virtualizer interface can be used to interact with that element. The `DomainVirtualizer` stores the abstract global resource information in the internal NF-FG format and also contains the algorithm for merging NF-FG objects received from different domains.

### Steps of a service request in the Adaptation module

The input parameter is an `InstantiateNFFGEvent` event which is raised by the `ResourceOrchestrationAPI` at the end of the process flow presented for the Orchestration module.

Before the orchestration process the ControllerAdapter initiates the domain managers with the help of the `ComponentConfigurator`. The managers collect the resource information using the ESCAPE adapters which is converted and merged into one global resource view supervised by the `DomainVirtualizer` class.

1. The triggering event called `InstallNFFGEvent` is handled by the `ControllerAdaptationAPI` class. The event contains the mapped NF-FG generated by the Orchestration module. Based on the type of the event, a dedicated event handler is invoked (Or-Ca interface).
2. The request is delegated to the corresponding function of the central ControllerAdapter class.
3. The ControllerAdapter uses its own splitting algorithm to split the given orchestrated service request according to the domains. The domains are defined from the registered domain managers.

- 
4. The ControllerAdapter initiates the installation process in which it uses the ComponentConfigurator component to notify the domain managers and forwards the relevant NF-FG parts.
  5. The domain managers install the given NF-FG using the ESCAPE adapters and update their topology information.
  6. When the adapters finish, the global resource view is updated by the DomainResourceManager and an InstallationFinishedEvent is raised by the top API class to notify the upper layers about the successful service request process. If the Infrastructure module was initiated the InternalDomainManager will start the network emulation and deploy the given NF-FG part.

The changes of the distinct domains are propagated upwards via the Virtualizer instances with the help of the DomainChangedEvent which is raised by the actual domain adapter classes. If a top module does not have an instantiated Virtualizer, a specific event is raised to request the missing Virtualizer. The process is the following.

1. If an ESCAPEVirtualizer is missing in the VirtualResourceManager of the Service module, then a MissingVirtualViewEvent is raised which is forwarded to the Resource module via the ServiceLayerAPI.
2. The ResourceOrchestrationAPI receives the event and applies for an ESCAPEVirtualizer at the VirtualizerManager.
3. If the needed Virtualizer does not exist yet, the ESCAPEVirtualizer is generated by the VirtualizerManager using the DomainVirtualizer.
4. If the DomainVirtualizer is not available for the VirtualizerManager, a GetGlobalResInfoEvent is raised to request the missing DoV.
5. The event is forwarded to the ControllerAdaptationAPI which responds the requested DomainVirtualizer in a GlobalResInfoEvent.
6. The event is handled by the ResourceOrchestrationAPI; the DomainVirtualizer is extracted from the event and stored in the VirtualizerManager.
7. The requested ESCAPEVirtualizer is generated by the VirtualizerManager using the DomainVirtualizer and returned via a VirtResInfoEvent to the ServiceLayerAPI which stores the Virtualizer in the VirtualResourceManager.

## 5.4 Infrastructure module

The Infrastructure module provides a simple implementation for an IL of the UNIFY architecture. The communication with the upper layer is managed by the POX event mechanism as in the case of the other modules. The static class structure of the Infrastructure module is shown in Figure 16.

The main interface of the Infrastructure module is realized by the InfrastructureLayerAPI which is derived from the AbstractAPI class.

The API class has the reference to the emulated Mininet-based network object which is realized by the ESCAPENetworkBridge class. The bridge class defines the top interface for the management of the network and hides the Mininet-specific implementations. The class is also responsible for the cleanup process to remove any remained temp files or configuration which Mininet cannot remove.

---

The building of the emulated network is carried out by the `ESCAPENetworkBuilder` class. This class can use predefined `Topology` class similar to the `Topo` class in Mininet to build the network. The necessary functions are defined in the `AbstractTopology` class. The builder class has the ability to parse a resource graph stored in the internal NF-FG format and build the Mininet network based on this resource representation.

ESCAPE uses a modified version of Mininet which was extended with specific Node types such as the `EE` (execution environment) or `NetconfAgent` class. The created network uses a specific `RemoteController` class, namely the `InternalControllerProxy` by default to connect the control channel of the initiated OpenFlow switches directly to the `InternalPOXAdapter` of the Adaptation module.

## 5.5 NF-FG module

In ESCAPE, we use an internal NF-FG model and representation to store SG, NF-FG and resource view in a common format. This model is an evolution of the models presented in [D3.2]. The goal of the refinement was to give a general graph representation which is in line with several embedding algorithms. We have described the YANG data model, and its tree representation is shown in Figure 17. It is worth noting, that regarding the main elements, there is a one-to-one mapping between the virtualizer model summarized in [D3.3] and the model used internally. We have defined three different types of nodes and edges, and additional metadata on the NF-FG itself. More specifically, `node_nfs`, `node_saps` and `node_infras` are lists storing NF, SAP and infrastructure (Big Switch with Big Software (BiS-BiS)) nodes, respectively. While `edge_links`, `edge_sg_nexthops` and `edge_reqs` are lists for describing static and dynamic infrastructure links (static links: BiS-BiS→BiS-BiS, SAP→BiS-BiS; dynamic links: NF→BiS-BiS), SG links (NF→NF, SAP→NF) and virtual links defining requirements between NFs (NF→NF, SAP→NF, NF→SAP).

The static object model of our internal NF-FG implementation, including the classes and their relations, is shown in Figure 18.

The functionality and attributes of NF-FG elements are defined in a hierarchical structure. The common attributes are defined in the `Element` class. The main base classes, namely the `Link` and `Node` classes implement the main functionality for the node and link instances of the graph representation. The specific `Node` and `Edge` classes define correspondingly the specific functions. For the complex attributes, particular classes are created, such as the `Flowrule` or the `NodeResource` classes. These main classes are grouped and stored in lists by the `NFFGModel`. This container class is only used in case of persisting or parsing the NF-FG representation.

Our NF-FG implementation uses the `MultiDiGraph` class of the `networkx` library to store the specific link and node classes. The `MultiDiGraph` object is wrapped by the `NFFG` class. The `NFFG` class defines the interface of the internal NF-FG representation, contains the helper functions for building NF-FG, graph operation, etc. This class also implements a workaround which is responsible for ensuring the `MultiDiGraph` class to use the own `Edge` and `Node` classes.

## 5.6 Implemented interfaces

Finally, in Table 1, we summarize our interfaces implementing the abstract interfaces which were highlighted in [D3.3] and defined in [D2.2]. Some of them are available from external entities via REST APIs, others are only internal interfaces. The name of the functions and parameters are also indicated.

Table 1: Implemented interfaces

intf.	functions	external	internal
U-SI	list SG	SAS REST API: escape/topology (by ServiceRequestHandler)	sas_API.py module: ServiceLayerAPI. api_sas_get_topology()
	request / release / update SG	SAS REST API: escape/sg (by ServiceRequestHandler)	sas_API.py module: ServiceLayerAPI. api_sas_sg_request(sg)
	list NFs from NFIB	-	nfib_mgmt.py module: NFIBManager.getNF(nf_id)
	add / remove / update NF in NFIB	-	nfib_mgmt.py module: NFIBManager. {add,remove,update}NF(nf)
SI-Or	instantiate / tear down / change NF-FG	ROS REST API: escape/edit-config (by ROSAgentRequestHandler)	ros_API.py module: ResourceOrchestrationAPI. api_ros_edit_config(nffg)
	get / send virtual resource info	ROS REST API: escape/get-config (by ROSAgentRequestHandler)	ros_API.py module: ResourceOrchestrationAPI. api_ros_get_config()
Or-Ca	instantiate / tear down / change NF-FG	-	cas_API.py module: ControllerAdaptationAPI. _handle_ InstallNFFGEVENT(event)
	get / send virtual resource info	-	cas_API.py module: ControllerAdaptationAPI. _handle_ GetGlobalResInfoEvent(event)
Cf-Or	instantiate / tear down / change NF-FG	Cf-Or REST API: cfor/edit-config (by CfOrRequestHandler)	ros_API.py module: ResourceOrchestrationAPI. api_cfor_edit_config(nffg)
	get / send virtual resource info	Cf-Or REST API: cfor/get-config (by CfOrRequestHandler)	ros_API.py module: ResourceOrchestrationAPI. api_cfor_get_config()

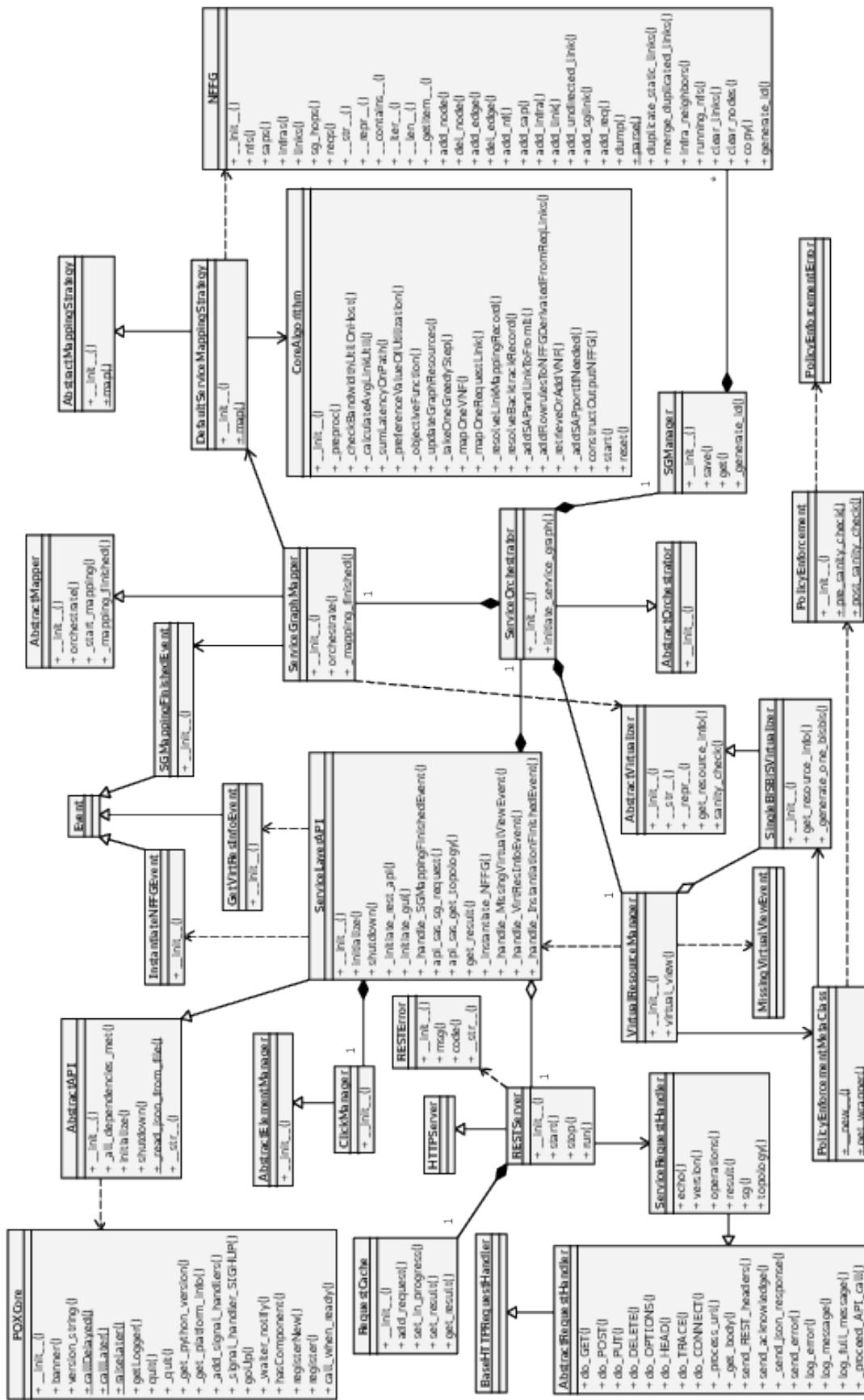


Figure 13: Class diagram of the Service module

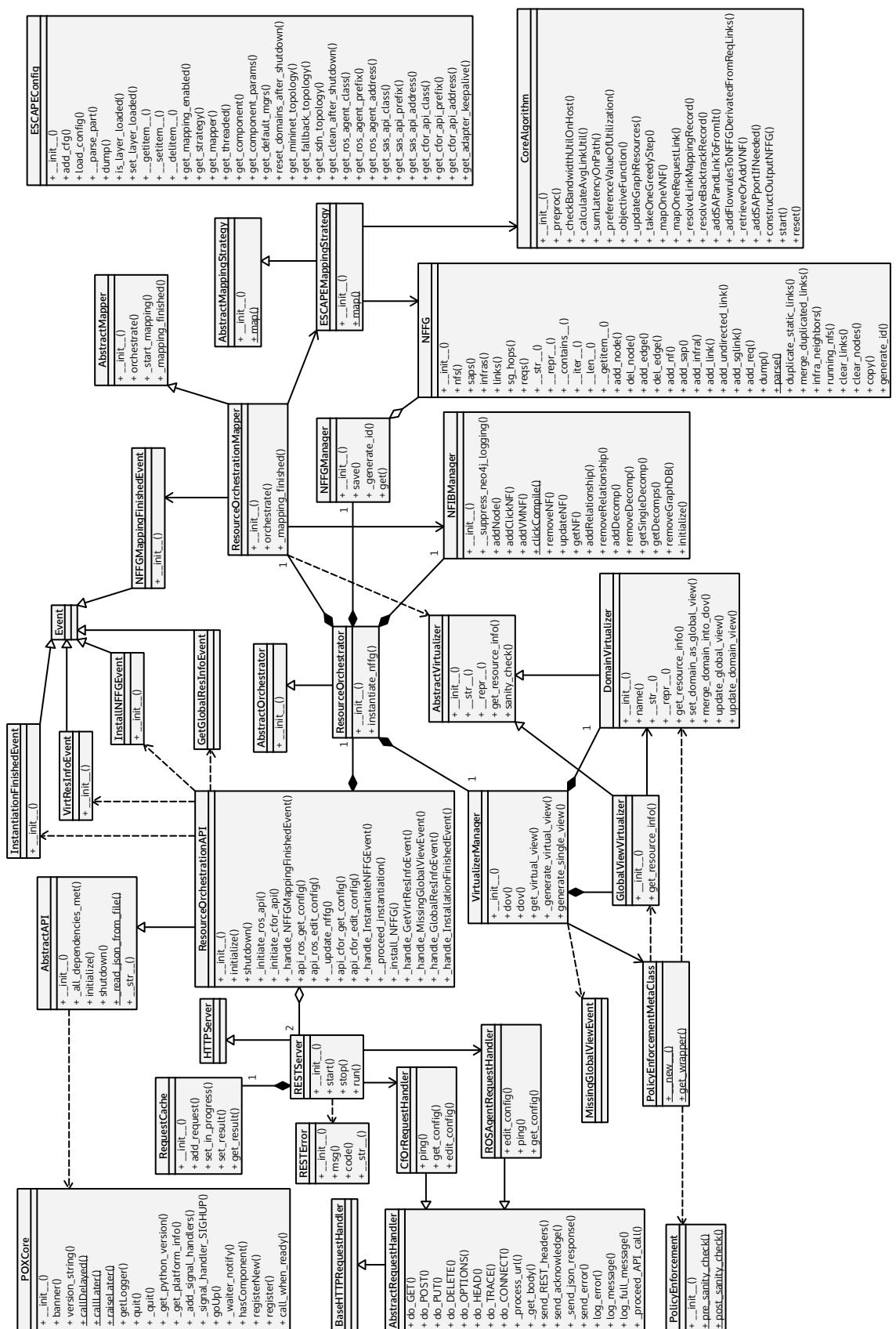


Figure 14: Class diagram of the Orchestration module

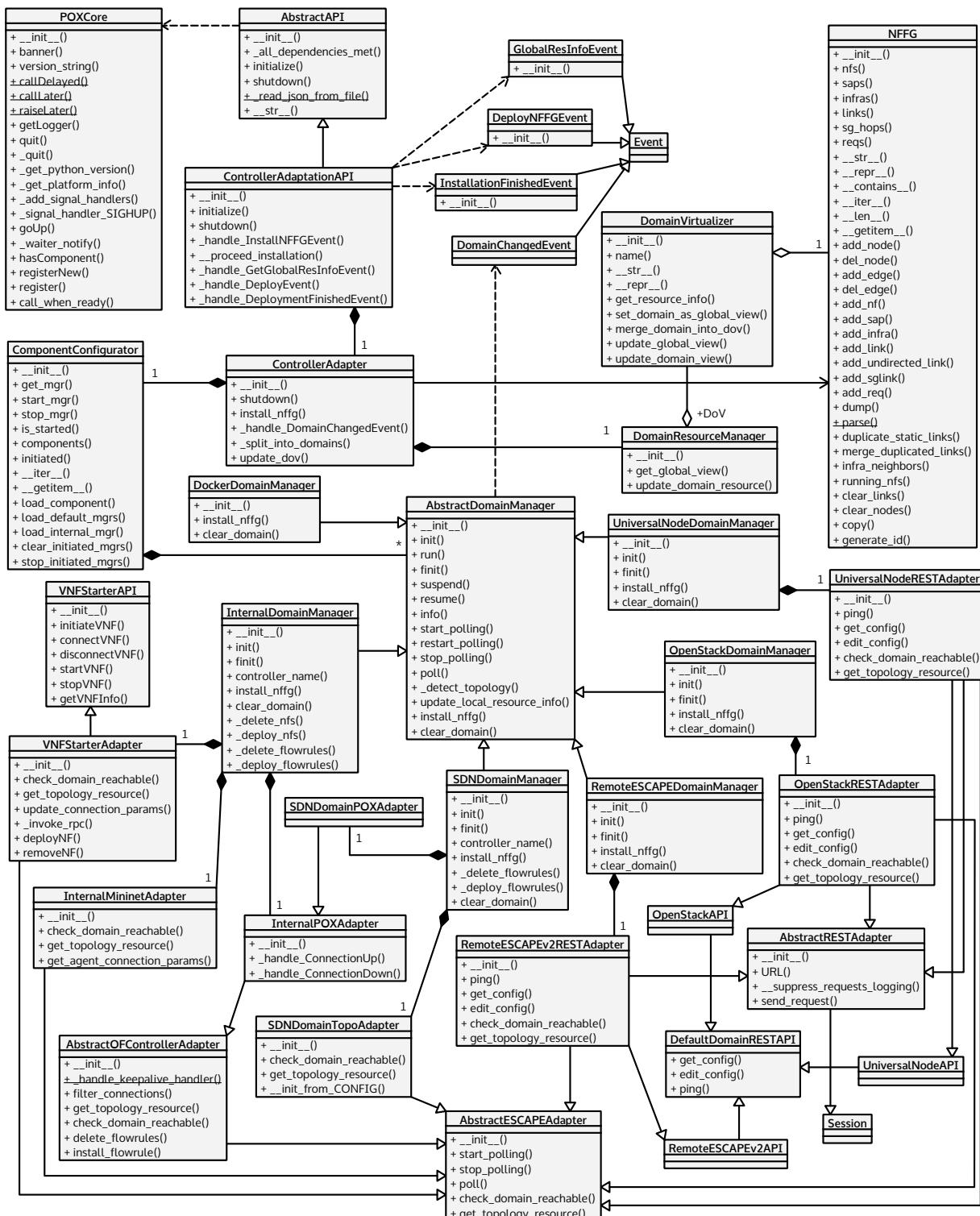


Diagram: CAS class diagram Page 1

Figure 15: Class diagram of the Adaptation module

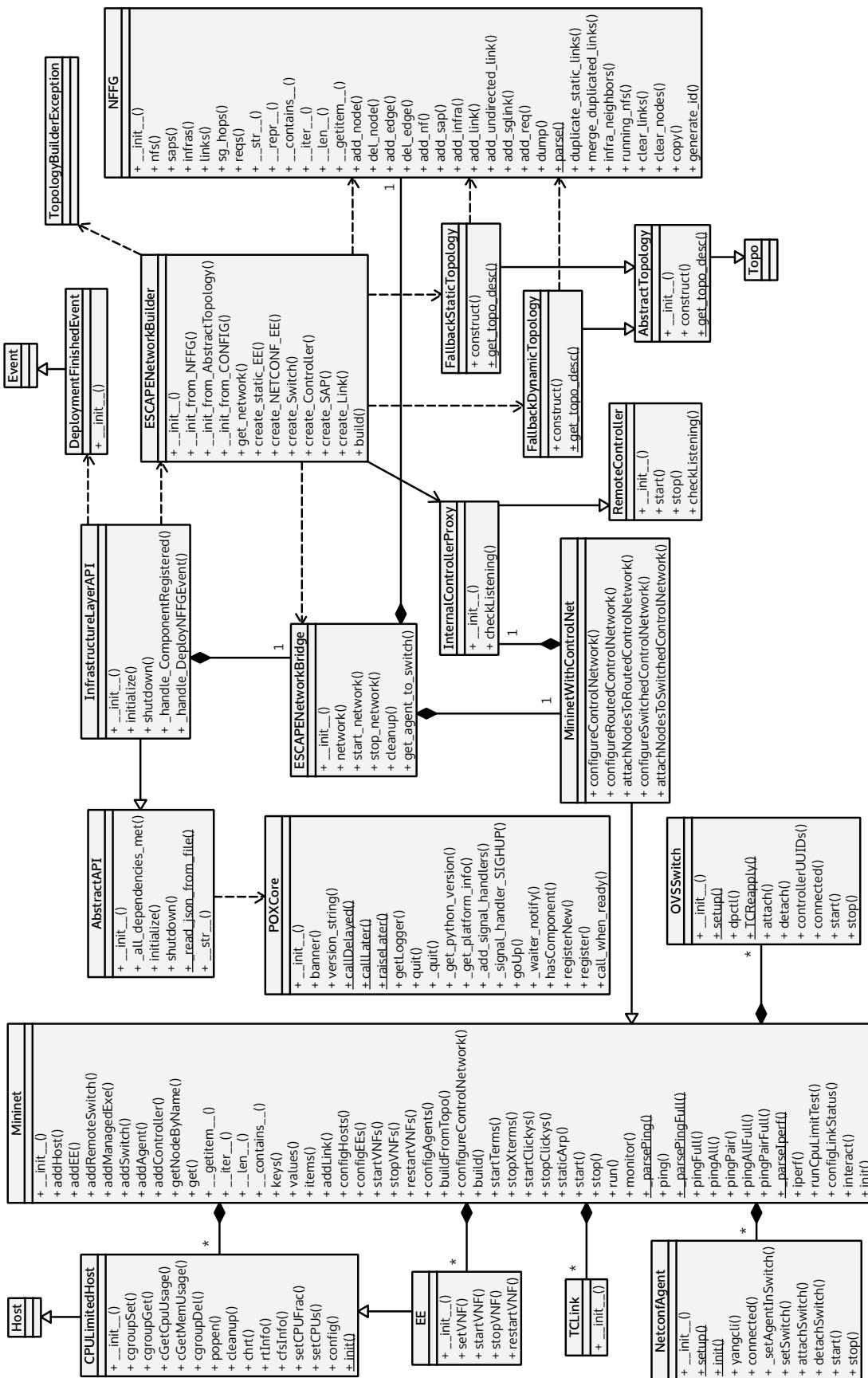


Diagram: II class diagram Page 1

```

module: nffg
  +-rw nffg
    +-rw parameters
      | +-rw id      string
      | +-rw name?   string
      | +-rw version  string
    +-rw node_nfs* [id]
      | +-rw id      string
      | +-rw name?   string
      | +-rw functional_type  string
      | +-rw specification
      | | +-rw deployment_type? string
      | +-rw resources
        | | +-rw cpu      string
        | | +-rw mem      string
        | | +-rw storage   string
        | | +-rw delay     string
        | | +-rw bandwidth  string
      | +-rw ports* [id]
        | | +-rw id      string
        | | +-rw property* string
    +-rw node_saps* [id]
      | +-rw id      string
      | +-rw name?   string
      | +-rw domain?  string
    +-rw ports* [id]
      | +-rw id      string
      | +-rw property* string
    +-rw node_infras* [id]
      | +-rw id      string
      | +-rw name?   string
      | +-rw domain?  string
      | +-rw type     string
      | +-rw supported* [functional_type]
      | | +-rw functional_type string
    +-rw resources
      | | +-rw cpu      string
      | | +-rw mem      string
      | | +-rw storage   string
      | | +-rw delay     string
      | | +-rw bandwidth  string
      | | +-rw flowrules* [id]
        | | | +-rw id      string
        | | | +-rw match    string
        | | | +-rw action   string
        | | | +-rw bandwidth? string
      | | +-rw edge_links* [id]
        | | | +-rw id      string
        | | | +-rw src_node string
        | | | +-rw src_port string
        | | | +-rw dst_node string
        | | | +-rw dst_port string
        | | | +-rw backward? string
        | | | +-rw reqs
        | | | | +-rw delay?   string
        | | | | +-rw bandwidth? string
      | | +-rw edge_sg_nexthops* [id]
        | | | +-rw id      string
        | | | +-rw src_node string
        | | | +-rw src_port string
        | | | +-rw dst_node string
        | | | +-rw dst_port string
        | | | +-rw flowclass? string
        | | | +-rw edge_reqs* [id]
          | | | | +-rw id      string
          | | | | +-rw src_node string
          | | | | +-rw src_port string
          | | | | +-rw dst_node string
          | | | | +-rw dst_port string
          | | | | +-rw reqs
          | | | | | +-rw delay?   string
          | | | | | +-rw bandwidth? string
          | | | | +-rw sg_path* [edge_sg_nexthop_id]
            | | | | | +-rw edge_sg_nexthop_id string

```

Figure 17: YANG data model of our internal NF-FG

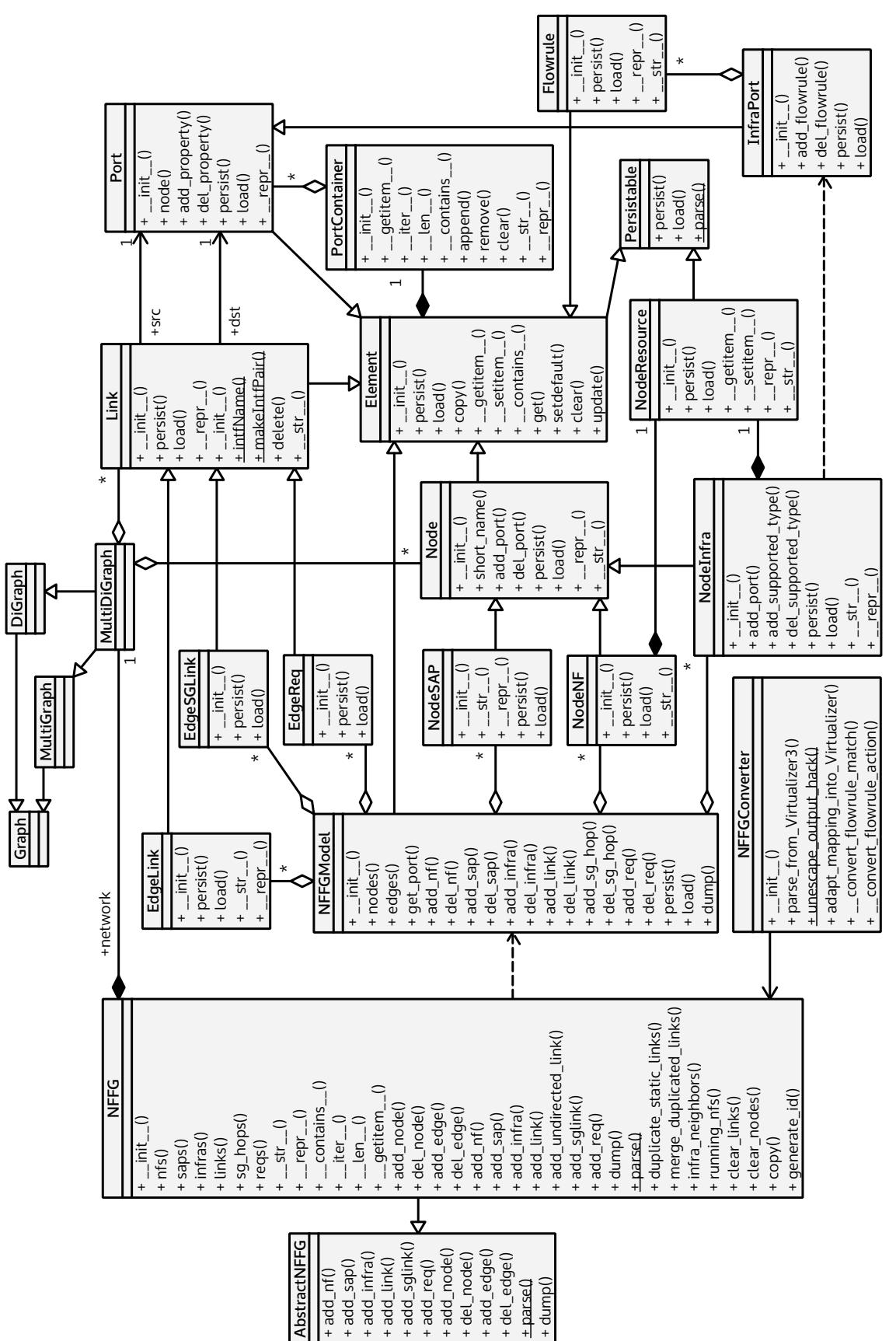


Figure 18: Class diagram of the NF-FG module

---

## References

- [D2.2] Robert Szabo, Balazs Sonkoly, Mario Kind et al. Deliverable 2.2: Final Architecture. Tech. rep. UNIFY Project, 2014.
- [D3.2] Pontus Skoldstrom et al. D3.2 Detailed functional specification and algorithm description. Tech. rep. D3.2. UNIFY Project, Apr. 2015.
- [D3.2a] Pontus Skoldstrom et al. D3.2a Network Function Forwarding Graph specification (Supplement to D3.2). Tech. rep. D3.2sup. UNIFY Project, July 2015.
- [D3.3] Jokin Garay et al. D3.3 Revised framework with functions and semantics. Tech. rep. D3.3. UNIFY Project, Oct. 2015.
- [Juju] Ubuntu Community. Juju. 2015. URL: <https://github.com/juju/juju>.
- [Nem+15] Balazs Nemeth, Janos Czentye, Gabor Vaszkun, Levente Csikor, and Balazs Sonkoly. "Customizable real-time service graph mapping algorithm in carrier grade networks". In: Proceedings of the 2015 IEEE-NFV-SDN. IEEE. 2015.
- [POX] Murphy McCauley. POX. 2015. URL: <https://github.com/noxrepo/pox>.
- [Son+15] Balázs Sonkoly, János Czentye, Robert Szabo, Dávid Jocha, János Elek, Sahel Saghaf, Wouter Tavernier, and Fulvio Risso. "Multi-domain service orchestration over networks and clouds: a unified approach". In: Proceedings of the 2015 ACM conference on SIGCOMM. ACM. 2015.



## D3.4 Annex I: ESCAPEv2 Documentation

Release 2.0.0

János Czentye (BME), Balázs Sonkoly (BME)

This project is co-funded  
by the European Union



---

## Overview

---

On the one hand, ESCAPE (Extensible Service ChAin Prototyping Environment) is a general prototyping framework which supports the development of several parts of the service chaining architecture including VNF implementation, traffic steering, virtual network embedding, etc. On the other hand, ESCAPE is a proof of concept prototype implementing a novel SFC (Service Function Chaining) architecture proposed by EU FP7 UNIFY project (<https://www.fp7-unify.eu/>). It is a realization of the UNIFY service programming and orchestration framework which enables the joint programming and virtualization of cloud and networking resources.

**Tip:** For more information on the concept, motivation and demo use-cases, we suggest the following papers.

**UNIFY Architecture:**

- Balázs Sonkoly, Robert Szabo, Dávid Jocha, János Czentye, Mario Kind, and Fritz-Joachim Westphal, *UNIFYing Cloud and Carrier Network Resources: An Architectural View*, in Proc. IEEE Global Telecommunications Conference (GLOBECOM), 2015.

**ESCAPE as a multi-domain orchestrator:**

- Balázs Sonkoly, János Czentye, Robert Szabo, Dávid Jocha, János Elek, Sahel Sahhaf, Wouter Tavernier, Fulvio Risso, *Multi-domain service orchestration over networks and clouds: a unified approach*, In Proceedings of ACM SIGCOMM (Demo), August 17-21, 2015, London, United Kingdom. [Download the paper](http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p377.pdf) (<http://conferences.sigcomm.org/sigcomm/2015/pdf/papers/p377.pdf>)
- [Demo video](https://www.youtube.com/watch?v=T3Fna5v-hFw) (<https://www.youtube.com/watch?v=T3Fna5v-hFw>)
- [Demo video as a presentation with manual control](http://prezi.com/f-ms1rwxxdwa/?utm_campaign=share&utm_medium=copy&rc=ex0share) ([http://prezi.com/f-ms1rwxxdwa/?utm\\_campaign=share&utm\\_medium=copy&rc=ex0share](http://prezi.com/f-ms1rwxxdwa/?utm_campaign=share&utm_medium=copy&rc=ex0share))

**Previous version of ESCAPE:**

- Attila Csoma, Balázs Sonkoly, Levente Csikor, Felicián Németh, András Gulyás, Wouter Tavernier, Sahel Sahhaf, *ESCAPE: Extensible Service ChAin Prototyping Environment using Mininet, Click, NETCONF and POX*, In Proceedings of ACM SIGCOMM (Demo), August 17-22, 2014, Chicago, IL, USA. [Download the paper](http://dl.acm.org/authorize?N71297) (<http://dl.acm.org/authorize?N71297>)
- The source code of the previous version of ESCAPE is available at our [github page](https://github.com/nemethf/escape) (<https://github.com/nemethf/escape>).

---

For further information contact [balazs.sonkoly@tmit.bme.hu](mailto:balazs.sonkoly@tmit.bme.hu) ([balazs.sonkoly@tmit.bme.hu](mailto:balazs.sonkoly@tmit.bme.hu))

---

## Installation

---

Because ESCAPEv2 relies on POX and written in Python there is no need for explicit compiling or installation. The only requirement need to be pre-installed is a Python interpreter.

The recommended Python version, in which the development and mostly the testing are performed, is the standard CPython **2.7.9** but the 2.7.6 (pre-build Mininet VM) and 2.7.10 versions are also tested and supported.

**Warning:** Only the standard CPython interpreter is supported!

If you want to use a different and separated Python version check the Virtual Environment section below.

### 2.1 The preferred way

1. Download one of pre-build Mininet image which has already had the necessary tools (Mininet scripts and Open vSwitch).

```
https://github.com/mininet/mininet/wiki/Mininet-VM-Images
```

The images are in an open virtual format (.ovf) which can be imported by most of the virtualization managers.

Username/password: **mininet/mininet**

Our implementation relies on Mininet 2.1.0, but ESCAPEv2 has been tested on the newest image too (Mininet 2.2.1 on Ubuntu 14.04 - 64 bit) and no problem has occurred yet!

2. Create the .ssh folder in the home directory and copy your private RSA key which you gave on the *fp7-unify.eu GitLab* site into the VM with the name id\_rsa. If you use the Mininet image then the following commands can be used in the VM to copy your RSA key from your host:

```
$ cd
$ mkdir .ssh
$ scp <your_user>@<host_ip>:~/ssh/<your_ssh_key> ~/.ssh/id_rsa
```

3. Clone the shared escape repository in a folder named: *escape*.

```
$ git clone git@gitlab.fp7-unify.eu:Balazs.Sonkoly/escape-shared.git escape
```

4. Install the necessary dependencies with the `install_dep.sh` script (system and Python packages, OpenYuma with VNFStarter module):

```
$ cd escape
$ ./install_dep.sh
```

**In a high level the script above does the following things:**

- Install the necessary system and Python packages

- Compile and install the [OpenYuma](https://github.com/OpenClovis/OpenYuma) (<https://github.com/OpenClovis/OpenYuma>) tools with our *VNF\_starter* module
- Compile and install [Click](http://read.cs.ucla.edu/click/click) (<http://read.cs.ucla.edu/click/click>) modular router and The Click GUI: [Clicky](http://read.cs.ucla.edu/click/clicky) (<http://read.cs.ucla.edu/click/clicky>)
- Install [neo4j](http://neo4j.com/) (<http://neo4j.com/>) graph database for NFIB

5. Run ESCAPEv2 with one of the commands listed in a later section. To see the available arguments of the top starting script check the help menu:

```
$ ./escape.py --help
```

## 2.2 The hard way

Obviously you can install ESCAPEv2 on your host or on an empty VM too. For that you need to install the requirements manually.

To install the Python dependencies and other system packages you can use the dependency installation script mentioned above or you can do it manually.

### Dependencies

If you don't want to install the Python dependencies globally you can follow the hard way and setup a virtual environment. Otherwise just run the following command(s):

Required system and Python packages:

```
$ sudo apt-get -y install libxml2-dev libxsolt1-dev zlib1g-dev libsqlite3-dev \
    python-pip python-libxml2 python-libxsolt1 python-lxml python-paramiko \
    python-dev python-networkx libxml2-dev libssh2-1-dev libgcrypt11-dev \
    libncurses5-dev libglib2.0-dev libgtk2.0-dev gcc make automake openssh-client \
    openssh-server ssh libssl-dev

$ sudo pip install requests jinja2 ncclient lxml networkx py2neo networkx_viewer \
    numpy
```

For doc generations:

```
$ sudo pip install sphinx
```

For domain emulation scripts:

```
$ sudo pip install tornado
```

Other required programs (OpenYuma, click, neo4j, etc.), which are installed by the *install\_dep.sh* script by default, are also need to be installed manually.

In extreme cases, e.g. the *install\_dep.sh* ran into an error, you should install these dependencies one by one according to your OS, distro or development environment. For that you can check the steps in the install script and/or the online documentations referenced in entry 4. of the previous subsection.

To use the Infrastructure Layer of ESCAPEv2, Mininet must be installed on the host (more precisely the **Open vSwitch** implementation and the specific **mnexec** utility script is also need to be installed globally).

If one version of Mininet has already been installed, there should be nothing to do. ESCAPEv2 uses the specifically-modified Mininet files in the project folder (*Mininet v2.1.0mod-ESCAPE*) which use the globally installed Mininet utility scripts (mnexec).

Otherwise these assets have to be install manually which could be done from our Mininet folder (escape/mininet) or from the official Mininet git repository (<https://github.com/mininet/mininet/>). Mininet has an install script for the installations (see the help with the *-h* flag):

```
$ sudo mininet/util/install.sh -en
```

But the script occasionally **NOT** works correctly, especially on newer distributions because the `sudo apt-get install openvswitch-switch` command will not install the newest version of OVS properly due some major changes in OVS architecture!

Run the following command to check the installation was correct:

```
$ sudo mn --test pingall
```

However you can install the Open vSwitch packages manually:

```
$ sudo apt-get install openvswitch-common openvswitch-switch \
  openvswitch-testcontroller
```

If the command complains about the Open vSwitch not installed then you have to install it from source. See more on <http://openvswitch.org/download/>. On the newest distributions (e.g. Ubuntu 15.04) more steps and explicit patching is required. For that the only way is sadly to use google and search for it based on your distro. But a good choice to start here: <https://github.com/mininet/mininet/wiki/Installing-new-version-of-Open-vSwitch>

---

**Hint:** If your intention is to run ESCAPEv2 in a virtual machine, you should really consider to use one of the pre-build Mininet VM images.

---

If you want to develop on your host machine, you should take care of a user for the netconfd server. This user's name and password will be used for the connection establishment between ESCAPEv2 and the Execution Environments (EE).

---

**Note:** These parameters can be changed conveniently in the global config under the config entry of *VNFStarter Adapter*.

---

An another solution is to define a system user for the netconfd. To create a user (advisable to use *mininet* as in the Mininet-based VM) use the following commands:

```
$ sudo adduser --system --shell /bin/bash --no-create-home mininet
$ sudo addgroup mininet sudo
```

For security reasons it's highly recommended to limit the SSH connections for the *mininet* user only to localhost.

```
$ sudo echo "AllowUsers    <your_user1 user2 ...>  mininet@localhost" >> \
  /etc/ssh/sshd_config
$ sudo service ssh reload
```

Check the created user with the following command:

```
$ ssh mininet@localhost
```

## 2.3 Setup a Virtual environment (optional)

ESCAPEv2 also supports Python-based virtual environment in order to setup a different Python version or even a different interpreter (not recommended) for ESCAPEv2 or to separate dependent packages from system-wide Python.

To setup a virtual environment based on `virtualenv` (<https://virtualenv.readthedocs.org/en/latest/>) Python package with a standalone CPython 2.7.10 interpreter run the following script:

```
$ ./set_virtualenv.sh
```

**This script does the following steps:**

- Install additional dependencies

- Download, compile and install the 2.7.10 (currently the newest) Python interpreter in a separated directory
- Setup a virtual environment in the main project directory independently from the system-wide Python packages
- Install the Python dependencies in this environment
- and finally create a `.use_virtualenv`" file to enable the newly created virtual environment for the topmost `escape.py` starting script.

Usage:

```
$ ./set_virtualenv.sh -h
Usage: ./set_virtualenv.sh [-p python_version] [-h]

Install script for ESCAPEv2 to setup virtual environment

optional parameters:

-p    set Python version (default: 2.7.10)
-h    show this help message and exit

Example: ./set_virtualenv.sh -p 2.7.9

Based on virtualenv. More information: virtualenv -h
```

The `escape.py` script can detect the `.use_virtualenv` file automatically and activates the virtual environment transparently. If you want to disable the virtual environment then just delete the `.use_virtualenv` file.

The `virtualenv` can also be enabled by the `--environment` flag of the topmost `escape.py` script.

In order to setup the environment manually, define other Python version/interpreter, enable system-wide Python / pip packages

```
$ virtualenv -p=<python_dir> --no-site-packages/system-site-packages <...> escape
```

or activate/deactivate the environment manually

```
$ cd escape
$ source bin/activate # activate virtual environment
$ deactivate # deactivate
```

check the content of the setup script or see the **Virtualenv User Guide** (<https://virtualenv.readthedocs.org/en/latest/userguide.html>).

---

## ESCAPEv2 example commands

---

ESCAPEv2 can be started with the topmost `escape.py` script in the project's root directory or can be started calling the `pox.py` script directly with the layer modules and necessary arguments under the `pox` directory.

### 3.1 The simplest use-case

Run ESCAPEv2 with the Mininet-based Infrastructure layer and debug logging mode:

```
$ ./escape.py -df
```

Usage:

```
$ ./escape.py -h
usage: escape.py [-h] [-v] [-a] [-c path] [-d] [-e] [-f] [-i] [-r] [-s file]
                 [-t file] [-x] [-4]

...
ESCAPEv2: Extensible Service ChAin Prototyping Environment using Mininet,
Click, NETCONF and POX

optional arguments:
  -h, --help            show this help message and exit
  -v, --version         show program's version number and exit

ESCAPEv2 arguments:
  -a, --agent           run in agent mode: start the ROS REST-API (without the
                        Service sublayer (SAS))
  -c path, --config path
                        override default config filename
  -d, --debug           run the ESCAPE in debug mode
  -e, --environment    run ESCAPEv2 in the pre-defined virtualenv environment
  -f, --full            run the infrastructure layer also
  -i, --interactive    run an interactive shell for observing internal states
  -r, --rosapi          start the REST-API for the Resource Orchestration
                        sublayer (ROS)
  -s file, --service file
                        skip the SAS REST-API initiation and read the service
                        request from the given file
  -t file, --topo file  read the topology from the given file explicitly
  -x, --clean           run the cleanup task standalone and kill remained
                        programs, interfaces, veth parts and junk files
  -4, --cfor             start the REST-API for the Cf-Or interface
                        optional POX modules
  ...
```

During a test or development the `--debug` flag is almost necessary.

If you want to run a test topology, use the `--full` flag to initiate the Infrastructure layer also. ESCAPEv2 will parse the topology description form file (`escape-mn-topo.nffg` by default) and start the Infrastructure layer with the Mininet-based emulation.

If the request is in a file it's more convenient to give it with the `--service` initial parameter and not bother with the REST-API.

An additional configuration file can be given with the `--config` flag. The configuration file is loaded during initialization and ESCAPEv2 only updates the default configuration instead of replaces it in order to minimize the sizes of the additional parameters.

The most common changes in the configurations is the file path of the initial topology which is used by the Infrastructure layer to initiate the Mininet-emulated network. To simplify this case the topology file can be given with the `--topo` parameter explicitly.

If an error is occurred or need to observe the internal states you can start ESCAPEv2 with an interactive Python shell using the `--interactive` flag.

With the `--environment` flag ESCAPEv2 can be started in a pre-defined virtualenv environment whether the virtualenv is permanently enabled with the `.use_virtualenv` file or not.

The main layers which grouping the entities are reachable through the main POX object called `core` with the names:

- `service` - Service layer
- `orchestration` - Resource Orchestration Sublayer
- `adaptation` - Controller Adaptation Sublayer
- `infrastructure` - Infrastructure layer

---

**Hint:** In the interactive shell the tab-auto completion is working in most cases.

---

So a possible scenario for testing ESCAPEv2 with a test request given in a file and check the state of the DoV:

```
$ ./escape.py -dfi -s pox/escape-mn-req.nffg
Starting ESCAPEv2...
Command: sudo /home/czentye/escape/pox/pox.py unify --full \
--sg_file=/home/czentye/escape/pox/escape-mn-req.nffg py --completion

...
ESCAPE> print core.adaptation.controller_adapter.domainResManager._dov
      .get_resource_info().dump()
{
  "parameters": {
    "id": "DoV",
    "name": "dov-140454330075984",
    "version": "1.0"
  },
  "node_saps": [
    {
      "id": "SAP1",
      ...
    }
  ]
}
```

## 3.2 More advanced commands (mostly advisable for testing purposes)

For more flexible control ESCAPEv2 can be started directly with POX's starting script under the `pox` folder.

---

**Note:** The topmost `escape.py` script uses this `pox.py` script to start ESCAPEv2. In debug mode the assem-

bled POX command is printed also.

Basic command:

```
$ ./pox.py unify
```

One of a basic commands for debugging:

```
$ ./pox.py --verbose unify py
```

For forcing to log on DEBUG level the --verbose flag of the pox.py script can be used. Or the *log.level* POX module can be used which would be the preferred way. E.g.:

```
$ ./pox.py --verbose <modules>
$ ./pox.py log.level --DEBUG <modules>
```

Basic command to initiate a built-in emulated network for testing:

```
# Infrastructure layer requires root privileges due to use of Mininet!
$ sudo ./pox.py unify --full
```

Minimal command with explicitly-defined components (components' order is irrelevant):

```
$ ./pox.py service orchestration adaptation
```

Without service layer:

```
$ ./pox.py orchestration adaptation
```

With infrastructure layer:

```
$ sudo ./pox.py service orchestration adaptation --with_infr infrastructure
```

Long version with debugging and explicitly-defined components (analogous with ./pox.py unify --full):

```
$ sudo ./pox.py --verbose log.level --DEBUG samples.pretty_log service \
orchestration adaptation --with_infr infrastructure
```

Start layers with graph-represented input contained in a specific file:

```
$ ./pox.py service --sg_file=<path> ...
$ ./pox.py unify --sg_file=<path>

$ ./pox.py orchestration --nffg_file=<path> ...
$ ./pox.py adaptation --mapped_nffg=<path> ...
```

Start ESCAPEv2 with built-in GUI:

```
$ ./pox.py service --gui ...
$ ./pox.py unify --gui
```

Start layer in standalone mode (no dependency check and handling) for test/debug:

```
$ ./pox.py service --standalone
$ ./pox.py orchestration --standalone
$ ./pox.py adaptation --standalone
$ sudo ./pox.py infrastructure --standalone

$ ./pox.py service orchestration --standalone
```

---

## REST APIs

---

ESCAPEv2 has currently 3 REST-APIs.

The Service layer has a REST-API for communication with users/GUI. This API is initiated by default when the layer was started.

The Resource Orchestration layer has 2 API which are only initiated if the appropriate flag is given to the starting script. The ROS API can be used for communicating with other UNIFY layer e.g. a Controller Adaptation Sublayer of a standalone ESCAPEv2 in a multi-level scenario or with a GUI. The CfOr API realizes the interface for service elasticity feature.

### 4.1 Common API functions

*Operations:* Every API has the following 3 function (defined in *AbstractRequestHandler*):

Path	Params	HTTP verbs	Description
/version	None	GET	Returns with the current version of ESCAPEv2
/ping	None	ALL	Returns with the “OK” string
/operations	None	GET	Returns with the implemented operations

### 4.2 Service API specific functions

The SAS API is automatically initiated by the Service layer. If the `--service` flag is used the service request is loaded from the given file and the REST-API initiation is skipped.

*Content Negotiation:* The Service layer’s RESTful API accepts and returns data only in JSON format.

Path	Params	HTTP verbs	Description
/topology	None	GET	Returns with the resource view of the Service layer
/sg	NFFG	ALL	Initiate given NFFG. Returns the initiation is accepted or not

### 4.3 ROS API specific functions

Can be started with the `--agent` or `--rosapi` initial flags.

Path	Params	HTTP verbs	Description
/get-config	None	GET	Returns with the resource view of the ROS
/edit-config	NFFG	ALL	Initiate given NFFG.

## 4.4 Cf-Or API specific functions

Can be started with the `--cfor` flag.

Path	Params	HTTP verbs	Description
<code>/get-config</code>	None	GET	Returns with the resource view from the assigned Virtualizer
<code>/edit-config</code>	NFFG	ALL	Initiate given NFFG.

---

## Configuration

---

ESCAPEv2 has a default configuration under the *escape* package (in the `__init__.py` file as `cfg`). This configuration contains the necessary information for manager/adapter initializations, remote connections, etc. and also provides the base for the internal running configuration.

If you want to override some of the parameters you can change the default values in the `cfg` directly (not preferred) or you can just define them in an additional config file.

The default configuration file which ESCAPEv2 is looking for is `escape.config`. At every start ESCAPEv2 checks the presence of this file and updates/overrides the running configuration if it's necessary.

The `escape.py` starting script also provides the opportunity to specify a different configuration file with the `--config` initial argument.

The additional config can be added only in JSON format, but the structure of the configuration is strictly follows the default configuration which is defined in Python with basic data structures.

The configuration units (coherent values, single boolean flags, paths, etc.) are handled through the main `ESCAPEConfig` class so every possible configuration entry has an assigned *getter* function in the main class.

---

**Important:** The configurations is parsed during the starting process. Changes in the config file have no effect at runtime.

---

## 5.1 Configuration structure

The configurations is divided to 4 parts according to the UNIFY's / ESCAPEv2's main layers, namely `service`, `orchestration`, `adaptation` and `infrastructure`.

### 5.1.1 service and orchestration

The top 2 layer (`service` and `orchestration`) has similar configuration parameters. In both layers the mapping process can be controlled with the following entries:

- **MAPPER** defines the mapping class which controls the mapping process (inherited from `AbstractMapper`)
- **STRATEGY** defines the mapping strategy class which calls the actual mapping algorithm (inherited from `AbstractMappingStrategy`)
- **PROCESSOR** defines the Processor class which contains the pre/post mapping functions for validation and other auxiliary functions (inherited from `AbstractMappingDataProcessor`)

The values of class configurations (such the entries above) always contains the **module** and **class** names of the actual class. With this approach ESCAPEv2 can also instantiate and use different implementations from external Python packages. The only requirement for these classes is to be included in the scope of ESCAPEv2 (more precisely in the PYTHONPATH of the Python interpreter which runs ESCAPEv2).

---

**Note:** Every additional subdirectory in the project's root is always added to the search path (scope) dynamically at initial time by the main `escape` module.

---

The mapping process and pre/post processing can be enabled/disabled with the `mapping-enabled` (boolean) and `enabled` (boolean) values under the appropriate entries.

The mapping algorithm called in the Strategy class can be initiated in a worker thread with the `THREADED` flag, but this feature is still in experimental phase.

These 2 layers can initiate REST-APIs also. The initial parameters are defined under the names of the APIs:

- **REST-API** - top REST-API in the SAS layer
- **SI-Or** - SI-Or interface in the ROS layer for external components i.e. for upper UNIFY entities, GUI or other ESCAPEv2 instance in a distributed, multi-layered scenario
- **Cf-Or** - Cf-Or interface in the ROS layer for supporting service elasticity feature

These REST-API configurations consist of

- a specific handler class which initiated for every request and handles the requests (inherited from `AbstractRequestHandler`) defined with the module and class pair
- address of the REST-API defined with the address and port (integer) pair
- prefix of the API which appears in the URL right before the REST functions
- optionally the type of used Virtualizer (`virtualizer_type`) which filters the data flow of the API (currently only supported the global (`GLOBAL`) and single BiS-BiS (`SINGLE`) Virtualizer)

### 5.1.2 adaptation

The adaptation layer contains the different Manager (inherited from `AbstractDomainManager`) and Adapter (inherited from `AbstractESCAPEAdapter`) classes under their specific name which is defined in the name class attribute. These configurations are used by the `ComponentConfigurator` to initiate the required component dynamically. The class configurations can be given by the module and class pair similar way as so far. Other values such as path, url, keepalive, etc. will be forwarded to the constructor of the component at initialization time so the possible config names and types result from the constructor attributes.

The `MANAGERS` config value contains the Managers need to be initiated.

---

**Hint:** In order to activate a manager and manage the specific domain add the config name of the DomainManager to the `MANAGERS` list. The manager will be initiated with other Managers at boot time of ESCAPEv2.

---

With the `RESET-DOMAINS-AFTER-SHUTDOWN` config entry can be enabled/disabled the cleanup of the domains.

### 5.1.3 infrastructure

The configuration of `infrastructure` layer controls the Mininet-based emulation.

The `TOPO` path value defines the file which will be parsed and processed to build the Mininet structure.

The `FALLBACK-TOPO` defines an inner class which can initiate a topology if the topology file is not found.

The `NETWORK-OPTS` is an optional data which can be added to override the default constructor parameters of the Mininet class.

The `Controller`, `EE`, `Switch`, `SAP` and `Link` dictionaries can contain optional parameters for the constructors of the internal Mininet-based representation. In most cases these parameters need to be left unchanged.

Other simple values can be added too to refine the control of the emulation such as enable/disable the xterm initiation for SAPs (`SAP-xterm`) or the cleanup task (`SHUTDOWN-CLEAN`).

## 5.2 Default configuration

The following snippet represents the default configuration of ESCAPEv2 in JSON format. An additional configuration file should be based on a subpart of this configurations structure.

```
{
  "service": {
    "MAPPER": {
      "module": "escape.service.sas_mapping",
      "class": "ServiceGraphMapper",
      "mapping-enabled": false
    },
    "STRATEGY": {
      "module": "escape.service.sas_mapping",
      "class": "DefaultServiceMappingStrategy",
      "THREADED": false
    },
    "PROCESSOR": {
      "module": "escape.util.mapping",
      "class": "ProcessorSkipper",
      "enabled": false
    },
    "REST-API": {
      "module": "escape.service.sas_API",
      "class": "ServiceRequestHandler",
      "prefix": "escape",
      "address": "0.0.0.0",
      "port": 8008
    }
  },
  "orchestration": {
    "MAPPER": {
      "module": "escape.orchest.ros_mapping",
      "class": "ResourceOrchestrationMapper",
      "mapping-enabled": true
    },
    "STRATEGY": {
      "module": "escape.orchest.ros_mapping",
      "class": "ESCAPEMappingStrategy",
      "THREADED": false
    },
    "PROCESSOR": {
      "module": "escape.util.mapping",
      "class": "ProcessorSkipper",
      "enabled": true
    },
    "S1-Or": {
      "module": "escape.orchest.ros_API",
      "class": "ROSAgentRequestHandler",
      "prefix": "escape",
      "address": "0.0.0.0",
      "port": 8888,
      "virtualizer_type": "GLOBAL"
    },
    "Cf-Or": {
      "module": "escape.orchest.ros_API",
      "class": "CfOrRequestHandler",
      "prefix": "cfor",
      "address": "0.0.0.0",
      "port": 8889,
      "virtualizer_type": "GLOBAL"
    }
  }
}
```

```

},
"adaptation": {
  "MANAGERS": [],
  "INTERNAL-POX": {
    "module": "escape.adapt.adapters",
    "class": "InternalPOXAdapter",
    "name": null,
    "address": "127.0.0.1",
    "port": 6653,
    "keepalive": false
  },
  "SDN-POX": {
    "module": "escape.adapt.adapters",
    "class": "SDNDomainPOXAdapter",
    "name": null,
    "address": "0.0.0.0",
    "port": 6633,
    "keepalive": false
  },
  "MININET": {
    "module": "escape.adapt.adapters",
    "class": "InternalMininetAdapter",
    "net": null
  },
  "SDN-TOPO": {
    "module": "escape.adapt.adapters",
    "class": "SDNDomainTopoAdapter",
    "path": "examples/sdn-topo.nffg"
  },
  "VNFStarter": {
    "module": "escape.adapt.adapters",
    "class": "VNFStarterAdapter",
    "username": "mininet",
    "password": "mininet",
    "server": "127.0.0.1",
    "port": 830,
    "timeout": null
  },
  "ESCAPE-REST": {
    "module": "escape.adapt.adapters",
    "class": "RemoteESCAPEv2RESTAdapter",
    "url": "http://localhost:8083"
  },
  "OpenStack-REST": {
    "module": "escape.adapt.adapters",
    "class": "OpenStackRESTAdapter",
    "url": "http://localhost:8081"
  },
  "UN-REST": {
    "module": "escape.adapt.adapters",
    "class": "UniversalNodeRESTAdapter",
    "url": "http://localhost:8082"
  },
  "INTERNAL": {
    "module": "escape.adapt.managers",
    "class": "InternalDomainManager",
    "poll": false
  },
  "REMOTE-ESCAPE": {
    "module": "escape.adapt.managers",
    "class": "RemoteESCAPEDomainManager",
    "poll": false
  }
},

```

```
"OPENSTACK": {
    "module": "escape.adapt.managers",
    "class": "OpenStackDomainManager",
    "poll": false
},
"UN": {
    "module": "escape.adapt.managers",
    "class": "UniversalNodeDomainManager",
    "poll": false
},
"DOCKER": {
    "module": "escape.adapt.managers",
    "class": "DockerDomainManager",
    "poll": false
},
"SDN": {
    "module": "escape.adapt.managers",
    "class": "SDNDomainManager",
    "poll": false
},
"RESET-DOMAINS-AFTER-SHUTDOWN": true
},
"infrastructure": {
    "TOPO": "examples/escape-mn-topo.nffg",
    "NETWORK-OPTS": null,
    "Controller": {
        "ip": "127.0.0.1",
        "port": 6653
    },
    "EE": null,
    "Switch": null,
    "SAP": null,
    "Link": null,
    "FALLBACK-TOPO": {
        "module": "escape.infr.topology",
        "class": "FallbackDynamicTopology"
    },
    "SAP-xterms": true,
    "SHUTDOWN-CLEAN": true
},
"additional-config-file": "escape.config"
}
```

# Development

---

Suggested IDE: [Pycharm Community Edition](https://www.jetbrains.com/pycharm/) (<https://www.jetbrains.com/pycharm/>)

Coding conventions:

- **Sizes:**
  - Tab size: 2
  - Indent: 2
  - Continuation indent: 5
  - Right margin (columns): 80
- Use spaces instead of tab characters
- Use one space before method declaration parentheses
- Use spaces around operators
- Not use spaces in named parameters and keywords argument
- Use double blank lines around classes and top-level functions

---

## Debugging

---

You can use PyCharm for debugging. In this case you have to specify a new Python interpreter using the `python_root_debugger.sh` script to be able to run ESCAPE with root privileges.

You can use POX's `py` stock component also which open an interactive Python shell. With that you can observe the internal state of the running ESCAPE instance, experiment or even call different functions.

POX uses a topmost object called `core` which serves a rendezvous point between POX's components (e.g. our components representing the UNIFY layers). Through that object we can reach every registered object easily. E.g. to shut down the REST API of the Service layer manually we can use the following function call:

```
$ Ready.  
$ ESCAPE>  
$ ESCAPE> core.service.rest_api.stop()
```

One instance of the `ESCAPEInteractiveHelper` is registered by default under the name: `helper`. An example to dump the running configuration of ESCAPEv2:

```
$ ESCAPE> core.helper.config()  
{  
    "infrastructure": {  
        "NETWORK-OPTS": null,  
        "FALLBACK-TOPO": {  
            "class": "BackupTopology",  
            "module": "escape.infr.topology"  
        }  
    }  
}
```

More help and description about the useful helper functions and the `core` object is in the comments/documentation and on the POX's [wiki](#) (<https://openflow.stanford.edu/display/ONL/POX+Wiki#POXWiki-POXAPIs>) site.

---

## API documentation

---

This documentation contains only the Python class structure and description of the multi-domain multi-level service orchestrator.

Our Mininet-based infrastructure, which is an extended version of Mininet, is not documented here.

## 8.1 ESCAPEv2 class structure

### 8.1.1 `escape` package

Unifying package for ESCAPEv2 functions.

‘cfg’ defines the default configuration settings such as the concrete RequestHandler and strategy classes, the initial Adapter classes, etc.

*CONFIG* contains the ESCAPEv2 dependent configuration as an *ESCAPEConfig*.

`escape.add_dependencies()`

Add dependency directories to PYTHONPATH. Dependencies are directories besides the escape.py initial script except pox.

**Returns** None

#### Submodules

##### `escape.service` package

Subpackage for classes related mostly to Service (Graph) Adaptation sublayer

#### Submodules

**`element_mgmt.py` module** Contains classes relevant to element management.



*AbstractElementManager* is an abstract class for element managers.

`ClickManager` represent the interface to Click elements.

**Module contents** Contains classes relevant to element management.

**class** `escape.service.element_mgmt.AbstractElementManager`  
 Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for element management components (EM).

**Warning:** Not implemented yet!

`__init__()`  
 Init

**class** `escape.service.element_mgmt.ClickManager`  
 Bases: `escape.service.element_mgmt.AbstractElementManager`

Manager class for specific VNF management based on Clicky.

**Warning:** Not implemented yet!

`__init__()`  
 Init.

**sas\_mapping.py module** Contains classes which implement SG mapping functionality.



`DefaultServiceMappingStrategy` implements a default mapping algorithm which map given SG on a single Bis-Bis.

`SGMappingFinishedEvent` can signal end of service graph mapping.

`ServiceGraphMapper` perform the supplementary tasks for SG mapping.

**Module contents** Contains classes which implement SG mapping functionality.

**class** `escape.service.sas_mapping.DefaultServiceMappingStrategy`  
 Bases: `escape.util.mapping.AbstractMappingStrategy`

Mapping class which maps given Service Graph into a single BiS-BiS.

`__init__()`  
 Init.

**classmethod map (graph, resource)**

Default mapping algorithm which maps given Service Graph on one BiS-BiS.

**Parameters**

- **graph** (*NFFG*) – Service Graph
- **resource** (*NFFG*) – virtual resource

**Returns** Network Function Forwarding Graph

**Return type** *NFFG*

**class escape.service.sas\_mapping.SGMappingFinishedEvent (nffg)**

Bases: *pox.lib.revent.revent.Event*

Event for signaling the end of SG mapping.

**\_\_init\_\_ (nffg)**

Init.

**Parameters** **nffg** (*NFFG*) – NF-FG need to be initiated

**class escape.service.sas\_mapping.ServiceGraphMapper (strategy=None)**

Bases: *escape.util.mapping.AbstractMapper*

Helper class for mapping Service Graph to NF-FG.

**\_eventMixin\_events = set([<class ‘escape.service.sas\_mapping.SGMappingFinishedEvent’>])****DEFAULT\_STRATEGY**

alias of *DefaultServiceMappingStrategy*

**\_\_init\_\_ (strategy=None)**

Init Service mapper.

**Returns** None

**\_perform\_mapping (input\_graph, resource\_view)**

Orchestrate mapping of given service graph on given virtual resource.

**Parameters**

- **input\_graph** (*NFFG*) – Service Graph
- **resource\_view** – virtual resource view
- **resource\_view** – *AbstractVirtualizer*

**Returns** Network Function Forwarding Graph

**Return type** *NFFG*

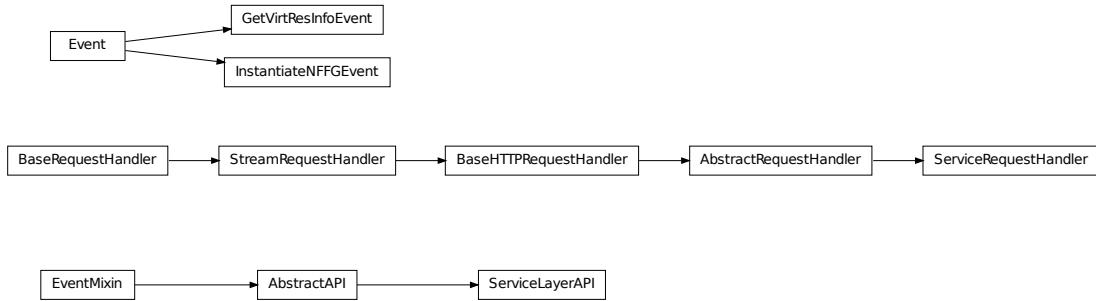
**\_mapping\_finished (nffg)**

Called from a separate thread when the mapping process is finished.

**Parameters** **nffg** (*NFFG*) – generated NF-FG

**Returns** None

**sas\_API.py module** Implements the platform and POX dependent logic for the Service Adaptation Sublayer.



`InstantiateNFFGEvent` can send NF-FG to the lower layer.

`GetVirtResInfoEvent` can request virtual resource info from lower layer.

`ServiceRequestHandler` implement the specific RESTful API functionality thereby realizes the UNIFY's U - SI API.

`ServiceLayerAPI` represents the SAS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Service Adaptation Sublayer.

**class escape.service.sas\_API.InstantiateNFFGEvent (nffg)**

Bases: `pox.lib.revent.revent.Event`

Event for passing NFFG (mapped SG) to Orchestration layer.

**\_\_init\_\_ (nffg)**

Init.

**Parameters** `nffg` (`NFFG`) – NF-FG need to be initiated

**class escape.service.sas\_API.GetVirtResInfoEvent (sid)**

Bases: `pox.lib.revent.revent.Event`

Event for requesting virtual resource info from Orchestration layer.

**\_\_init\_\_ (sid)**

Init.

**Parameters** `sid` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – Service layer ID

**class escape.service.sas\_API.ServiceRequestHandler (request, client\_address, server)**

Bases: `escape.util.api.AbstractRequestHandler`

Request Handler for Service Adaptation SubLayer.

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually: use relevant helper function of core object: `callLater/raiseLater` or use `schedule_as_coop_task` decorator defined in `util.misc` on the called function.

```
request_perm = {'POST': ('ping', 'result', 'sg', 'topology'), 'GET': ('ping', 'version', 'operations', 'topology')}
```

```
bounded_layer = 'service'
```

```
log = <logging.Logger object at 0x4ccca10>
```

```
result()
```

Return the result of a request given by the id.

**sg()**  
Main API function for Service Graph initiation  
Bounded to POST HTTP verb

**topology()**  
Provide internal topology description

```
class escape.service.sas_API.ServiceLayerAPI (standalone=False, **kwargs)
    Bases: escape.util.api.AbstractAPI

    Entry point for Service Adaptation Sublayer.

    Maintain the contact with other UNIFY layers.

    Implement the U - S1 reference point.

    _core_name = 'service'
    LAYER_ID = 'ESCAPE-service'
    dependencies = ('orchestration',)

    __init__(standalone=False, **kwargs)
```

**See also:**

*AbstractAPI.\_\_init\_\_()*

**initialize()**

**See also:**

*AbstractAPI.initialize()*

**shutdown(event)**

**See also:**

*AbstractAPI.shutdown()*

**\_initiate\_rest\_api()**  
Initialize and set up REST API in a different thread.

**Returns** None

**\_initiate\_gui()**  
Initiate and set up GUI.

**\_handle\_SGMappingFinishedEvent(event)**  
Handle SGMappingFinishedEvent and proceed with *NFFG* instantiation.

**Parameters** **event** (*SGMappingFinishedEvent*) – event object

**Returns** None

**api\_sas\_sg\_request(\*args, \*\*kwargs)**  
Initiate service graph in a cooperative micro-task.

**Parameters** **service\_nffg** (*NFFG*) – service graph instance

**Returns** None

**api\_sas\_sg\_request\_delayed(\*args, \*\*kwargs)**  
Initiate service graph in a cooperative micro-task.

**Parameters** **service\_nffg** (*NFFG*) – service graph instance

**Returns** None

**api\_sas\_get\_topology()**

Return with the topology description.

**Returns** topology description requested from the layer's Virtualizer

**Return type** *NFFG*

**get\_result(*id*)**

Return the state of a request given by *id*.

**Parameters** *id* (*str or int*) – request id

**Returns** state

**Return type** *str* (<https://docs.python.org/2.7/library/functions.html#str>)

**\_instantiate\_NFFG(*nffg*)**

Send NFFG to Resource Orchestration Sublayer in an implementation-specific way.

General function which is used from microtask and Python thread also.

**Parameters** *nffg* (*NFFG*) – mapped Service Graph

**Returns** None

**\_handle\_MissingVirtualViewEvent(*event*)**

Request virtual resource info from Orchestration layer (UNIFY SI - Or API).

Invoked when a MissingVirtualViewEvent raised.

Service layer is identified with the sid value automatically.

**Parameters** *event* (*MissingVirtualViewEvent*) – event object

**Returns** None

**\_handle\_VirtResInfoEvent(*event*)**

Save requested virtual resource info as an *AbstractVirtualizer*.

**Parameters** *event* (*VirtResInfoEvent*) – event object

**Returns** None

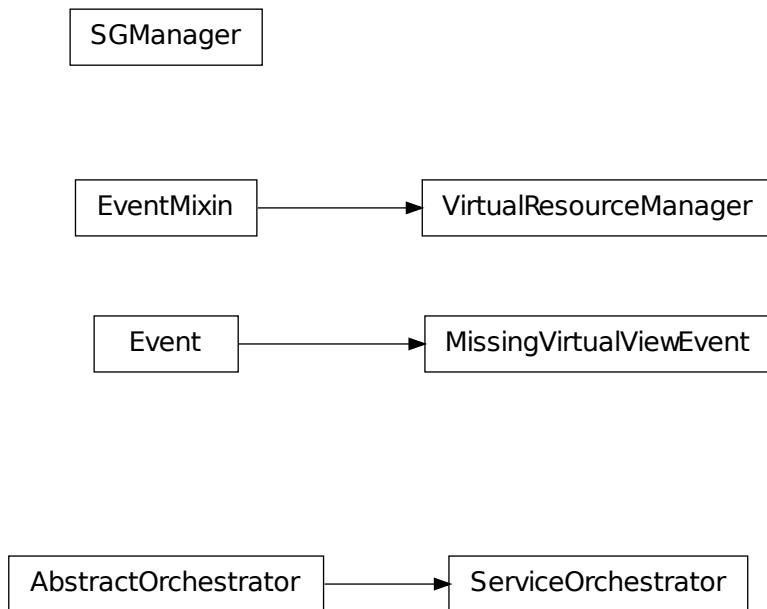
**\_handle\_InstantiationFinishedEvent(*event*)****\_ServiceLayerAPI\_\_proceed\_sg\_request(*service\_nffg*)**

Initiate a Service Graph (UNIFY U-SI API).

**Parameters** *service\_nffg* (*NFFG*) – service graph instance

**Returns** None

**sas\_orchestration.py module** Contains classes relevant to Service Adaptation Sublayer functionality.



*ServiceOrchestrator* orchestrates SG mapping and centralize layer logic.

*SGManager* stores and handles Service Graphs.

*MissingVirtualViewEvent* can signal missing virtual info.

*VirtualResourceManager* contains the functionality tided to the layer's virtual view and virtual resources.

**Module contents** Contains classes relevant to Service Adaptation Sublayer functionality.

**class escape.service.sas\_orchestration.MissingVirtualViewEvent**

Bases: `pox.lib.revent.revent.Event`

Event for signaling missing virtual resource view

**class escape.service.sas\_orchestration.ServiceOrchestrator(layer\_API)**

Bases: `escape.util.mapping.AbstractOrchestrator`

Main class for the actual Service Graph processing.

**DEFAULT\_MAPPER**

alias of `ServiceGraphMapper`

**\_\_init\_\_(layer\_API)**

Initialize main Service Layer components.

**Parameters** `layer_API` (`ServiceLayerAPI`) – layer API instance

**Returns** None

**initiate\_service\_graph(sg)**

Main function for initiating Service Graphs.

**Parameters** `sg` (`NFFG`) – service graph stored in NFFG instance

**Returns** NF-FG description

**Return type** `NFFG`

```
class escape.service.sas_orchestration.SGManager
Bases: object (https://docs.python.org/2.7/library/functions.html#object)
Store, handle and organize Service Graphs.
Currently it just stores SGs in one central place.

__init__()
Init.

save(sg)
Save SG in a dict.

Parameters sg (NFFG) – Service Graph
Returns computed id of given Service Graph
Return type int (https://docs.python.org/2.7/library/functions.html#int)

get(graph_id)
Return service graph with given id.

Parameters graph_id (int (https://docs.python.org/2.7/library/functions.html#int)) –
graph ID
Returns stored Service Graph
Return type NFFG

_generate_id(sg)
Try to generate a unique id for SG.

Parameters sg (NFFG) – SG

class escape.service.sas_orchestration.VirtualResourceManager
Bases: pox.lib.revent.revent.EventMixin
Support Service Graph mapping, follow the used virtual resources according to the Service Graph(s) in effect.

Handles object derived from :class:`AbstractVirtualizer` and requested from lower layer.

_eventMixin_events = set([<class 'escape.service.sas_orchestration.MissingVirtualViewEvent'>])

__init__()
Initialize virtual resource manager.

Returns None

virtual_view
Return resource info of actual layer as an NFFG instance.
If it isn't exist requires it from Orchestration layer.

Returns resource info as a Virtualizer
Return type AbstractVirtualizer
```

## **escape.orchest** package

Subpackage for classes related to UNIFY's Resource Orchestration Sublayer (ROS)

### Submodules

***nfib\_mgmt.py* module** Contains the class for managing NFIB.

## NFIBManager

*NFIBManager* manages the handling of Network Function Information Base.

**Module contents** Contains the class for managing NFIB.

**class escape.orchest.nfib\_mgmt.NFIBManager**  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Manage the handling of Network Function Information Base.

Use neo4j implementation for storing and querying NFs and NF decompositions.

**\_\_init\_\_()**  
Init.

**addNode (node)**  
Add new node to the DB.

**Parameters** `node` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – node to be added to the DB

**Returns** success of addition

**Return type** Boolean

**addClickNF (nf)**  
Add new click-based NF to the DB

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – nf to be added to the DB

**Returns** success of addition

**Return type** Boolean

**addVMNF (nf)**

**static clickCompile (nf)**  
Compile source of the click-based NF

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – the click-based NF

**Returns** success of compilation

**Return type** Boolean

**removeNF (nf\_id)**

Remove an NF and all its decompositions from the DB.

**Parameters** `nf_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) (<https://docs.python.org/2.7/library/string.html#module-string>) – the id of the NF to be removed from the DB

**Returns** success of removal

**Return type** Boolean

**updateNF (nf)**

Update the information of a NF.

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – the information for the NF to be updated

**Returns** success of the update

**Return type** Boolean

#### `getNF(nf_id)`

Get the information for the NF with id equal to nf\_id.

**Parameters** `nf_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the NF to get the information for

**Returns** the information of NF with id equal to nf\_id

**Return type** [dict](https://docs.python.org/2.7/library/stdtypes.html#dict) ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict))

#### `addRelationship(relationship)`

Add relationship between two existing nodes

**Parameters** `relationship` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – relationship to be added between two nodes

**Returns** success of the addition

**Return type** Boolean

#### `removeRelationship(relationship)`

Remove the relationship between two nodes in the DB.

**Parameters** `relationship` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – the relationship to be removed

**Returns** the success of the removal

**Return type** Boolean

#### `addDecomp(nf_id, decomp_id, decomp)`

Add new decomposition for a high-level NF.

**Parameters**

- `nf_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the NF for which a decomposition is added
- `decomp_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the new decomposition
- `decomp` (*Networkx.Digraph*) – the decomposition to be added to the DB

**Returns** success of the addition

**Return type** Boolean

#### `removeDecomp(decomp_id)`

Remove a decomposition from the DB.

**Parameters** `decomp_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the decomposition to be removed from the DB

**Returns** the success of the removal

**Return type** Boolean

#### `getSingleDecomp(decomp_id)`

Get a decomposition with id decomp\_id.

: param decomp\_id: the id of the decomposition to be returned : type decomp\_id: str : return: decomposition with id equal to decomp\_id : rtype: tuple of networkx.DiGraph and Relationships

**getDecomps (nffg)**

Get all decompositions for a given nffg.

: param nffg: the nffg for which the decompositions should be returned : type nffg: nffg : return: all the decompositions for the given nffg : rtype: dict

**removeGraphDB ()**

Remove all nodes and relationships from the DB.

**Returns** None

**initialize()**

Initialize NFIB with test data.

**\_NFIBManager\_\_initialize()**

Initialize NFIB with test data.

**\_NFIBManager\_\_suppress\_neo4j\_logging (level=None)**

Suppress annoying and detailed logging of *py2neo* and *httpstream* packages.

**Parameters** **level** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – level of logging (default: WARNING)

**Returns** None

**policy\_enforcement.py module** Contains functionality related to policy enforcement.

PolicyEnforcementMetaClass

PolicyEnforcementError

PolicyEnforcement

*PolicyEnforcementError* represents a violation during the policy checking process.

*PolicyEnforcementMetaClass* contains the main general logic which handles the Virtualizers and enforce policies.

*PolicyEnforcement* implements the actual enforcement logic.

**Module contents** Contains functionality related to policy enforcement.

**exception escape.orchest.policy\_enforcement.PolicyEnforcementError**

Bases: *exceptions.RuntimeError* (<https://docs.python.org/2.7/library/exceptions.html#exceptions.RuntimeError>)

Exception class to signal policy enforcement error.

**class escape.orchest.policy\_enforcement.PolicyEnforcementMetaClass**Bases: [type](#) (<https://docs.python.org/2.7/library/functions.html#type>)

Meta class for handling policy enforcement in the context of classes inherited from [AbstractVirtualizer](#).

If the [PolicyEnforcement](#) class contains a function which name matches one in the actual Virtualizer then PolicyEnforcement's function will be called first.

**Warning:** Therefore the function names must be identical!

**Note:** If policy checking fails a [PolicyEnforcementError](#) should be raised and handled in a higher layer..

To use policy checking set the following class attribute:

```
__metaclass__ = PolicyEnforcementMetaClass
```

**static \_\_new\_\_(mcs, name, bases, attrs)**

Magic function called before subordinated class even created

**Parameters**

- **name** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – given class name
- **bases** ([tuple](#) (<https://docs.python.org/2.7/library/functions.html#tuple>)) – bases of the class
- **attrs** ([dict](#) (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – given attributes

**Returns** inferred class instance

**Return type** [AbstractVirtualizer](#)

**classmethod get\_wrapper(mcs, orig\_func, hooks)**

Return a decorator function which do the policy enforcement check.

**Parameters**

- **orig\_func** ([func](#)) – original function
- **hooks** ([tuple](#) (<https://docs.python.org/2.7/library/functions.html#tuple>)) – tuple of pre and post checking functions

**Raise** PolicyEnforcementError

**Returns** decorator function

**Return type** func

**class escape.orchest.policy\_enforcement.PolicyEnforcement**Bases: [object](#) (<https://docs.python.org/2.7/library/functions.html#object>)

Proxy class for policy checking.

Contains the policy checking function.

Binding is based on function name (checking function have to exist in this class and its name have to stand for the *pre\_* or *post\_* prefix and the name of the checked function).

**Warning:** Every PRE policy checking function is classmethod and need to have two parameter for nameless (args) and named(kwargs) params:

Example:

```
def pre_sanity_check (cls, args, kwargs):
```

**Warning:** Every POST policy checking function is classmethod and need to have three parameter for nameless (args), named (kwargs) params and return value:

Example:

```
def post_sanity_check (cls, args, kwargs, ret_value):
```

---

**Note:** The first element of args is the supervised Virtualizer ('self' param in the original function)

---

`__init__()`

Init

**classmethod `pre_sanity_check` (args, kwargs)**

Implements the the sanity check before virtualizer's sanity check is called.

#### Parameters

- **args** (*tuple*) (<https://docs.python.org/2.7/library/functions.html#tuple>) – original nameless arguments
- **kwargs** (*dict*) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – original named arguments

#### Returns None

**classmethod `post_sanity_check` (args, kwargs, ret\_value)**

Implements the the sanity check after virtualizer's sanity check is called.

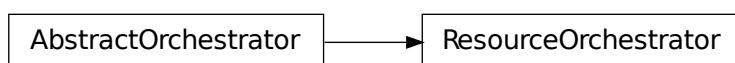
#### Parameters

- **args** (*tuple*) (<https://docs.python.org/2.7/library/functions.html#tuple>) – original nameless arguments
- **kwargs** (*dict*) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – original named arguments
- **ret\_value** – return value of Virtualizer's policy check function

#### Returns None

**ros\_orchestration.py module** Contains classes relevant to Resource Orchestration Sublayer functionality.

NFFGManager



`ResourceOrchestrator` orchestrates `NFFG` mapping and centralize layer logic.

`NFFGManager` stores and handles Network Function Forwarding Graphs.

**Module contents** Contains classes relevant to Resource Orchestration Sublayer functionality.

**class** `escape.orchest.ros_orchestration.ResourceOrchestrator` (`layer_API`)

Bases: `escape.util.mapping.AbstractOrchestrator`

Main class for the handling of the ROS-level mapping functions.

**DEFAULT\_MAPPER**

alias of `ResourceOrchestrationMapper`

**\_\_init\_\_** (`layer_API`)

Initialize main Resource Orchestration Layer components.

**Parameters** `layer_API` (`ResourceOrchestrationAPI`) – layer API instance

**Returns** None

**instantiate\_nffg** (`nffg`)

Main API function for NF-FG instantiation.

**Parameters** `nffg` (`NFFG`) – NFFG instance

**Returns** mapped NFFG instance

**Return type** `NFFG`

**class** `escape.orchest.ros_orchestration.NFFGManager`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Store, handle and organize Network Function Forwarding Graphs.

**\_\_init\_\_** ()

Init.

**save** (`nffg`)

Save NF-FG in a dict.

**Parameters** `nffg` (`NFFG`) – Network Function Forwarding Graph

**Returns** generated ID of given NF-FG

**Return type** `int` (<https://docs.python.org/2.7/library/functions.html#int>)

**\_generate\_id** (`nffg`)

Try to generate a unique id for NFFG.

**Parameters** `nffg` (`NFFG`) – NFFG

**get** (`nffg_id`)

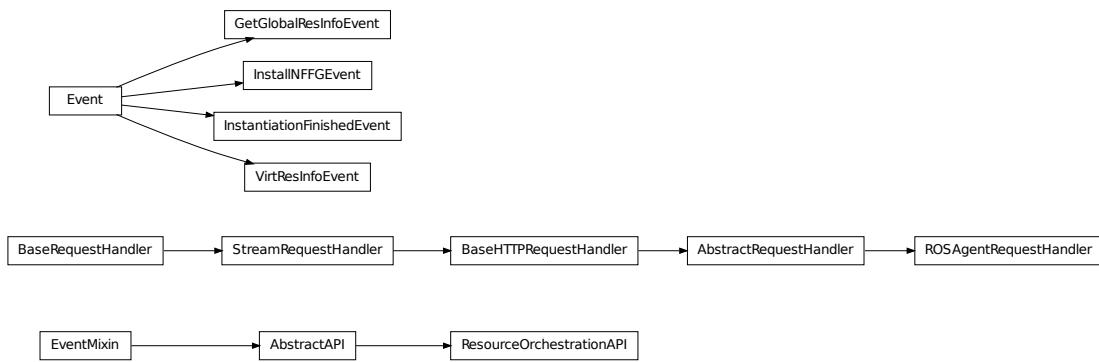
Return NF-FG with given id.

**Parameters** `nffg_id` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – ID of NF-FG

**Returns** NF-FG instance

**Return type** `NFFG`

**ros\_API.py module** Implements the platform and POX dependent logic for the Resource Orchestration Sublayer.



*InstallNFFGEvent* can send mapped NF-FG to the lower layer.

*VirtResInfoEvent* can send back virtual resource info requested from upper layer.

*GetGlobalResInfoEvent* can request global resource info from lower layer.

*InstantiationFinishedEvent* can signal info about NFFG instantiation.

*ROSAgentRequestHandler* implements the REST-API functions for agent mode.

*ResourceOrchestrationAPI* represents the ROS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Resource Orchestration Sublayer.

**class escape.orchest.ros\_API.*InstallNFFGEvent* (*mapped\_nffg*)**

Bases: pox.lib.revent.revent.Event

Event for passing mapped *NFFG* to Controller Adaptation Sublayer.

**\_\_init\_\_** (*mapped\_nffg*)

Init

**Parameters** *mapped\_nffg* (*NFFG*) – NF-FG graph need to be installed

**class escape.orchest.ros\_API.*VirtResInfoEvent* (*virtualizer*)**

Bases: pox.lib.revent.revent.Event

Event for sending back requested Virtual view an a specific Virtualizer.

**\_\_init\_\_** (*virtualizer*)

Init

**Parameters** *virtualizer* (*AbstractVirtualizer*) – virtual resource info

**class escape.orchest.ros\_API.*GetGlobalResInfoEvent***

Bases: pox.lib.revent.revent.Event

Event for requesting DomainVirtualizer from CAS.

**class escape.orchest.ros\_API.*InstantiationFinishedEvent* (*id, result, error=None*)**

Bases: pox.lib.revent.revent.Event

Event for signalling end of mapping process finished with success.

**\_\_init\_\_** (*id, result, error=None*)

**class escape.orchest.ros\_API.*CfOrRequestHandler* (*request, client\_address, server*)**

Bases: *escape.util.api.AbstractRequestHandler*

Request Handler for the Cf-OR interface.

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually: use relevant helper function of core object: `callLater/raiseLater` or use `schedule_as_coop_task` decorator defined in util.misc on the called function.

Contains handler functions for REST-API.

```
request_perm = {'POST': ('ping', 'get_config', 'edit_config'), 'GET': ('ping', 'version', 'operations', 'get_config')}
bounded_layer = 'orchestration'
static_prefix = 'cfor'

log = <logging.Logger object at 0x5e85550>
rpc_mapper = {'edit-config': 'edit_config', 'get-config': 'get_config'}

__init__ (request, client_address, server)
    Init.

get_config()
    Response configuration.

edit_config()
    Receive configuration and initiate orchestration.
```

**class** `escape.orchest.ros_API.ROSAgentRequestHandler` (*request, client\_address, server*)  
Bases: `escape.util.api.AbstractRequestHandler`

Request Handler for agent behaviour in Resource Orchestration SubLayer.

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually: use relevant helper function of core object: `callLater/raiseLater` or use `schedule_as_coop_task` decorator defined in util.misc on the called function.

Contains handler functions for REST-API.

```
request_perm = {'POST': ('ping', 'get_config', 'edit_config'), 'GET': ('ping', 'version', 'operations', 'get_config')}
bounded_layer = 'orchestration'
static_prefix = 'escape'

log = <logging.Logger object at 0x5e85b10>
rpc_mapper = {'edit-config': 'edit_config', 'get-config': 'get_config'}

__init__ (request, client_address, server)
    Init.

get_config()
    Response configuration.

edit_config()
    Receive configuration and initiate orchestration.

_update_REMOTE_ESCAPE_domain (nffg_part)
    Update domain descriptor of infra: REMOTE -> INTERNAL

    Parameters nffg_part (NFFG) – NF-FG need to be updated
    Returns updated NFFG
    Return type NFFG
```

```
class escape.orchest.ros_API.ResourceOrchestrationAPI(standalone=False,
                                                       **kwargs)
    Bases: escape.util.api.AbstractAPI
    Entry point for Resource Orchestration Sublayer (ROS).
    Maintain the contact with other UNIFY layers.
    Implement the SI - Or reference point.
    _core_name = 'orchestration'
    dependencies = ('adaptation',)
    __init__(standalone=False, **kwargs)
```

**See also:**`AbstractAPI.__init__()``initialize()`**See also:**`AbstractAPI.initialize()``shutdown(event)`**See also:**`AbstractAPI.shutdown()``_initiate_ros_api()`

Initialize and setup REST API in a different thread.

If agent\_mod is set rewrite the received NFFG domain from REMOTE to INTERNAL.

**Returns** None`_initiate_cfor_api()`

Initialize and setup REST API in a different thread.

**Returns** None`_handle_NFFGMappingFinishedEvent(event)`Handle NFFGMappingFinishedEvent and proceed with `NFFG` installation.**Parameters** `event(NFFGMappingFinishedEvent)` – event object**Returns** None`api_ros_get_config()`

Implementation of REST-API RPC: get-config.

**Returns** dump of global view (DoV)**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)`api_ros_edit_config(nffg)`

Implementation of REST-API RPC: edit-config

**Parameters** `nffg(NFFG)` – NFFG need to deploy`api_cfor_get_config()`

Implementation of Cf-Or REST-API RPC: get-config.

**Returns** dump of a single BiSBiS view based on DoV**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**api\_cfor\_edit\_config (nffg)**  
 Implementation of Cf-Or REST-API RPC: edit-config

**Parameters** `nffg` (*NFFG*) – NFFG need to deploy

**\_handle\_InstantiateNFFGEvent (event)**  
 Instantiate given NF-FG (UNIFY SI - Or API).

**Parameters** `event` (*InstantiateNFFGEvent*) – event object contains NF-FG

**Returns** None

**\_install\_NFFG (mapped\_nffg)**  
 Send mapped *NFFG* to Controller Adaptation Sublayer in an implementation-specific way.

General function which is used from microtask and Python thread also.

**Parameters** `mapped_nffg` (*NFFG*) – mapped NF-FG

**Returns** None

**\_handle\_GetVirtResInfoEvent (event)**  
 Generate virtual resource info and send back to SAS.

**Parameters** `event` (*GetVirtResInfoEvent*) – event object contains service layer id

**Returns** None

**\_handle\_MissingGlobalViewEvent (event)**  
 Request Global infrastructure View from CAS (UNIFY Or - CA API).

Invoked when a MissingGlobalViewEvent raised.

**Parameters** `event` (*MissingGlobalViewEvent*) – event object

**Returns** None

**\_handle\_GlobalResInfoEvent (event)**  
 Save requested Global Infrastructure View as the DomainVirtualizer.

**Parameters** `event` (*GlobalResInfoEvent*) – event object contains resource info

**Returns** None

**\_handle\_InstallationFinishedEvent (event)**  
 Get information from NFFG installation process.

**Parameters** `event` (*InstallationFinishedEvent*) – event object info

**Returns** None

**\_ResourceOrchestrationAPI\_\_proceed\_instantiation (\*args, \*\*kwargs)**  
 Helper function to instantiate the NFFG mapping from different source.

**Parameters** `nffg` (*NFFG*) – pre-mapped service request

**Returns** None

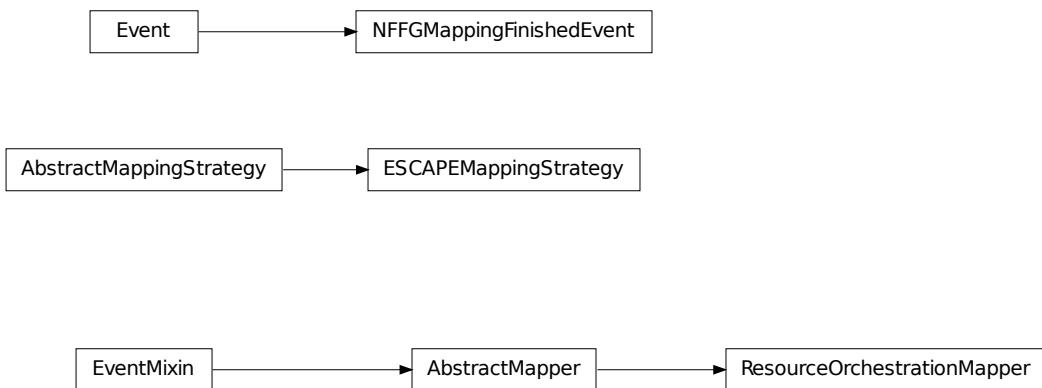
**\_ResourceOrchestrationAPI\_\_update\_nffg (nffg\_part)**  
 Update domain descriptor of inbras: REMOTE -> INTERNAL

**Parameters** `nffg_part` (*NFFG*) – NF-FG need to be updated

**Returns** updated NFFG

**Return type** *NFFG*

**ros\_mapping.py module** Contains classes which implement *NFFG* mapping functionality.



*ESCAPEMappingStrategy* implements a default *NFFG* mapping algorithm of ESCAPEv2.

*NFFGMappingFinishedEvent* can signal the state of NFFG mapping.

*ResourceOrchestrationMapper* perform the supplementary tasks for *NFFG* mapping.

**Module contents** Contains classes which implement *NFFG* mapping functionality.

**class escape.orchest.ros\_mapping.ESCAPEMappingStrategy**

Bases: *escape.util.mapping.AbstractMappingStrategy*

Implement a strategy to map initial *NFFG* into extended *NFFG*.

**\_\_init\_\_(self)**  
Init

**classmethod map(self, graph, resource)**  
Default mapping algorithm of ESCAPEv2.

#### Parameters

- **graph** (*NFFG*) – Network Function Forwarding Graph
- **resource** (*NFFG*) – global virtual resource info

**Returns** mapped Network Function Forwarding Graph

**Return type** *NFFG*

**class escape.orchest.ros\_mapping.NFFGMappingFinishedEvent(nffg)**

Bases: *pox.lib.revent.revent.Event*

Event for signaling the end of NF-FG mapping.

**\_\_init\_\_(self, nffg)**  
Init.

**Parameters** **nffg** (*NFFG*) – NF-FG need to be installed

**class escape.orchest.ros\_mapping.ResourceOrchestrationMapper(strategy=None)**

Bases: *escape.util.mapping.AbstractMapper*

Helper class for mapping NF-FG on global virtual view.

**\_eventMixin\_events = set([<class 'escape.orchest.ros\_mapping.NFFGMappingFinishedEvent'>])**

**DEFAULT\_STRATEGY**

alias of *ESCAPEMappingStrategy*

**`__init__(strategy=None)`**

Init Resource Orchestrator mapper.

**Returns** None

**`_perform_mapping(input_graph, resource_view)`**

Orchestrates mapping of given NF-FG on given global resource.

**Parameters**

- **input\_graph** (*NFFG*) – Network Function Forwarding Graph
- **resource\_view** (*DomainVirtualizer*) – global resource view

**Returns** mapped Network Function Forwarding Graph

**Return type** *NFFG*

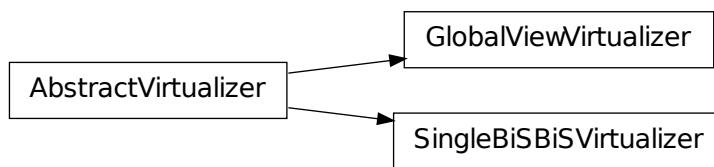
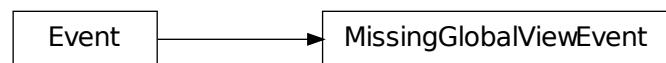
**`_mapping_finished(nffg)`**

Called from a separate thread when the mapping process is finished.

**Parameters** **nffg** (*NFFG*) – mapped NF-FG

**Returns** None

***virtualization\_mgmt.py* module** Contains components relevant to virtualization of resources and views.



*MissingGlobalViewEvent* can signal missing global view.

*AbstractVirtualizer* contains the central logic of Virtualizers.

*GlobalViewVirtualizer* implements a non-filtering/non-virtualizing logic.

*SingleBiSBiSVirtualizer* implement the default, 1-Bis-Bis virtualization logic of the Resource Orchestration Sublayer.

*VirtualizerManager* stores and handles the virtualizers.

**Module contents** Contains components relevant to virtualization of resources and views.

**class** `escape.orchest.virtualization_mgmt.MissingGlobalViewEvent`  
 Bases: `pox.lib.revent.revent.Event`

Event for signaling missing global resource view.

**class** `escape.orchest.virtualization_mgmt.AbstractVirtualizer(id)`  
 Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for actual Virtualizers.

Follows the Proxy design pattern.

**\_\_metaclass\_\_**

alias of `PolicyEnforcementMetaClass`

**\_\_init\_\_(id)**

Init.

**Parameters** `id` – id of the assigned entity

**Type** `id`: str

**\_\_str\_\_()**

**\_\_repr\_\_()**

**get\_resource\_info()**

Hides object's mechanism and return with a resource object derived from `NFFG`.

**Warning:** Derived class have to override this function

**Raise** `NotImplementedError`

**Returns** resource info

**Return type** `NFFG`

**sanity\_check(\*args, \*\*kwargs)**

Place-holder for sanity check which implemented in `PolicyEnforcement`.

**Parameters** `nffg` (`NFFG`) – NFFG instance

**Returns** None

**class** `escape.orchest.virtualization_mgmt.GlobalViewVirtualizer(global_view, id)`  
 Bases: `escape.orchest.virtualization_mgmt.AbstractVirtualizer`

Virtualizer class for experimenting and testing.

No filtering, just offer the whole global resource view.

**\_\_init\_\_(global\_view, id)**

Init.

**Parameters**

- **global\_view** (`DomainVirtualizer`) – virtualizer instance represents the global view
- **id** – id of the assigned entity

**Type** `id`: str

**get\_resource\_info()**

Hides object's mechanism and return with a resource object derived from `NFFG`.

**Returns** Virtual resource info as an NFFG

**Return type** `NFFG`

```
class escape.orchest.virtualization_mgmt.SingleBiSBiSVirtualizer(global_view,
    id)
Bases: escape.orchest.virtualization_mgmt.AbstractVirtualizer
```

Actual Virtualizer class for ESCAPEv2.

Default virtualizer class which offer the trivial one BisBis view.

```
__init__(global_view, id)
    Init.
```

#### Parameters

- **global\_view** ([DomainVirtualizer](#)) – virtualizer instance represents the global view
- **id** – id of the assigned entity

**Type** id: str

```
get_resource_info()
```

Hides object's mechanism and return with a resource object derived from [NFFG](#).

**Returns** Virtual resource info as an NFFG

**Return type** [NFFG](#)

```
_generate_one_bisbis()
```

Generate trivial virtual topology a.k.a 1 BisBis.

**Returns** 1 Bisbis topo

**Return type** [NFFG](#)

```
class escape.orchest.virtualization_mgmt.VirtualizerManager
```

Bases: pox.lib.revent.revent.EventMixin

Store, handle and organize instances of derived classes of [AbstractVirtualizer](#).

```
_eventMixin_events = set([<class 'escape.orchest.virtualization_mgmt.MissingGlobalViewEvent'>])
```

```
TYPES = {'SINGLE': <class 'escape.orchest.virtualization_mgmt.SingleBiSBiSVirtualizer'>, 'GLOBAL': <class 'escap
```

```
__init__()
```

Initialize virtualizer manager.

**Returns** None

**dov**

Getter method for the DomainVirtualizer.

Request DoV from Adaptation if it hasn't set yet.

Use: `virtualizerManager.dov`.

**Returns** Domain Virtualizer (DoV)

**Return type** [DomainVirtualizer](#)

```
get_virtual_view(virtualizer_id, type=None, cls=None)
```

Return the Virtual View as a derived class of [AbstractVirtualizer](#).

#### Parameters

- **virtualizer\_id** (int or str) – unique id of the requested Virtual view
- **type** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – type of the Virtualizer predefined in this class
- **cls** ([AbstractVirtualizer](#)) – specific Virtualizer class if type is not given

**Returns** virtual view

**Return type** [AbstractVirtualizer](#)

**\_generate\_single\_view(id)**

Generate a Single BiSBiS virtualizer, store and return with it.

**Parameters** `id` (*int or str*) – unique virtualizer id

**Returns** generated Virtualizer

**Return type** `SingleBiSBiSVirtualizer`

**\_generate\_global\_view(id)**

Generate a Global View virtualizer, store and return with it.

**Parameters** `id` (*int or str*) – unique virtualizer id

**Returns** generated Virtualizer

**Return type** `GlobalViewVirtualizer`

**`escape.adapt` package**

Sublayer for classes related to UNIFY’s Controller Adaptation Sublayer (CAS)

**Submodules**

**`adaptation.py` module** Contains classes relevant to the main adaptation function of the Controller Adaptation Sublayer.

DomainResourceManager

ControllerAdapter

ComponentConfigurator

AbstractVirtualizer

DomainVirtualizer

`ComponentConfigurator` creates, initializes, stores and manages different adaptation components, i.e. derived classes of `AbstractDomainManager` and `AbstractESCAPEAdapter`.

`ControllerAdapter` implements the centralized functionality of high-level adaptation and installation of `NFFG`.

`DomainVirtualizer` implements the standard virtualization/generalization logic of the Resource Orchestration Sublayer.

*DomainResourceManager* stores and manages the global Virtualizer.

**Module contents** Contains classes relevant to the main adaptation function of the Controller Adaptation Sub-layer

**class** `escape.adapt.adaptation.ComponentConfigurator (ca, lazy_load=True)`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Initialize, configure and store DomainManager objects. Use global config to create managers and adapters.

Follows Component Configurator design pattern.

**\_\_init\_\_ (ca, lazy\_load=True)**

For domain adapters the configurator checks the CONFIG first.

**Warning:** Adapter classes must be subclass of AbstractESCAPEAdapter

**Note:** Arbitrary domain adapters is searched in `escape.adapt.domain_adapters`

### Parameters

- **ca** (`ControllerAdapter`) – ControllerAdapter instance
- **lazy\_load** (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – load adapters only at first reference (default: True)

**get\_mgr (domain\_name)**

Return the DomainManager with given name and create+start if needed.

**Parameters** `domain_name (str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager

**Returns** None

**start\_mgr (domain\_name, autostart=True)**

Create, initialize and start a DomainManager with given name and start the manager by default.

### Parameters

- **domain\_name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager
- **autostart** (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – also start the domain manager (default: True)

**Returns** domain manager

**Return type** `AbstractDomainManager`

**stop\_mgr (domain\_name)**

Stop and derefer a DomainManager with given name and remove from the repository also.

**Parameters** `domain_name (str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager

**Returns** None

**is\_started (domain\_name)**

Return with the value the given domain manager is started or not.

**Parameters** `domain_name (str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager

**Returns** is loaded or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**components**

Return the dict of initiated Domain managers.

**Returns** container of initiated DomainManagers

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**initiated****\_\_iter\_\_()**

Return with an iterator over the (domain\_name, DomainManager) items.

**\_\_getitem\_\_(item)**

Return with the DomainManager given by name: item.

**Parameters** `item` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – component name

**Returns** component

**Return type** `AbstractDomainManager`

**load\_component(component\_name)**

Load given component (DomainAdapter/DomainManager) from config. Initiate the given component class, pass the additional attributes, register the event listeners and return with the newly created object.

**Parameters** `component_name` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – component's name

**Returns** initiated component

**Return type** `AbstractESCAPEAdapter` or `AbstractDomainManager`

**load\_default\_mgrs()**

Initiate and start default DomainManagers defined in CONFIG.

**Returns** None

**load\_internal\_mgr()**

Initiate the DomainManager for the internal domain.

**Returns** None

**clear\_initiated\_mgrs()**

Clear initiated DomainManagers based on the first received config.

**Returns** None

**stop\_initiated\_mgrs()**

Stop initiated DomainManagers.

**Returns** None

**class escape.adapt.adaptation.ControllerAdapter(layer\_API, with\_infr=False)**

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Higher-level class for `NFFG` adaptation between multiple domains.

`DOMAIN_MAPPING = {‘OPENSTACK’: ‘OPENSTACK’, ‘SDN’: ‘SDN’, ‘INTERNAL’: ‘INTERNAL’, ‘UNIVERSAL’: ‘UNIVERSAL’}`

**\_\_init\_\_(layer\_API, with\_infr=False)**

Initialize Controller adapter.

For domain components the ControllerAdapter checks the CONFIG first.

**Parameters**

- `layer_API` (`ControllerAdaptationAPI`) – layer API instance
- `with_infr` (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – using emulated infrastructure (default: False)

**shutdown ()**

Shutdown ControllerAdapter, related components and stop DomainManagers.

**Returns** None

**install\_nffg (mapped\_nffg)**

Start NF-FG installation.

Process given *NFFG*, slice information self.\_\_global\_nffg on domains and invoke DomainManagers to install domain specific parts.

**Parameters** *mapped\_nffg* (*NFFG*) – mapped NF-FG instance which need to be installed

**Returns** None or internal domain NFFG part

**\_handle\_DomainChangedEvent (event)**

Handle DomainChangedEvents, process changes and store relevant information in DomainResourceManager.

**\_split\_into\_domains (nffg)**

Split given *NFFG* into separate parts self.\_\_global\_nffg on original domains.

**Warning:** Not implemented yet!

**Parameters** *nffg* (*NFFG*) – mapped NFFG object

**Returns** sliced parts as a list of (domain\_name, nffg\_part) tuples

**Return type** list (<https://docs.python.org/2.7/library/functions.html#list>)

**update\_dov (nffg\_part)**

Update the global view with installed Nfs/Flowrules.

**class** escape.adapt.adaptation.DomainVirtualizer (*domainResManager*,  
  *global\_res=None*)

Bases: *escape.orchest.virtualization\_mgmt.AbstractVirtualizer*

Specific Virtualizer class for global domain virtualization.

Implement the same interface as *AbstractVirtualizer*

Use *NFFG* format to store the global infrastructure info.

**\_\_init\_\_ (domainResManager, global\_res=None)**

Init.

**Parameters**

- **domainResManager** (*DomainResourceManager*) – domain resource manager
- **global\_res** (*NFFG*) – initial global resource (optional)

**Returns** None

**name****\_\_str\_\_ ()****\_\_repr\_\_ ()****get\_resource\_info ()**

Return the global resource info represented this class.

**Returns** global resource info

**Return type** *NFFG*

**set\_domain\_as\_global\_view (domain, nffg)**

Set the copy of given NFFG as the global view of DoV.

**Parameters** *nffg* (*NFFG*) – NFFG instance intended to use as the global view

**Returns** None

**merge\_domain\_into\_dov** (*domain, nffg*)

Add a newly detected domain to DoV.

Based on the feature: escape.util.nffg.NFFGToolBox#merge\_domains

**update\_global\_view** (*global\_nffg*)

Update the merged Global view with the given probably modified global view.

**Parameters** *global\_nffg* (*NFFG*) – updated global view which replace the stored one

**update\_domain\_view** (*domain, nffg*)

Update the existing domain in the merged Global view.

**class** `escape.adapt.adaptation.DomainResourceManager`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Handle and store the global resources view.

**\_\_init\_\_** ()

Init.

**get\_global\_view** ()

Getter for *DomainVirtualizer*.

**Returns** global infrastructure view as the Domain Virtualizer

**Return type** *DomainVirtualizer*

**update\_domain\_resource** (*domain, nffg*)

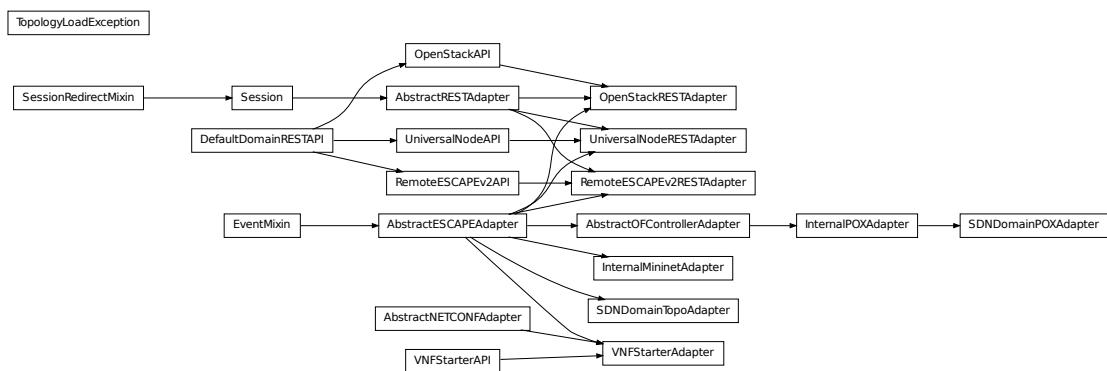
Update the global view data with the specific domain info.

**Parameters**

- **domain** (*str*) (<https://docs.python.org/2.7/library/functions.html#str>) – domain name
- **nffg** (*NFFG*) – infrastructure info collected from the domain

**Returns** None

**adapters.py module** Contains Adapter classes which contains protocol and technology specific details for the connections between ESCAPEv2 and other different domains.



*InternalPOXAdapter* implements the OF controller functionality for the Mininet-based emulated topology.

*SDNDomainPOXAdapter* implements the OF controller functionality for the external SDN/OpenFlow switches.

*InternalMininetAdapter* implements Mininet related functionality transparently e.g. start/stop/clean topology built from an :any:'NFFG'.

*SDNDomainTopoAdapter* implements SDN topology related functions.

`VNFStarterAdapter` is a helper/wrapper class for vnf\_starter NETCONF module.

`RemoteESCAPEv2RESTAdapter` is a wrapper class for REST-based communication with another ESCAPE instance started in agent mode.

`OpenStackRESTAdapter` is a wrapper class for OpenStack-REST-like API functions.

`UniversalNodeRESTAdapter` is a wrapper class for REST-like communication with the Universal Node domain.

**Module contents** Contains Adapter classes which contains protocol and technology specific details for the connections between ESCAPEv2 and other different domains.

#### `exception escape.adapt.adapters.TopologyLoadException`

Bases: `exceptions.Exception` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception>)

Exception class for topology errors.

`class escape.adapt.adapters.InternalPOXAdapter (name=None, address='127.0.0.1', port=6653, keepalive=False)`

Bases: `escape.util.domain.AbstractOFControllerAdapter`

Adapter class to handle communication with internal POX OpenFlow controller.

Can be used to define a controller (based on POX) for other external domains.

`name = 'INTERNAL-POX'`

`infra_to_dpid = {}`

`saps = {}`

`__init__(name=None, address='127.0.0.1', port=6653, keepalive=False)`

Initialize attributes, register specific connection Arbiter if needed and set up listening of OpenFlow events.

#### Parameters

- `name` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name used to register component into pox.core
- `address` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – socket address (default: 127.0.0.1)
- `port` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – socket port (default: 6633)

`check_domain_reachable()`

Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

`get_topology_resource()`

Return with the topology description as an `NFFG`.

**Returns** the emulated topology description

**Return type** `NFFG`

`_handle_ConnectionUp(event)`

Handle incoming OpenFlow connections.

`_handle_ConnectionDown(event)`

Handle disconnected device.

`_identify_ovs_device(connection)`

Identify the representing Node of the OVS switch according to the given connection and extend the dpid-infra binding dictionary.

The discovery algorithm takes the advantage of the naming convention of Mininet for interfaces in an OVS switch e.g.: EE1, EE1-eth1, EE1-eth2, etc.

**Parameters** `connection` (`pox.openflow.of_01.Connection`) – inner Connection class of POX

**Returns** None

```
class escape.adapt.adapters.SDNDomainPOXAdapter(name=None, address='0.0.0.0',
                                                port=6653, keepalive=False)
Bases: escape.adapt.adapters.InternalPOXAdapter
```

Adapter class to handle communication with external SDN switches.

`name = 'SDN-POX'`

`infra_to_dpid = {'MT2': 365441792307142, 'MT1': 365441792306724}`

`dpid_to_infra = {365441792306724: 'MT1', 365441792307142: 'MT2'}`

`__init__(name=None, address='0.0.0.0', port=6653, keepalive=False)`

`get_topology_resource()`

`check_domain_reachable()`

```
class escape.adapt.adapters.InternalMininetAdapter(net=None)
Bases: escape.util.domain.AbstractESCAPEAdapter
```

Adapter class to handle communication with Mininet domain.

Implement VNF managing API using direct access to the `mininet.net.Mininet` object.

`_eventMixin_events = set([<class 'escape.util.domain.DomainChangedEvent'>])`

`name = 'MININET'`

`__init__(net=None)`

Init.

**Parameters** `net` (`ESCAPENetworkBridge`) – set pre-defined network (optional)

`get_mn_wrapper()`

Return the specific wrapper for `mininet.net.Mininet` object represents the emulated network.

**Returns** emulated network wrapper

**Return type** `ESCAPENetworkBridge`

`check_domain_reachable()`

Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

`get_topology_resource()`

Return with the topology description as an `NFFG`.

**Returns** the emulated topology description

**Return type** `NFFG`

`get_agent_connection_params(ee_name)`

Return the connection parameters for the agent of the switch given by the `switch_name`.

**Parameters** `ee_name` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the container Node

**Returns** connection params

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

```
class escape.adapt.adapters.SDNDomainTopoAdapter (path=None)
Bases: escape.util.domain.AbstractESCAPEAdapter
```

Adapter class to return the topology description of the SDN domain.

Currently it just read the static description from file, and not discover it.

**name = ‘SDN-TOPO’**

**\_\_init\_\_** (*path=None*)

**check\_domain\_reachable()**

Checker function for domain. Naively return True.

**Returns** the domain is detected or not

**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource()**

Return with the topology description as an *NFFG* parsed from file.

**Returns** the static topology description

**Return type** *NFFG*

**\_SDNDomainTopoAdapter\_\_init\_from\_CONFIG** (*path=None*)

Load a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name SDN-TOPO.

**Parameters** **path** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – additional file path

**Returns** None

```
class escape.adapt.adapters.VNFStarterAdapter (**kwargs)
```

Bases: *escape.util.netconf.AbstractNETCONFAdapter*, *escape.util.domain.AbstractESCAPEAdapter*, *escape.util.domain.VNFStarterAPI*

This class is devoted to provide NETCONF specific functions for vnf\_starter module. Documentation is transferred from *vnf\_starter.yang*.

This class is devoted to start and stop CLICK-based VNFs that will be connected to a mininet switch.

Follows the MixIn design pattern approach to support NETCONF functionality.

**RPC\_NAMESPACE = u’http://csikor.tmit.bme.hu/netconf/unify/vnf\_starter’**

**name = ‘VNFStarter’**

**\_\_init\_\_** (\*\*kwargs)

Init.

#### Parameters

- **server** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – server address
- **port** (*int* (<https://docs.python.org/2.7/library/functions.html#int>)) – port number
- **username** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – username
- **password** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – password
- **timeout** (*int* (<https://docs.python.org/2.7/library/functions.html#int>)) – connection timeout (default=30)

#### Returns

**check\_domain\_reachable()**

Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource()**

Return with the topology description as an [NFFG](#).

**Returns** the emulated topology description

**Return type** [NFFG](#)

**update\_connection\_params(\*\*kwargs)**

Update connection params.

**Returns** only updated params

**Return type** [dict](#) (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**\_invoke\_rpc(request\_data)**

Override parent function to catch and log exceptions gracefully.

**initiateVNF(vnf\_type, vnf\_description=None, options=None)**

This RCP will start a VNF.

0.initiate new VNF (initiate datastructure, generate unique ID)

1.set its arguments (control port, control ip, and VNF type/command)

2.returns the connection data, which from the vnf\_id is the most important

**Parameters**

- **vnf\_type** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – pre-defined VNF type (see in vnf\_starter/available\_vnfs)
- **vnf\_description** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – Click description if there are no pre-defined type
- **options** ([collections.OrderedDict](#) (<https://docs.python.org/2.7/library/collections.html#collections.OrderedDict>)) – unlimited list of additional options as name-value pairs

**Returns** RPC reply data

**Return type** [dict](#) (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**Raises** RPCError, OperationError, TransportError

**connectVNF(vnf\_id, vnf\_port, switch\_id)**

This RPC will practically start and connect the initiated VNF/CLICK to the switch.

0.create virtualEthernet pair(s)

1.connect either end of it (them) to the given switch(es)

This RPC is also used for reconnecting a VNF. In this case, however, if the input fields are not correctly set an error occurs

**Parameters**

- **vnf\_id** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)
- **vnf\_port** ([str or int](#)) – VNF port (mandatory)
- **switch\_id** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – switch ID (mandatory)

**Returns** Returns the connected port(s) with the corresponding switch(es).

**Raises** RPCError, OperationError, TransportError

**disconnectVNF(vnf\_id, vnf\_port)**

This RPC will disconnect the VNF(s)/CLICK(s) from the switch(es).

0.ip link set uny\_0 down

- 1.ip link set uny\_1 down
- 2.(if more ports) repeat 1. and 2. with the corresponding data

**Parameters**

- **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)
- **vnf\_port** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF port (mandatory)

**Returns** reply data**Raises** RPCError, OperationError, TransportError**startVNF** (*vnf\_id*)

This RPC will actually start the VNF/CLICK instance.

**Parameters** **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)**Returns** reply data**Raises** RPCError, OperationError, TransportError**stopVNF** (*vnf\_id*)

This RPC will gracefully shut down the VNF/CLICK instance.

0.if disconnect() was not called before, we call it

1.delete virtual ethernet pairs

2.stop (kill) click

3.remove vnf's data from the data structure

**Parameters** **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)**Returns** reply data**Raises** RPCError, OperationError, TransportError**getVNFInfo** (*vnf\_id=None*)

This RPC will send back all data of all VNFs that have been initiated by this NETCONF Agent. If an input of vnf\_id is set, only that VNF's data will be sent back. Most of the data this RPC replies is used for DEBUG, however 'status' is useful for indicating to upper layers whether a VNF is UP\_AND\_RUNNING.

**Parameters** **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (default: list info about all VNF)**Returns** reply data**Raises** RPCError, OperationError, TransportError**deployNF** (*nf\_type*, *nf\_ports*, *infra\_id*, *nf\_desc=None*, *nf\_opt=None*)

Initiate and start the given NF using the general RPC calls.

**Parameters**

- **nf\_type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – pre-defined NF type (see in vnf\_starter/available\_vnfs)
- **nf\_ports** (*str or int or tuple*) – NF port number or list of ports (mandatory)
- **infra\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – id of the base node (mandatory)

- **nf\_desc** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – Click description if there are no pre-defined type
- **nf\_opt** (*collections.OrderedDict* (<https://docs.python.org/2.7/library/collections.html#collections.OrderedDict>)) – unlimited list of additional options as name-value pairs

**Returns** initiated NF description parsed from RPC reply

**Return type** *dict* (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**removeNF** (*vnf\_id*)

Stop and remove the given NF using the general RPC calls.

**class** *escape.adapt.adapters.RemoteESCAPEv2RESTAdapter* (*url*)

Bases: *escape.util.domain.AbstractRESTAdapter*, *escape.util.domain.AbstractESCAPEAdapter*, *escape.util.domain.RemoteESCAPEv2API*

This class is devoted to provide REST specific functions for remote ESCAPEv2 domain.

**name** = ‘ESCAPE-REST’

**\_\_init\_\_** (*url*)

Init.

**Parameters** **url** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – remote ESCAPEv2 RESTful API URL

**ping** ()

**get\_config** ()

**edit\_config** (*data*)

**check\_domain\_reachable** ()

**get\_topology\_resource** ()

**class** *escape.adapt.adapters.OpenStackRESTAdapter* (*url*)

Bases: *escape.util.domain.AbstractRESTAdapter*, *escape.util.domain.AbstractESCAPEAdapter*, *escape.util.domain.OpenStackAPI*

This class is devoted to provide REST specific functions for OpenStack domain.

**name** = ‘OpenStack-REST’

**\_\_init\_\_** (*url*)

Init.

**Parameters** **url** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – OpenStack RESTful API URL

**ping** ()

**get\_config** ()

**edit\_config** (*data*)

**check\_domain\_reachable** ()

**get\_topology\_resource** ()

**class** *escape.adapt.adapters.UniversalNodeRESTAdapter* (*url*)

Bases: *escape.util.domain.AbstractRESTAdapter*, *escape.util.domain.AbstractESCAPEAdapter*, *escape.util.domain.UniversalNodeAPI*

This class is devoted to provide REST specific functions for UN domain.

**name** = ‘UN-REST’

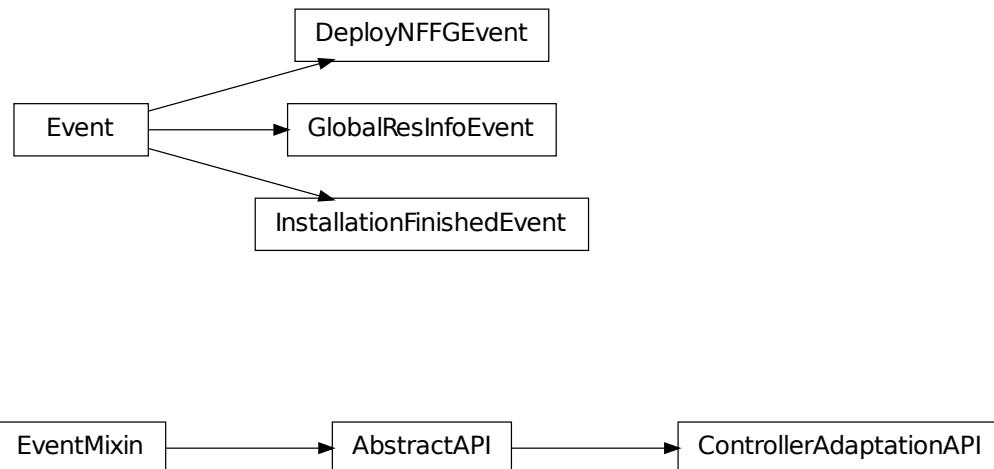
**\_\_init\_\_** (*url*)

Init.

**Parameters** `url` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – Universal Node RESTful API URL

```
ping()
get_config()
edit_config(data)
check_domain_reachable()
get_topology_resource()
```

**`cas_API.py` module** Implements the platform and POX dependent logic for the Controller Adaptation Sublayer.



`GlobalResInfoEvent` can send back global resource info requested from upper layer.

`InstallationFinishedEvent` can send back status about the NFFG installation.

`DeployNFFGEvent` can send NFFG to Infrastructure layer for deploying.

`ControllerAdaptationAPI` represents the CAS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Controller Adaptation Sublayer.

```
class escape.adapt.cas_API.GlobalResInfoEvent(dov)
    Bases: pox.lib.revent.revent.Event
```

Event for sending back requested Global Resource View.

```
__init__(dov)
    Init.
```

**Parameters** `dov` (`DomainVirtualizer`) – Domain Virtualizer which handles the Global Infrastructure View.

```
class escape.adapt.cas_API.InstallationFinishedEvent(id, result, error=None)
    Bases: pox.lib.revent.revent.Event
```

Event for signalling end of mapping process.

```
__init__(id, result, error=None)
```

```
class escape.adapt.cas_API.DeployNFFGEvent (nffg_part)
    Bases: pox.lib.revent.revent.Event

    Event for passing mapped NFFG to internally emulated network based on Mininet for testing.

    __init__ (nffg_part)
class escape.adapt.cas_API.ControllerAdaptationAPI (standalone=False, **kwargs)
    Bases: escape.util.api.AbstractAPI

    Entry point for Controller Adaptation Sublayer (CAS).

    Maintain the contact with other UNIFY layers.

    Implement the Or - Ca reference point.

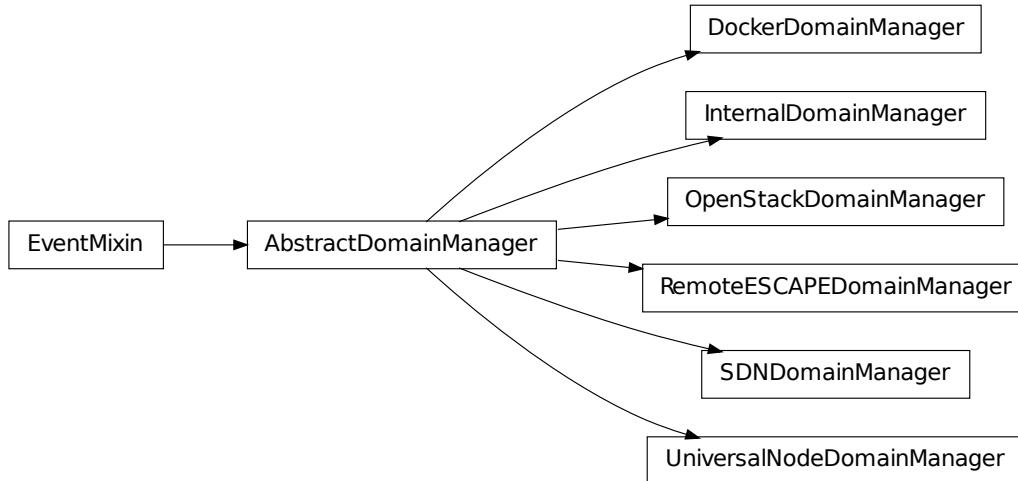
    _core_name = ‘adaptation’

    __init__ (standalone=False, **kwargs)
```

**See also:**[AbstractAPI.\\_\\_init\\_\\_\(\)](#)**initialize()****See also:**[AbstractAPI.initialize\(\)](#)**shutdown (*event*)****See also:**[AbstractAPI.shutdown\(\)](#)**\_handle\_InstallNFFGEvent (*event*)**  
Install mapped NF-FG (UNIFY Or - Ca API).**Parameters** **event** ([InstallNFFGEvent](#)) – event object contains mapped NF-FG**Returns** None**\_handle\_GetGlobalResInfoEvent (*event*)**  
Generate global resource info and send back to ROS.**Parameters** **event** ([GetGlobalResInfoEvent](#)) – event object**Returns** None**\_handle\_DeployEvent (*event*)**  
Receive processed NF-FG from domain adapter(s) and forward to Infrastructure**Parameters** **event** ([DeployNFFGEvent](#)) – event object**Returns** None**\_handle\_DeploymentFinishedEvent (*event*)**  
Receive successful NF-FG deployment event and propagate upwards**Parameters** **event** ([DeploymentFinishedEvent](#)) – event object**Returns** None**\_ControllerAdaptationAPI\_proceed\_installation (\*args, \*\*kwargs)**  
Helper function to instantiate the NFFG mapping from different source.**Parameters** **mapped\_nffg** ([NFFG](#)) – pre-mapped service request**Returns** None

**managers.py module** Contains Manager classes which contains the higher-level logic for complete domain management.

Uses Adapter classes for ensuring protocol-specific connections with entities in the particular domain.



*InternalDomainManager* represent the top class for interacting with the emulated infrastructure.

*RemoteESCAPEDomainManager* ensures the connection with a different ESCAPE instance started in agent mode.

*OpenStackDomainManager* implements the related functionality for managing the OpenStack-based domain.

*UniversalNodeDomainManager* implements the related functionality for managing the domain based on the Universal Node conception.

*DockerDomainManager* is a placeholder class for managing Docker-based network entities.

*SDNDomainManager* interacts and handles legacy OpenFlow 1.0 switches aggregated into a separate domain.

**Module contents** Contains Manager classes which contains the higher-level logic for complete domain management. Uses Adapter classes for ensuring protocol-specific connections with entities in the particular domain.

**class escape.adapt.managers.*InternalDomainManager* (\*\*kwargs)**

Bases: *escape.util.domain.AbstractDomainManager*

Manager class to handle communication with internally emulated network.

---

**Note:** Uses *InternalMininetAdapter* for managing the emulated network and *InternalPOXAdapter* for controlling the network.

---

**name = 'INTERNAL'**

**\_\_init\_\_(\*\*kwargs)**  
Init

**init(configurator, \*\*kwargs)**  
Initialize Internal domain manager.

#### Parameters

- **configurator** (*ComponentConfigurator*) – component configurator for configuring adapters

- **kwargs** (*dict* (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – optional parameters

**Returns** None

### **fini()**

Stop polling and release dependent components.

**Returns** None

### **controller\_name**

#### **\_setup\_sap\_hostnames()**

Setup hostnames in /etc/hosts for SAPs.

**Returns** None

#### **\_collect\_SAP\_infos()**

Collect necessary information from SAPs for traffic steering.

**Returns** None

### **install\_nffg(*nffg\_part*)**

Install an *NFFG* related to the internal domain.

**Parameters** *nffg\_part* (*NFFG*) – NF-FG need to be deployed

**Returns** None

### **clear\_domain()**

Infrastructure Layer has already been stopped and probably cleared.

Skip cleanup process here.

**Returns** None

### **\_delete\_nfs()**

Stop and delete deployed NFs.

**Returns** None

### **\_deploy\_nfs(*nffg\_part*)**

Install the NFs mapped in the given NFFG.

If an NF is already defined in the topology and it's state is up and running then the actual NF's initiation will be skipped!

**Parameters** *nffg\_part* (*NFFG*) – NF-FG need to be deployed

**Returns** None

### **\_delete\_flowrules(*nffg\_part*)**

Delete all flowrules from the first (default) table of all infras.

### **\_deploy\_flowrules(*nffg\_part*)**

Install the flowrules given in the NFFG.

If a flowrule is already defined it will be updated.

**Parameters** *nffg\_part* (*NFFG*) – NF-FG need to be deployed

**Returns** None

**class** *escape.adapt.managers.SDNDomainManager* (\*\*kwargs)

Bases: *escape.util.domain.AbstractDomainManager*

Manager class to handle communication with POX-controlled SDN domain.

---

**Note:** Uses *InternalPOXAdapter* for controlling the network.

---

**name = ‘SDN’**

---

```

__init__(**kwargs)
    Init

init(configurator, **kwargs)
    Initialize SDN domain manager.

    Returns None

fini()
    Stop polling and release dependent components.

    Returns None

controller_name

install_nffg(nffg_part)
    Install an NFFG related to the SDN domain.

    Parameters nffg_part (NFFG) – NF-FG need to be deployed

    Returns None

_delete_flowrules(nffg_part)
    Delete all flowrules from the first (default) table of all infras.

    Returns None

_deploy_flowrules(nffg_part)
    Install the flowrules given in the NFFG.

    If a flowrule is already defined it will be updated.

    Parameters nffg_part (NFFG) – NF-FG need to be deployed

    Returns None

clear_domain()
    Delete all flowrule in the registered SDN/OF switches.

    Returns None

class escape.adapt.managers.RemoteESCAPEDomainManager(**kwargs)
    Bases: escape.util.domain.AbstractDomainManager

    Manager class to handle communication with other ESCAPEv2 processes started in agent-mode through a REST-API which is provided by the Resource Orchestration Sublayer.



---



Note: Uses RemoteESCAPEv2RESTAdapter for communicate with the remote domain.



---


name = 'REMOTE-ESCAPE'

__init__(**kwargs)
    Init

init(configurator, **kwargs)
    Initialize Internal domain manager.

    Returns None

fini()
    Stop polling and release dependent components.

    Returns None

install_nffg(nffg_part)
    Install an NFFG related to the internal domain.

    Parameters nffg_part (NFFG) – NF-FG need to be deployed

    Returns None

```

**clear\_domain()**  
Reset remote domain based on the original (first response) topology.

**Returns** None

**class** `escape.adapt.managers.OpenStackDomainManager(**kwargs)`  
Bases: `escape.util.domain.AbstractDomainManager`

Manager class to handle communication with OpenStack domain.

---

**Note:** Uses OpenStackRESTAdapter for communicate with the remote domain.

---

**name = ‘OPENSTACK’**

**\_\_init\_\_(\*\*kwargs)**  
Init.

**init(configurator, \*\*kwargs)**  
Initialize OpenStack domain manager.

**Returns** None

**finit()**

Stop polling and release dependent components.

**Returns** None

**install\_nffg(nffg\_part)**

**clear\_domain()**  
Reset remote domain based on the original (first response) topology.

**Returns** None

**class** `escape.adapt.managers.UniversalNodeDomainManager(**kwargs)`  
Bases: `escape.util.domain.AbstractDomainManager`

Manager class to handle communication with Universal Node (UN) domain.

---

**Note:** Uses UniversalNodeRESTAdapter for communicate with the remote domain.

---

**name = ‘UN’**

**\_\_init\_\_(\*\*kwargs)**  
Init.

**init(configurator, \*\*kwargs)**  
Initialize OpenStack domain manager.

**Returns** None

**finit()**

Stop polling and release dependent components.

**Returns** None

**install\_nffg(nffg\_part)**

**clear\_domain()**  
Reset remote domain based on the original (first response) topology.

**Returns** None

**class** `escape.adapt.managers.DockerDomainManager(**kwargs)`  
Bases: `escape.util.domain.AbstractDomainManager`

Adapter class to handle communication component in a Docker domain.

**Warning:** Not implemented yet!

```
name = 'DOCKER'

__init__(**kwargs)
    Init

install_nffg(nffg_part)
clear_domain()
```

### `escape.infr` package

Sublayer for classes related to UNIFY's Infrastructure Layer (IL)

#### Submodules

`il_API.py` module Emulate UNIFY's Infrastructure Layer for testing purposes based on Mininet.



`DeploymentFinishedEvent` can send status info about NFFG deployment.

`InfrastructureLayerAPI` represents the IL layer and implement all related functionality.

**Module contents** Emulate UNIFY's Infrastructure Layer for testing purposes based on Mininet.

```
class escape.infr.il_API.DeploymentFinishedEvent(success, error=None)
    Bases: pox.lib.revent.revent.Event

    Event for signaling NF-FG deployment

    __init__(success, error=None)

class escape.infr.il_API.InfrastructureLayerAPI(standalone=False, **kwargs)
    Bases: escape.util.api.AbstractAPI

    Entry point for Infrastructure Layer (IL).

    Maintain the contact with other UNIFY layers.

    Implement a specific part of the Co - Rm reference point.

    _core_name = 'infrastructure'

    _eventMixin_events = set([<class 'escape.infr.il_API.DeploymentFinishedEvent'>])
```

`__init__(standalone=False, **kwargs)`

See also:

`AbstractAPI.__init__()`

`initialize()`

See also:

`AbstractAPI.initialize()`

`shutdown(event)`

See also:

`AbstractAPI.shutdown()`

`_handle_ComponentRegistered(event)`

Wait for controller (internal POX module)

Parameters `event` (`ComponentRegistered`) – registered component event

Returns None

`_handle_DeployNFFGEvent(*args, **kwargs)`

Install mapped NFFG part into the emulated network.

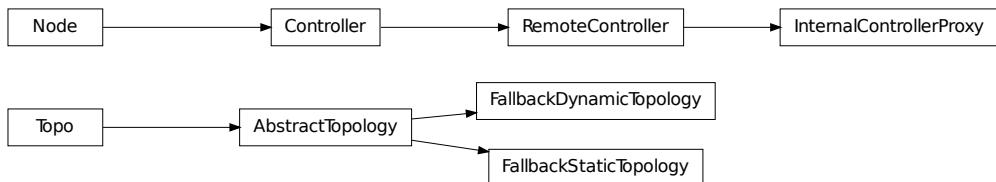
:param event:event object :return: `DeployNFFGEvent`

**`topology.py` module** Wrapper module for handling emulated test topology based on Mininet.

`TopologyBuilderException`

`ESCAPENetworkBuilder`

`ESCAPENetworkBridge`



`AbstractTopology` can represent an emulated topology for the high-level API.

`FallbackStaticTopology` represents the static fallback topology.

`FallbackDynamicTopology` represents the static fallback topology.

`InternalControllerProxy` represents the connection between the internal controller and the emulated network.

`ESCAPENetworkBridge` represents the emulated topology in high level.

`TopologyBuilderException` can signal various error related to the topology emulation.

`ESCAPENetworkBuilder` can construct an `ESCAPENetworkBridge` object.

**Module contents** Wrapper module for handling emulated test topology based on Mininet.

```
class escape.infr.topology.AbstractTopology (hopts=None, sopts=None, lopts=None, eopts=None)
```

Bases: `mininet.topo.Topo`

Abstract class for representing emulated topology.

Have the functions to build a ESCAPE-specific topology.

Can be used to define reusable topology similar to Mininet's high-level API. Reusable, convenient and pre-defined way to define a topology, but less flexible and powerful.

```
default_host_opts = None
```

```
default_switch_opts = None
```

```
default_link_opts = None
```

```
default_EE_opts = None
```

```
TYPE = None
```

```
__init__ (hopts=None, sopts=None, lopts=None, eopts=None)
```

```
construct (builder=None)
```

Base class for construct the topology.

```
static get_topo_desc ()
```

Return the NFFG object represents the specific, constructed topology

```
class escape.infr.topology.FallbackStaticTopology (hopts=None, sopts=None, lopts=None, eopts=None)
```

Bases: `escape.infr.topology.AbstractTopology`

Topology class for testing purposes and serve as a fallback topology.

Use the static way for topology compilation.

```
TYPE = 'STATIC'
```

```
construct (builder=None)
```

```
static get_topo_desc ()
```

```
class escape.infr.topology.FallbackDynamicTopology (hopts=None, sopts=None, lopts=None, eopts=None)
```

Bases: `escape.infr.topology.AbstractTopology`

Topology class for testing purposes and serve as a fallback topology.

Use the dynamic way for topology compilation.

```
TYPE = 'DYNAMIC'
```

```
construct (builder=None)
```

Set a topology with NETCONF capability for mostly testing.

**Returns** None

```
static get_topo_desc ()
```

```
class escape.infr.topology.InternalControllerProxy (name='InternalPOXController', ip='127.0.0.1', port=6653, **kwargs)
```

Bases: `mininet.node.RemoteController`

Controller class for emulated Mininet network. Making connection with internal controller initiated by InternalPOXAdapter.

**\_\_init\_\_(name='InternalPOXController', ip='127.0.0.1', port=6653, \*\*kwargs)**  
Init.

#### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the controller (default: InternalPOXController)
- **ip** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – IP address (default: 127.0.0.1)
- **port** (*int* (<https://docs.python.org/2.7/library/functions.html#int>)) – port number (default 6633)

**checkListening()**

Check the controller port is open.

**class escape.infr.topology.ESCAPENetworkBridge (network=None, topo\_desc=None)**  
Bases: *object* (<https://docs.python.org/2.7/library/functions.html#object>)

Internal class for representing the emulated topology.

Represents a container class for network elements such as switches, nodes, execution environments, links etc. Contains network management functions similar to Mininet's mid-level API extended with ESCAPEv2 related capabilities

Separate the interface using internally from original Mininet object to implement loose coupling and avoid changes caused by Mininet API changes e.g. 2.1.0 -> 2.2.0.

Follows Bridge design pattern.

**\_\_init\_\_(network=None, topo\_desc=None)**

Initialize Mininet implementation with proper attributes. Use network as the hided Mininet topology if it's given.

#### Parameters

- **topo\_desc** (*NFFG*) – static topology description e.g. the related NFFG
- **network** (*mininet.net.MininetWithControlNet*) – use this specific Mininet object for init (default: None)

**Returns** None

**network**

Internal network representation.

**Returns** network representation

**Return type** *mininet.net.MininetWithControlNet*

**runXTerms()**

Start an xterm to every SAP if it's enabled in the global config. SAP are stored as hosts in the Mininet class.

**Returns** None

**start\_network()**

Start network.

**stop\_network()**

Stop network.

**cleanup()**

Clean up junk which might be left over from old runs.

**..seealso::** *mininet.clean.cleanup()*

**get\_agent\_to\_switch(switch\_name)**

Return the agent to which the given switch is tided..

**Parameters** **switch\_name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the switch

**Returns** the agent

**Return type** mininet.node.NetconfAgent

**exception escape.infr.topology.TopologyBuilderException**

Bases: `exceptions.Exception` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception>)

Exception class for topology errors.

**class escape.infr.topology.ESCAPENetworkBuilder(*net=None, opts=None, fallback=True, run\_dry=True*)**

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Builder class for topology.

Update the network object based on the parameters if it's given or create an empty instance.

Always return with an ESCAPENetworkBridge instance which offer a generic interface for created mininet.net.Mininet object and hide implementation's nature.

Follows Builder design pattern.

**default\_opts = {‘listenPort’: None, ‘autoSetMacs’: False, ‘inNamespace’: False, ‘autoStaticArp’: True, ‘controller’: None}**

**DEFAULT\_NFFG\_FORMAT = ‘NFFG’**

**TYPE\_EE\_LOCAL = ‘LOCAL’**

**TYPE\_EE\_REMOTE = ‘REMOTE’**

**dpidBase = 1**

**dpidLen = 16**

**\_\_init\_\_(*net=None, opts=None, fallback=True, run\_dry=True*)**

Initialize NetworkBuilder.

If the topology definition is not found, an exception will be raised or an empty mininet.net.Mininet topology will be created if `run_dry` is set.

**Parameters**

- **net** (mininet.net.Mininet) – update given Mininet object instead of creating a new one
- **opts** (*dict* (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – update default options with the given opts
- **fallback** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – search for fallback topology (default: True)
- **run\_dry** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – do not raise an Exception and return with bare Mininet obj.

**Returns** None

**get\_network()**

Return the bridge to the constructed network.

**Returns** object representing the emulated network

**Return type** `ESCAPENetworkBridge`

**create\_static\_EE(*name, cls=None, \*\*params*)**

Create and add a new EE to Mininet in the static way.

This function is for only backward compatibility.

**Warning:** Not tested yet!

### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the Execution Environment
- **cls** (*mininet.node.EE*) – custom EE class/constructor (optional)
- **cores** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – Specify (real) cores that our cgroup can run on (optional)
- **frac** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – Set overall CPU fraction for this EE (optional)
- **vlanif** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – set vlan interfaces (optional)

**Returns** newly created EE object

**Return type** *mininet.node.EE*

**create\_NETCONF\_EE** (*name, type='LOCAL', \*\*params*)

Create and add a new EE to Mininet network.

The type of EE can be {local|remote} NETCONF-based.

### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the EE: switch: name, agent: agt\_+'name'
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – type of EE {local|remote}
- **opts** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – additional options for the switch in EE
- **dpid** – remote switch DPID (remote only)
- **username** – NETCONF username (remote only)
- **passwd** – NETCONF password (remote only)
- **ip** – control Interface for the agent (optional)
- **agentPort** – port to listen on for NETCONF connections, (else set automatically)
- **minPort** – first VNF control port which can be used (else set automatically)
- **cPort** – number of VNF control ports (and VNFs) which can be used ( default: 10)

**Returns** tuple of newly created *mininet.node.Agent* and *mininet.node.Switch* object

**Return type** *tuple* (<https://docs.python.org/2.7/library/functions.html#tuple>)

**create\_Switch** (*name, cls=None, \*\*params*)

Create and add a new OF switch instance to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Switch class type.

### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of switch
- **cls** (*mininet.node.Switch*) – custom switch class/constructor (optional)
- **dpid** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – DPID for switch (default: derived from name)

- **opts** ([str](https://docs.python.org/2.7/library/functions.html#str)) – additional switch options
- **listenPort** ([int](https://docs.python.org/2.7/library/functions.html#int)) – custom listening port (optional)
- **inNamespace** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – override the switch spawn in namespace (optional)
- **of\_ver** ([int](https://docs.python.org/2.7/library/functions.html#int)) – override OpenFlow version (optional)
- **ip** – set IP address for the switch (optional)

**Returns** newly created Switch object

**Return type** `mininet.node.Switch`

**create\_Controller** (*name*, *controller=None*, *\*\*params*)

Create and add a new OF controller to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Controller class type.

**Warning:** Should not call this function and use the default InternalControllerProxy!

#### Parameters

- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) – name of controller
- **controller** (`mininet.node.Controller`) – custom controller class/constructor (optional)
- **inNamespace** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – override the controller spawn in namespace (optional)

**Returns** newly created Controller object

**Return type** `mininet.node.Controller`

**create\_SAP** (*name*, *cls=None*, *\*\*params*)

Create and add a new SAP to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Host class type.

#### Parameters

- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) – name of SAP
- **cls** (`mininet.node.Host`) – custom hosts class/constructor (optional)

**Returns** newly created Host object as the SAP

**Return type** `mininet.node.Host`

**bind\_inter\_domain\_SAPs** (*nffg*)

Search for inter-domain SAPs in given [NFFG](#), create them as a switch port and bind them to a physical interface given in sap.domain attribute.

**Parameters** **nffg** ([NFFG](#)) – topology description

**Returns** None

**\_ESCAPENetworkBuilder\_\_get\_new\_dpid()**

Generate a new DPID and return the valid format for Mininet/OVS.

**Returns** new DPID

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**\_ESCAPENetworkBuilder\_\_init\_from\_AbstractTopology (topo\_class)**

Build topology from pre-defined Topology class.

**Parameters** `topo_class` (*AbstractTopology*) – topology

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_CONFIG (format='NFFG')**

Build a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name TOPO.

**Parameters** `format` (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – NF-FG storing format (default: internal NFFG representation)

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_NFFG (nffg)**

Initialize topology from an *NFFG* representation.

**Parameters** `nffg` (*NFFG*) – topology object structure

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_file (path, format='NFFG')**

Build a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name TOPO.

**Parameters**

- `path` (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – file path
- `format` (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – NF-FG storing format (default: internal NFFG representation)

**Returns** None

**create\_Link (src, dst, src\_port=None, dst\_port=None, \*\*params)**

Create an undirected connection between src and dst.

Source and destination ports can be given optionally:

**Parameters**

- `src` – source Node
- `dst` – destination Node
- `src_port` – source Port (optional)
- `dst_port` – destination Port (optional)
- `params` – additional link parameters

**Returns**

**build (topo=None)**

Initialize network.

- 1.If the additional `topology` is given then using that for init.
- 2.If TOPO is not given, search topology description in CONFIG with the name ‘TOPO’.
- 3.If TOPO not found or an Exception was raised, search for the fallback topo with the name Fallback-TOPO.
- 4.If Fallback-TOPO not found raise an exception or run a bare Mininet object if the `run_dry` attribute is set

**Parameters** `topo` (*NFFG* or *AbstractTopology* or `None`) – optional topology representation

**Returns** object representing the emulated network

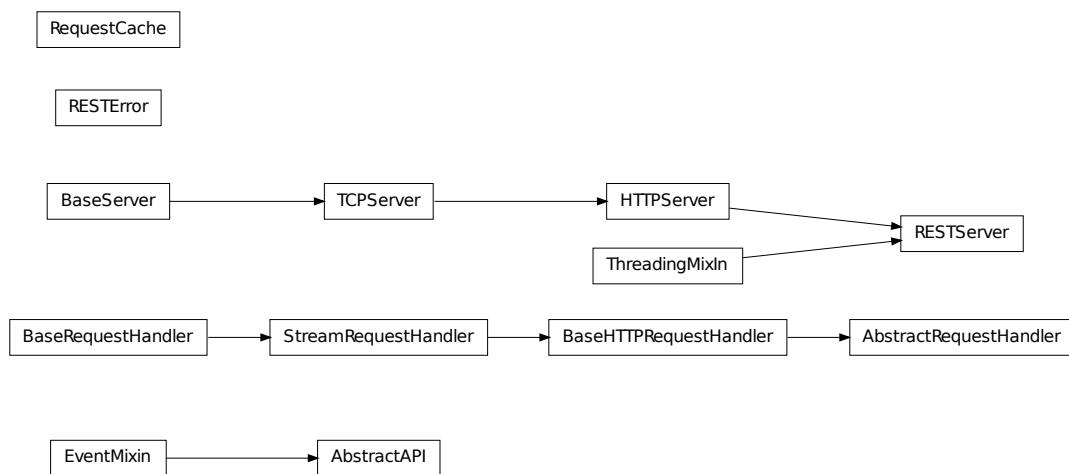
**Return type** `ESCAPENetworkBridge`

### `escape.util` package

Additional functions, classes, components

#### Submodules

`api.py` module Contains abstract classes for concrete layer API modules.



`AbstractAPI` contains the register mechanism into the POX core for layer APIs, the event handling/registering logic and defines the general functions for initialization and finalization steps.

`RESTServer` is a general HTTP server which parse HTTP request and forward to explicitly given request handler.

`RequestCache` stores HTTP request states.

`RESTError` can signal various error related to RESTful communication.

`AbstractRequestHandler` is a base class for concrete request handling. It implements the general URL and request body parsing functions.

**Module contents** Contains abstract classes for concrete layer API modules.

```
class escape.util.api.AbstractAPI(standalone=False, **kwargs)
    Bases: pox.lib.revent.revent.EventMixin
```

Abstract class for UNIFY's API.

Contain common functions.

Follows Facade design pattern -> simplified entry/exit point of the layers.

```
_core_name = 'AbstractAPI'
```

```
dependencies = ()
```

```
__init__(standalone=False, **kwargs)
```

Abstract class constructor.

Handle core registration along with `_all_dependencies_met()`.

Set given parameters (standalone parameter is mandatory) automatically as:

```
self._<param_name> = <param_value>
```

Base constructor functions have to be called as the last step in derived classes. Same situation with `_all_dependencies_met()` respectively. Must not override these function, just use `initialize()` for init steps. Actual API classes must only call `super()` (<https://docs.python.org/2.7/library/functions.html#super>) in their constructor with the form:

```
super(<API Class name>, self).__init__(standalone=standalone, **kwargs)
```

**Warning:** Do not use prefixes in the name of event handlers, because of automatic dependency discovery considers that as a dependent component and this situation cause a dead lock (component will be waiting to each other to set up)

**Parameters** `standalone` ([bool](https://docs.python.org/2.7/library/functions.html#bool))  
– started in standalone mode or not

`_all_dependencies_met()`

Called when every component on which depends are initialized on POX core.

Contain dependency relevant initialization.

**Returns** None

`initialize()`

Init function for child API classes to simplify dynamic initialization.

Called when every component on which depends are initialized and registered in POX core.

This function should be overwritten by child classes.

**Returns** None

`shutdown(event)`

Finalization, deallocation, etc. of actual component.

Should be overwritten by child classes.

**Parameters** `event` ([GoingDownEvent](#)) – shutdown event raised by POX core

**Returns** None

`static _read_json_from_file(graph_file)`

Read the given file and return a string formatted as JSON.

**Parameters** `graph_file` ([str](https://docs.python.org/2.7/library/functions.html#str)) –  
file path

**Returns** JSON data

**Return type** `str` ([str](https://docs.python.org/2.7/library/functions.html#str))

`__str__()`

Print class type and non-private attributes with their types for debugging.

**Returns** specific string

**Return type** `str` ([str](https://docs.python.org/2.7/library/functions.html#str))

`class escape.util.api.RequestCache`

Bases: `object` ([object](https://docs.python.org/2.7/library/functions.html#object))

Store HTTP request states.

`UNKNOWN = 'UNKNOWN'`

`INITIATED = 'INITIATED'`

`IN_PROGRESS = 'IN_PROGRESS'`

```

SUCCESS = 'SUCCESS'
ERROR = 'ERROR'

__init__()

add_request (id)
    Add a request to the cache.

    Parameters id (str or int) – request id

set_in_progress (id)
    Set the result of the request given by the id.

    Parameters id (str or int) – request id

set_result (id, result)
    Set the result of the request given by the id.

    Parameters
        • id (str or int) – request id
        • result (bool (https://docs.python.org/2.7/library/functions.html#bool)) – the result

get_result (id)

class escape.util.api.RESTServer (RequestHandlerClass, address='127.0.0.1', port=8008)
    Bases: SocketServer.ThreadingMixIn, BaseHTTPServer.HTTPServer
    (https://docs.python.org/2.7/library/basehttpserver.html#BaseHTTPServer.HTTPServer)
    Base HTTP server for RESTful API.

    Initiate an HTTPServer and run it in different thread.

    __init__ (RequestHandlerClass, address='127.0.0.1', port=8008)
        Set up an BaseHTTPServer.HTTPServer (https://docs.python.org/2.7/library/basehttpserver.html#BaseHTTPServer) in a different thread.

        Parameters
            • RequestHandlerClass (AbstractRequestHandler) – Class of a handler which handles HTTP request
            • address (str (https://docs.python.org/2.7/library/functions.html#str)) – Used IP address
            • port (int (https://docs.python.org/2.7/library/functions.html#int)) – Used port number

    start ()
        Start RESTServer thread.

    stop ()
        Stop RESTServer thread.

    run ()
        Handle one request at a time until shutdown.

exception escape.util.api.RESTError (msg=None, code=0)
    Bases: exceptions.Exception (https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception)
    Exception class for REST errors.

    __init__ (msg=None, code=0)
        msg
        code
        __str__()

```

```
class escape.util.api.AbstractRequestHandler(request, client_address, server)
```

Bases: `BaseHTTPServer.BaseHTTPRequestHandler` (<https://docs.python.org/2.7/library/basehttpserver.html#BaseHTTPServer.BaseHTTPRequestHandler>)

Minimalistic RESTful API for Layer APIs.

Handle /escape/\* URLs.

Method calling permissions represented in escape\_intf dictionary.

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually - use relevant helper function of core object: `callLater()` / `raiseLater()` or use `schedule_as_coop_task()` decorator defined in `escape.util.misc` on the called function

```
server_version = 'ESCAPE/2.0.0'
```

```
static_prefix = 'escape'
```

```
request_perm = {'POST': ('ping',), 'GET': ('ping', 'version', 'operations')}
```

```
bounded_layer = None
```

```
rpc_mapper = None
```

```
log = <logging.Logger object at 0x48a6f50>
```

```
do_GET()
```

Get information about an entity. R for CRUD convention.

```
do_POST()
```

Create an entity. C for CRUD convention.

```
do_PUT()
```

Update an entity. U for CRUD convention.

```
do_DELETE()
```

Delete an entity. D for CRUD convention.

```
do_OPTIONS()
```

Handling unsupported HTTP verbs.

**Returns** None

```
do_HEAD()
```

Handling unsupported HTTP verbs.

**Returns** None

```
do_TRACE()
```

Handling unsupported HTTP verbs.

**Returns** None

```
do_CONNECT()
```

Handling unsupported HTTP verbs.

**Returns** None

```
_process_url()
```

Split HTTP path and call the carved function if it is defined in this class and in request\_perm.

**Returns** None

```
_get_body()
```

Parse HTTP request body as a plain text.

---

**Note:** Call only once by HTTP request.

---

**Note:** Parsed JSON object is Unicode.

---

GET, DELETE messages don't have body - return empty dict by default.

**Returns** request body in str format

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

#### `send_REST_headers()`

Set the allowed REST verbs as an HTTP header (Allow).

**Returns** None

#### `send_acknowledge(msg='{"result": "Accepted"}')`

Send back acknowledge message.

##### Parameters

- **msg** – response body
- **msg** – dict

**Returns** None

#### `_send_json_response(data, encoding='utf-8')`

Send requested data in JSON format.

##### Parameters

- **data** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – data in JSON format
- **encoding** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – Set data encoding (optional)

**Returns** None

#### `error_content_type = 'text/json'`

#### `send_error(code, message=None)`

Override original function to send back allowed HTTP verbs and set format to JSON.

#### `log_error(mformat, *args)`

Overwritten to use POX logging mechanism.

#### `log_message(mformat, *args)`

Disable logging of incoming messages.

#### `log_full_message(mformat, *args)`

Overwritten to use POX logging mechanism.

#### `_proceed_API_call(function, *args, **kwargs)`

Fail-safe method to call API function.

The cooperative micro-task context is handled by actual APIs.

Should call this with params, not directly the function of actual API.

##### Parameters

- **function** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – function name
- **args** (`tuple` (<https://docs.python.org/2.7/library/functions.html#tuple>)) – Optional params
- **kwargs** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – Optional named params

**Returns** None

**ping()**  
For testing REST API aliveness and reachability.

**version()**  
Return with version  
**Returns** None

**operations()**  
Return with allowed operations  
**Returns** None

**config.py module** Contains manager and handling functions for global ESCAPE configuration.

### ESCAPEConfig

*ESCAPEConfig* is a wrapper class for CONFIG.

**Module contents** Contains manager and handling functions for global ESCAPE configuration.

**class escape.util.config.ESCAPEConfig (default=None)**  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Wrapper class for configuration to hide specialties with respect to storing, loading, parsing and getting special data.

Contains functions for config handling and manipulation.

Should be instantiated once!

**\_\_metaclass\_\_**  
alias of `Singleton`

**LAYERS** = ('service', 'orchestration', 'adaptation', 'infrastructure')

**DEFAULT\_CFG** = 'additional-config-file'

**\_\_init\_\_ (default=None)**  
Init configuration from given data or an empty dict.

**Parameters** `default` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – default configuration

**in\_initiated**

**add\_cfg (cfg)**  
Override configuration.

**Parameters** `cfg` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – new configuration

**Returns** None

**load\_config (config=None)**  
Load static configuration from file if it exist or leave the default intact.

---

**Note:** The CONFIG is updated per data under the Layer entries. This means that the minimal amount

of data have to given is the hole sequence or collection under the appropriate key. E.g. the hole data under the ‘STRATEGY’ key in ‘orchestration’ layer.

---

**Parameters** `config` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)) – config file name relative to pox.py (optional)

**Returns** self

**Return type** `ESCAPEConfig`

**dump()**

Return with the entire configuration in JSON.

**Returns** config

**Return type** `str` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)

**is\_layer\_loaded(layer)**

Return the value given UNIFY’s layer is loaded or not.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** layer condition

**Return type** `bool` ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool)

**set\_layer\_loaded(layer)**

Set the given layer LOADED value.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** None

**\_\_getitem\_\_(item)**

Can be used the config as a dictionary: CONFIG[...]

**Parameters** `item` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)) – layer key

**Returns** layer config

**Return type** `dict` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) (https://docs.python.org/2.7/library/stdtypes.html#dict)

**\_\_setitem\_\_(key, value)**

Disable explicit layer config modification.

**\_\_delitem\_\_(key)**

Disable explicit layer config deletion.

**static get\_project\_root\_dir()**

Return the absolute path of project dir

**Returns** path of project dir

**Return type** `str` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)

**get\_mapping\_enabled(layer)**

Return the mapping process is enabled for the layer or not.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** enabled value (default: True)

**Return type** `bool` ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool)

**get\_strategy(layer)**

Return with the Strategy class of the given layer.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** Strategy class

**Return type** `AbstractMappingStrategy`

**get\_mapper** (`layer`)

Return with the Mapper class of the given layer.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** Mapper class

**Return type** `AbstractMapper`

**get\_mapping\_processor** (`layer`)

Return with Validator class of the given layer.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** Validator class

**Return type** `AbstractMappingDataProcessor`

**get\_processor\_enabled** (`layer`)

Return the mapping process is enabled for the `layer` or not.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** enabled value (default: True)

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_threaded** (`layer`)

Return with the value if the mapping strategy is needed to run in separated thread or not. If value is not defined: return False.

**Parameters** `layer` ([str](https://docs.python.org/2.7/library/functions.html#str)) – layer name

**Returns** threading value

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_component** (`component`)

Return with the class of the adaptation component.

**Parameters** `component` ([str](https://docs.python.org/2.7/library/functions.html#str)) – component name

**Returns** component class

**get\_component\_params** (`component`)

Return with the initial parameters of the given component defined in CONFIG. The param's name must be identical with the attribute name of the component constructor.

**Parameters** `component` ([str](https://docs.python.org/2.7/library/functions.html#str)) – component name

**Returns** initial params

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_managers** ()

Return the default DomainManagers for initialization on start.

**Returns** list of `AbstractDomainManager`

**Return type** `list` (<https://docs.python.org/2.7/library/functions.html#list>)

**reset\_domains\_after\_shutdown()**  
Return with the shutdown strategy to reset domain or not.

**get\_mn\_network\_opts()**  
Return the optional Mininet parameters for initiation.

**Returns** optional constructor params (default: empty dict)

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_mininet\_topology()**  
Return the Mininet topology class.

**Returns** topo class

**get\_fallback\_topology()**  
Return the fallback topology class.

**Returns** fallback topo class

**Return type** `:any:`AbstractTopology``

**get\_sdn\_topology()**  
Return the path of the SDN topology config file.

**Returns** topo class

**get\_clean\_after\_shutdown()**  
Return with the value if a cleaning process need to be done or not.

**Returns** cleanup (default: False)

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_ros\_agent\_class()**  
Return with the request handler class of Agent REST API.

**Returns** agent class

**Return type** `AbstractRequestHandler`

**get\_ros\_agent\_prefix()**  
Return the REST API prefix for agent request handler.

**Returns** prefix

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**get\_ros\_agent\_address()**  
Return the REST API (address, port) for agent REST server.

**Returns** address and port

**Return type** `tuple` (<https://docs.python.org/2.7/library/functions.html#tuple>)

**get\_sas\_api\_class()**  
Return with the request handler class of Service Layer REST API.

**Returns** REST API class

**Return type** `AbstractRequestHandler`

**get\_sas\_api\_prefix()**  
Return the REST API prefix for Service Layer request handler.

**Returns** prefix

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**get\_sas\_api\_address()**  
Return the REST API (address, port) for Service Layer REST server.

**Returns** address and port

**Return type** tuple (<https://docs.python.org/2.7/library/functions.html#tuple>)

**get\_cfor\_api\_class()**

Return with the request handler class of Cf-Or REST API.

**Returns** REST API class

**Return type** *AbstractRequestHandler*

**get\_cfor\_api\_prefix()**

Return the REST API prefix for Cf-Or request handler.

**Returns** prefix

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**get\_cfor\_api\_address()**

Return the REST API (address, port) for Cf-Or REST server.

**Returns** address and port

**Return type** tuple (<https://docs.python.org/2.7/library/functions.html#tuple>)

**get\_api\_virtualizer(layer\_name, api\_name)**

Return the type of the assigned Virtualizer.

**Parameters** *api\_name* (str (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the REST-API in the global config.

**Returns** type of the Virtualizer as in *VirtualizerManager.TYPES*

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**get\_adapter\_keepalive(adapter)**

Return the value if the keepalive functionality (periodic OF Echo request) is need to be initiated or not.

**Returns** keepalive

**Return type** bool (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_SAP\_xterms()**

Return the value if need to initiate xtemrs assigned to SAPs.

**Returns** xterms

**Return type** bool (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_Controller\_params()**

Return the additional parameter which are forwarded to the constructor of the specific *InternalControllerProxy* class during Mininet building.

**Returns** additional parameters as a dict (default: empty dict)

**Return type** dict (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_EE\_params()**

Return the additional parameter which are forwarded to the constructor of the `mininet.node.EE` class during Mininet building.

**Returns** additional parameters as a dict (default: empty dict)

**Return type** dict (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_Switch\_params()**

Return the additional parameter which are forwarded to the constructor of the specific `mininet.node.Switch` class during Mininet building.

**Returns** additional parameters as a dict (default: empty dict)

**Return type** dict (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_SAP\_params()**  
 Return the additional parameter which are forwarded to the constructor of the mininet.node.Host class during Mininet building.

**Returns** additional parameters as a dict (default: empty dict)

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**get\_Link\_params()**  
 Return the additional parameter which are forwarded to the constructor of the mininet.node.Link class during Mininet building.

**Returns** additional parameters as a dict (default: empty dict)

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**\_ESCAPEConfig\_parse\_part(inner\_part, loaded\_part)**

Inner function to parse and check a part of configuration and update the stored one according the detected changes. Uses recursion.

#### Parameters

- **inner\_part** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – part of inner representation of config (CONFIG)
- **loaded\_part** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – part of loaded configuration (escape.config)

**Returns** original config is changed or not.

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**conversion.py module** Contains helper classes for conversion between different NF-FG representations.

NFFGConverter

```

classDiagram
    class AbstractNFFG
    class Virtualizer3BasedNFFGBuilder
    AbstractNFFG --> Virtualizer3BasedNFFGBuilder
  
```

*Virtualizer3BasedNFFGBuilder* contains the common function for an NFFG representation based on virtualizer3.py.

*NFFGConverter* contains conversation logic for different *NFFG* representations.

**Module contents** Contains helper classes for conversion between different NF-FG representations.

**class escape.util.conversion.Virtualizer3BasedNFFGBuilder**  
*Bases:* `escape.util.nffg.AbstractNFFG`

Builder class for construct an NFFG in XML format rely on ETH's nffglib.py.

**Note:** Only tailored to the current virtualizer3.py (2015.08.14) and OpenStack domain. Should not use for general purposes, major part could be unimplemented!

**DEFAULT\_INFRA\_TYPE** = ‘BisBis’

**DEFAULT\_NODE\_TYPE** = ‘0’

**PORT\_ABSTRACT** = ‘port-abstract’

**PORT\_SAP** = ‘port-sap’

**\_\_init\_\_()**

Init. Create an empty virtualizer container and the necessary sub-objects.

**Returns** None

**dump()**

Return the constructed NFFG as a string in XML format.

**Returns** NFFG in XML format

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**\_\_str\_\_()**

Dump the constructed NFFG as a pretty string.

**Returns** NFFG in XML format

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**build()**

Return the constructed XML object a.k.a. the Virtualizer.

**Returns** NFFG

**Return type** Virtualizer

**classmethod parse** (*data*)

Parse the given XML-formatted string and return the constructed Virtualizer.

**Parameters** **data** (str (<https://docs.python.org/2.7/library/functions.html#str>)) – raw text formatted in XML

**Returns** parsed XML object-structure

**Return type** Virtualizer

**id**

Return the id of the NFFG.

**Returns** id

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**name**

Return the name of NFFG.

**Returns** name

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**nodes**

Return the list of nodes.

**Returns** nodes

**Return type** list([InfraNodeGroup](#)) ### RETURN DICT {nodeID:[InfraNodeGroup](#)}

**links**

Return the list of links. If links is not exist, create the empty container on the fly.

**Returns** links

**Return type** list(Links) ### RETURN DICT {(src,dst):Link}

**add\_edge** (src, dst, link)

**add\_node** (parent, id=None, name=None, type=None)

Add an empty node(NodeGroup) to its parent. If the parameters are not given, they are generated from default names and the actual container's size.

#### Parameters

- **parent** (*InfraNodeGroup or NodeGroup or NFInstances or SupportedNFs*) – container of the new node
- **id** (*str or int*) – ID of node
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name (optional)
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – node type (default: 0)

**Returns** node object

**Return type** NodeGroup

**add\_infrastructure\_node** (id=None, name=None, type=None)

Add an infrastructure node to NFFG (as a BiS-BiS), which is a special node directly under the Virtualizer main container object.

#### Parameters

- **id** (*str or int*) – ID of infrastructure node
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name (optional)
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – node type (default: BisBis)

**Returns** infrastructure node object

**Return type** InfraNodeGroup

**add\_node\_port** (parent, type='port-abstract', id=None, name=None, param=None)

Add a port to a Node. The parent node could be the nodes which can has ports i.e. a special infrastructure node, initiated and supported NF objects. If the type is a SAP type, the param attribute is read as the sap-type. If the param attribute starts with “vxlan:” then the sap-type will be set to “vxlan” and the vxlan tag will be set to the number after the colon.

#### Parameters

- **parent** (*InfraNodeGroup or NodeGroup*) – parent node
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – type of the port
- **id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – port ID (optional)
- **name** (*str (optional)*) – port name (optional)
- **param** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – additional parameters: abstract: capability; sap: sap-type

**Returns** port object

**Return type** PortGroup

**add\_node\_resource** (parent, cpu=None, mem=None, storage=None)

Add software resources to a Node or an infrastructure Node.

#### Parameters

- **parent** (*InfraNodeGroup or NodeGroup*) – parent node
- **cpu** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – In virtual CPU (vCPU) units

- **mem** ([str](https://docs.python.org/2.7/library/functions.html#str)) – Memory with units, e.g., 1Gbyte
- **storage** ([str](https://docs.python.org/2.7/library/functions.html#str)) – Storage with units, e.g., 10Gbyte

**Returns** resource object

**Return type** NodeResources

**add\_link\_resource** (*parent*, *delay=None*, *bandwidth=None*)

Add link resources to a connection.

**Parameters**

- **parent** (*Flowentry or Link*) – container of the connection
- **delay** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – delay value with unit; e.g. 5ms (optional)
- **bandwidth** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – bandwidth value with unit; e.g. 10Mbps (optional)

**Returns** connection resource

**Return type** LinkResource

**add\_nf\_instance** (*parent*, *id=None*, *name=None*, *type=None*)

Add an NF instance to an Infrastructure Node.

**Parameters**

- **parent** (*InfraNodeGroup*) – container of the new node
- **id** (*str or int*) – ID of node
- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – name (optional)
- **type** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – node type (default: 0)

**Returns** NF instance object

**Return type** NodeGroup

**add\_supported\_nf** (*parent*, *id=None*, *name=None*, *type=None*)

Add a supported NF to an Infrastructure Node.

**Parameters**

- **parent** (*InfraNodeGroup*) – container of the new node
- **id** (*str or int*) – ID of node
- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – name (optional)
- **type** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – node type (default: 0)

**Returns** supported NF object

**Return type** NodeGroup

**add\_flow\_entry** (*parent*, *in\_port*, *out\_port*, *match=None*, *action=None*, *delay=None*, *bandwidth=None*)

Add a flowentry to an Infrastructure Node.

**Parameters**

- **parent** (*InfraNodeGroup*) – container of the flowtable
- **in\_port** (*PortGroup*) – related in port object
- **match** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – matching rule

- **in\_port** – related out port object
- **action** (*list or tuple or str*) – forwarding actions
- **delay** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – delay value with unit; e.g. 5ms (optional)
- **bandwidth** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – bandwidth value with unit; e.g. 10Mbps (optional)

**Returns** flowentry

**Return type** FlowEntry

**add\_inter\_infra\_link** (*src, dst, \*\*kwargs*)

Add link between Infrastructure nodes a.k.a define link in Virtualizer.

**Parameters**

- **src** (*PortGroup*) – source port
- **dst** (*PortGroup*) – destination port

**Returns** link object

**Return type** LinksGroup

**add\_nf()**

Add a Network Function Node.

**add\_sap()**

**add\_infra** (*id=None, name=None, type=None*)

Add an Infrastructure Node.

**add\_link** (*src, dst*)

**add\_smlink** (*src, dst*)

**add\_req** (*src, dst*)

**del\_node** (*node*)

**del\_edge** (*src, dst*)

**\_Virtualizer3BasedNFFGBuilder\_\_UUID\_NUM = 0**

**\_Virtualizer3BasedNFFGBuilder\_\_add\_connection** (*parent, src, dst, id=None, name=None, delay=None, bandwidth=None*)

Add a connection a.k.a a <link> to the Virtualizer or to a Node.

**Parameters**

- **parent** (*Virtualizer or NodeGroup*) – parent node
- **src** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – relative path to the source port
- **dst** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – relative path to the destination port
- **id** (*str or int*) – link ID (optional)
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – link name (optional)
- **delay** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – delay value with unit; e.g. 5ms (optional)
- **bandwidth** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – bandwidth value with unit; e.g. 10Mbps (optional)

**Returns** link object

**Return type** LinksGroup

```
escape.util.conversion.test_xml_based_builder()
escape.util.conversion.test_virtualizer3_based_builder()
escape.util.conversion.test_topo_un()
escape.util.conversion.test_topo_os()
```

**class** escape.util.conversion.NFFGConverter (*domain, logger=None*)  
Bases: [object](#) (<https://docs.python.org/2.7/library/functions.html#object>)

Convert different representation of NFFG in both ways.

**\_\_init\_\_** (*domain, logger=None*)

**parse\_from\_Virtualizer3** (*xml\_data*)

Convert Virtualizer3-based XML str → NFFGModel based NFFG object

**Parameters** **xml\_data** – XML plain data formatted with Virtualizer

**Type** xml\_data: str

**Returns** created NF-FG

**Return type** [NFFG](#)

**static unescape\_output\_hack** (*data*)

**adapt\_mapping\_into\_Virtualizer** (*virtualizer, nffg*)

Install NFFG part or complete NFFG into given Virtualizer.

**Parameters**

- **virtualizer** – Virtualizer object based on ETH’s XML/Yang version.
- **nffg** – splitted NFFG (not necessarily in valid syntax)

**Returns** modified Virtualizer object

**\_NFFGConverter\_\_convert\_flowrule\_action** (*domain, action*)

Convert Flowrule action field from NFFG format to Virtualizer according to domain.

**Parameters**

- **domain** – domain name
- **action** – flowrule action field

**Returns** converted data

**\_NFFGConverter\_\_convert\_flowrule\_match** (*domain, match*)

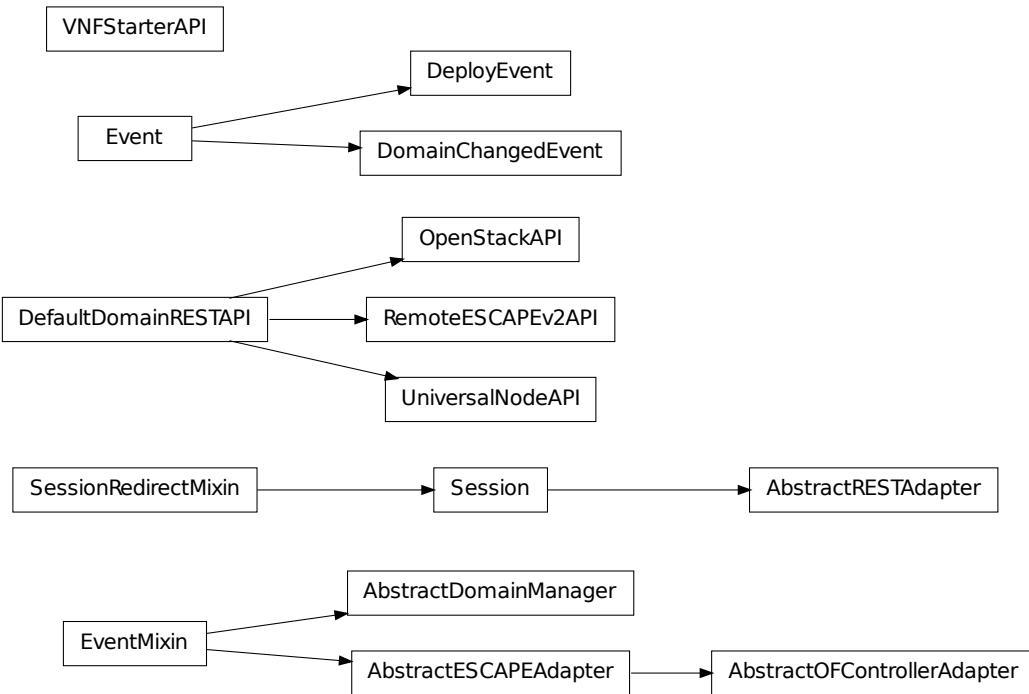
Convert Flowrule match field from NFFG format to Virtualizer according to domain.

**Parameters**

- **domain** – domain name
- **match** – flowrule match field

**Returns** converted data

**domain.py module** Implement the supporting classes for domain adapters.



*DomainChangedEvent* signals changes for *ControllerAdapter* in an unified way.

*DeployEvent* can send NFFG to Infrastructure layer for deploying.

*AbstractDomainManager* contains general logic for top domain managers.

*AbstractESCAPEAdapter* contains general logic for actual Adapters.

*AbstractOFControllerAdapter* contains general logic for actual OF controller based Adapters.

*DefaultDomainRESTAPI* defines unified interface for domain's REST-API.

*VNFStarterAPI* defines the interface for VNF management based on VNFStarter YANG description.

*OpenStackAPI* defines the interface for communication with OpenStack domain.

*UniversalNodeAPI* defines the interface for communication with Universal Node domain.

*RemoteESCAPEv2API* defines the interface for communication with a remote ESCAPE instance started in agent mode.

*AbstractRESTAdapter* contains the general functions for communication through an HTTP/RESTful API.

Requirements:

```
sudo pip install requests
```

**Module contents** Implement the supporting classes for domain adapters.

```
class escape.util.domain.DomainChangedEvent(domain, cause, data=None)
Bases: pox.lib.revent.revent.Event
```

Event class for signaling all kind of change(s) in specific domain.

This event's purpose is to hide the domain specific operations and give a general and unified way to signal domain changes to ControllerAdapter in order to handle all the changes in the same function/algoritm.

**TYPE**

alias of enum

**\_\_init\_\_(domain, cause, data=None)**

Init event object

**Parameters**

- **domain** ([str](https://docs.python.org/2.7/library/functions.html#str)) – domain name.  
Should be `AbstractESCAPEAdapter.name`
- **cause** ([str](https://docs.python.org/2.7/library/functions.html#str)) – type of the domain change: `DomainChangedEvent.TYPE`
- **data** ([NFFG](#) or str) – data connected to the change (optional)

**Returns** None**class escape.util.domain.DeployEvent(nffg\_part)**

Bases: `pox.lib.revent.revent.Event`

Event class for signaling NF-FG deployment to infrastructure layer API.

Used by DirectMininetAdapter for internal NF-FG deployment.

**\_\_init\_\_(nffg\_part)****class escape.util.domain.AbstractDomainManager(\*\*kwargs)**

Bases: `pox.lib.revent.revent.EventMixin`

Abstract class for different domain managers. DomainManagers is top level classes to handle and manage domains transparently.

Follows the MixIn design pattern approach to support general manager functionality for topmost Controller-Adapter class.

Follows the Component Configurator design pattern as base component class.

**\_eventMixin\_events = set([<class ‘escape.util.domain.DomainChangedEvent’>])**

**name = ‘UNDEFINED’**

**POLL\_INTERVAL = 3**

**\_\_init\_\_(\*\*kwargs)**

Init.

**init(configurator, \*\*kwargs)**

Abstract function for component initialization.

**Parameters**

- **configurator** (`ComponentConfigurator`) – component configurator for configuring adapters
- **kwargs** ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – optional parameters

**Returns** None**run()**

Abstract function for starting component.

**Returns** None**finit()**

Abstract function for starting component.

**suspend()**

Abstract class for suspending a running component.

---

**Note:** Not used currently!

---

**Returns** None

**resume()**

Abstract function for resuming a suspended component.

---

**Note:** Not used currently!

**Returns** None

**info()**

Abstract function for requesting information about the component.

---

**Note:** Not used currently!

**Returns** None

**start\_polling(wait=1)**

Initialize and start a Timer co-op task for polling.

**Parameters** **wait** ([int](https://docs.python.org/2.7/library/functions.html#int)) – polling period (default: 1)

**restart\_polling(wait=3)**

Reinitialize and start a Timer co-op task for polling.

**Parameters** **wait** ([int](https://docs.python.org/2.7/library/functions.html#int)) – polling period (default: 3)

**stop\_polling()**

Stop timer.

**poll()**

Poll the defined domain agent. Handle different connection errors and go to slow/rapid poll. When an agent is (re)detected update the current resource information.

**\_detect\_topology()**

Check the undetected topology is up or not.

**Returns** detected or not

**Return type** [bool](https://docs.python.org/2.7/library/functions.html#bool) (<https://docs.python.org/2.7/library/functions.html#bool>)

**update\_local\_resource\_info(data=None)**

Update the resource information of this domain with the requested configuration.

**Returns** None

**install\_nffg(nffg\_part)**

Install an [NFFG](#) related to the specific domain.

**Parameters** **nffg\_part** ([NFFG](#)) – NF-FG need to be deployed

**Returns** None

**clear\_domain()**

Clear the Domain according to the first received config.

**class escape.util.domain.[AbstractESCAPEAdapter](#)**

Bases: `pox.lib.revent.revent.EventMixin`

Abstract class for different domain adapters.

Domain adapters can handle domains as a whole or well-separated parts of a domain e.g. control part of an SDN network, infrastructure containers or other entities through a specific protocol (NETCONF, HTTP/REST).

Follows the Adapter design pattern (Adaptor base class).

Follows the MixIn design pattern approach to support general adapter functionality for manager classes mostly.

```
_eventMixin_events = set([<class 'escape.util.domain.DomainChangedEvent'>])
name = None

__init__()
    Init.

start_polling(wait=1)
    Initialize and start a Timer co-op task for polling.

    Parameters wait (int (https://docs.python.org/2.7/library/functions.html#int)) – polling
        period (default: 1)

stop_polling()
    Stop timer.

poll()
    Template function to poll domain state. Called by a Timer co-op multitask. If the function return with
    False the timer will be cancelled.

check_domain_reachable()
    Checker function for domain polling.

    Returns the domain is detected or not

    Return type bool (https://docs.python.org/2.7/library/functions.html#bool)

get_topology_resource()
    Return with the topology description as an NFFG.

    Returns the emulated topology description

    Return type NFFG

class escape.util.domain.AbstractOFControllerAdapter(name=None, address='127.0.0.1', port=6653, keepalive=False)
    ad-
    dress='127.0.0.1', port=6653,
    keepalive=False)

Bases: escape.util.domain.AbstractESCAPEAdapter

Abstract class for different domain adapters which need SDN/OF controller capability.

__interval = 20
__switch_timeout = 5
infra_to_dpid = {}
saps = {}

__init__(name=None, address='127.0.0.1', port=6653, keepalive=False)
    Initialize attributes, register specific connection Arbiter if needed and set up listening of OpenFlow
    events.

    Parameters
        • name (str (https://docs.python.org/2.7/library/functions.html#str)) – name used to
            register component into pox.core
        • address (str (https://docs.python.org/2.7/library/functions.html#str)) – socket ad-
            dress (default: 127.0.0.1)
```

- **port** ([int](https://docs.python.org/2.7/library/functions.html#int)) – socket port (default: 6633)

**classmethod \_handle\_keepalive\_handler (ofnexus)**

**filter\_connections (event)**

Handle which connection should be handled by this Adapter class.

This adapter accept every OpenFlow connection by default.

**Parameters** **event** (`pox.openflow.ConnectionUp`) – POX internal ConnectionUp event (event.dpid, event.connection)

**Returns** True or False obviously

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource ()**

**check\_domain\_reachable ()**

**delete\_flowrules (id)**

Delete all flowrules from the first (default) table of an OpenFlow switch.

**Parameters** **id** ([str](https://docs.python.org/2.7/library/functions.html#str)) – ID of the infra element stored in the NFFG

**Returns** None

**install\_flowrule (id, match, action)**

Install a flowrule in an OpenFlow switch.

**Parameters**

- **id** ([str](https://docs.python.org/2.7/library/functions.html#str)) – ID of the infra element stored in the NFFG
- **match** ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – match part of the rule (keys: in\_port, vlan\_id)
- **action** ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – action part of the rule (keys: out, vlan\_push, vlan\_pop)

**Returns** None

**class escape.util.domain.VNFStarterAPI**

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Define interface for managing VNFs.

**See also:**

`vnf_starter.yang`

Follows the MixIn design pattern approach to support VNFStarter functionality.

**VNF\_HEADER\_COMP = ‘headerCompressor’**

**VNF\_HEADER\_DECOMP = ‘headerDecompressor’**

**VNF\_FORWARDER = ‘simpleForwarder’**

**class VNFStatus**

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Helper class for define VNF status code constants.

From YANG: Enum for indicating statuses.

**FAILED = -1**

**s\_FAILED = ‘FAILED’**

**INITIALIZING = 0**

```
s_INITIALIZING = 'INITIALIZING'
UP_AND_RUNNING = 1
s_UP_AND_RUNNING = 'UP_AND_RUNNING'

class VNFStarterAPI.ConnectedStatus
    Bases: object (https://docs.python.org/2.7/library/functions.html#object)
        Helper class for define VNF connection code constants.
        From YANG: Connection status.

    DISCONNECTED = 0
    s_DISCONNECTED = 'DISCONNECTED'
    CONNECTED = 1
    s_CONNECTED = 'CONNECTED'

VNFStarterAPI.__init__()

VNFStarterAPI.initiateVNF (vnf_type, vnf_description=None, options=None)
    Initiate/define a VNF.

Parameters

- vnf_type (str (https://docs.python.org/2.7/library/functions.html#str)) – pre-defined VNF type (see in vnf_starter/available_vnfs)
- vnf_description (str (https://docs.python.org/2.7/library/functions.html#str)) – Click description if there are no pre-defined type
- options (collections.OrderedDict (https://docs.python.org/2.7/library/collections.html#collections.OrderedDict)) – unlimited list of additional options as name-value pairs

Returns parsed RPC response

Return type dict (https://docs.python.org/2.7/library/stdtypes.html#dict)

VNFStarterAPI.connectVNF (vnf_id, vnf_port, switch_id)
    Connect a VNF to a switch.

Parameters

- vnf_id (str (https://docs.python.org/2.7/library/functions.html#str)) – VNF ID (mandatory)
- vnf_port (str (https://docs.python.org/2.7/library/functions.html#str)) – VNF port (mandatory)
- switch_id (str (https://docs.python.org/2.7/library/functions.html#str)) – switch ID (mandatory)

Returns Returns the connected port(s) with the corresponding switch(es).

Return type dict (https://docs.python.org/2.7/library/stdtypes.html#dict)

VNFStarterAPI.disconnectVNF (vnf_id, vnf_port)
    Disconnect VNF from a switch.

Parameters

- vnf_id (str (https://docs.python.org/2.7/library/functions.html#str)) – VNF ID (mandatory)
- vnf_port (str (https://docs.python.org/2.7/library/functions.html#str)) – VNF port (mandatory)

Returns reply data

Return type dict (https://docs.python.org/2.7/library/stdtypes.html#dict)
```

`VNFStarterAPI.startVNF(vnf_id)`

Start VNF.

**Parameters** `vnf_id` ([str](https://docs.python.org/2.7/library/functions.html#str)) – VNF ID (mandatory)

**Returns** reply data

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

`VNFStarterAPI.stopVNF(vnf_id)`

Stop VNF.

**Parameters** `vnf_id` ([str](https://docs.python.org/2.7/library/functions.html#str)) – VNF ID (mandatory)

**Returns** reply data

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

`VNFStarterAPI.getVNFInfo(vnf_id=None)`

Request info from available VNF instances.

**Parameters** `vnf_id` ([str](https://docs.python.org/2.7/library/functions.html#str)) – particular VNF id (default: list info about all VNF)

**Returns** parsed RPC reply

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

`class escape.util.domain.DefaultDomainRESTAPI`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Define unified interface for managing UNIFY domains with REST-API.

Follows the MixIn design pattern approach to support OpenStack functionality.

`get_config()`

Queries the infrastructure view with a netconf-like “get-config” command.

**Returns** infrastructure view

**Return type** `:any::NFFG`

`edit_config(data)`

Send the requested configuration with a netconf-like “edit-config” command.

**Parameters** `data` (`:any::NFFG`) – whole domain view

**Returns** status code

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

`ping()`

Call the ping RPC.

**Returns** response text (should be: ‘OK’)

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

`class escape.util.domain.OpenStackAPI`

Bases: `escape.util.domain.DefaultDomainRESTAPI`

Define interface for managing OpenStack domain.

---

**Note:** Fitted to the API of ETH REST-like server which rely on virtualizer3!

---

Follows the MixIn design pattern approach to support OpenStack functionality.

```
class escape.util.domain.UniversalNodeAPI
    Bases: escape.util.domain.DefaultDomainRESTAPI
```

Define interface for managing Universal Node domain.

---

**Note:** Fitted to the API of ETH REST-like server which rely on virtualizer3!

---

Follows the MixIn design pattern approach to support UN functionality.

```
class escape.util.domain.RemoteESCAPEv2API
```

**Bases:** escape.util.domain.DefaultDomainRESTAPI

Define interface for managing remote ESCAPEv2 domain.

Follows the MixIn design pattern approach to support remote ESCAPEv2 functionality.

```
class escape.util.domain.AbstractRESTAdapter (base_url, auth=None)
```

**Bases:** requests.sessions.Session

Abstract class for various adapters rely on a RESTful API. Contains basic functions for managing HTTP connections.

Based on :any:`Session` class.

Follows Adapter design pattern.

```
custom_headers = {'User-Agent': 'ESCAPE/2.0.0'}
```

```
CONNECTION_TIMEOUT = 5
```

```
GET = 'GET'
```

```
POST = 'POST'
```

```
__init__ (base_url, auth=None)
```

#### URL

```
send_request (method, url=None, body=None, **kwargs)
```

Prepare the request and send it. If valid URL is given that value will be used else it will be append to the end of the `base_url`. If `url` is not given only the `base_url` will be used.

#### Parameters

- **method** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – HTTP method
- **url** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – valid URL or relevant part follows `self.base_url`
- **body** (`NFG` or dict or bytes or str) – request body

#### Returns

raw response data

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

```
_AbstractRESTAdapter__suppress_requests_logging (level=None)
```

Suppress annoying and detailed logging of `requests` and `urllib3` packages.

**Parameters** **level** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – level of logging (default: WARNING)

#### Returns

```
send_no_error (method, url=None, body=None, **kwargs)
```

Send REST request with handling exceptions.

#### Parameters

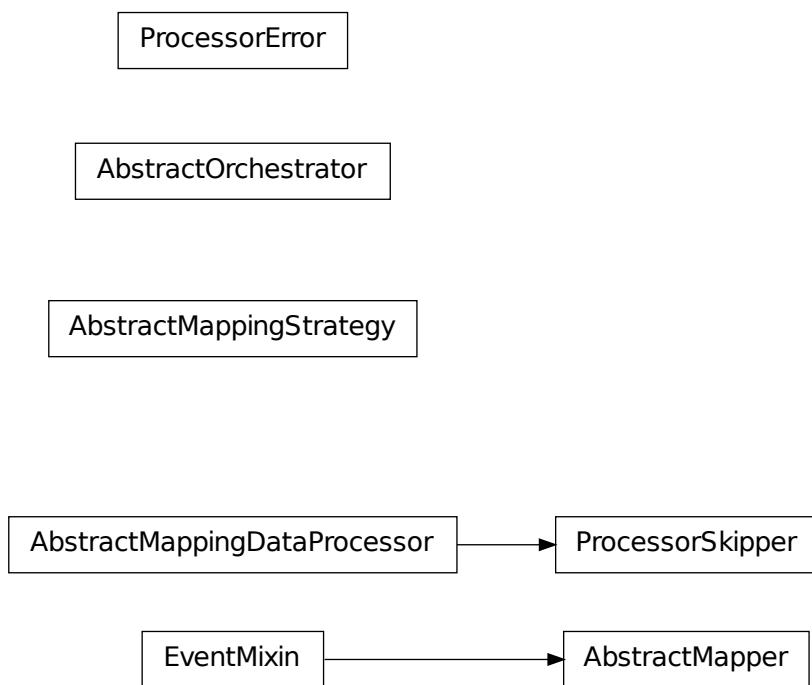
- **method** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – HTTP method

- **url** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – valid URL or relevant part follows `self.base_url`
- **body** ([NFFG](#) or dict or bytes or str) – request body

**Returns** raw response data

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**mapping.py module** Contains abstract classes for NFFG mapping.



*AbstractMapper* is an abstract class for orchestration method which should implement mapping preparations and invoke actual mapping algorithm.

*AbstractMappingStrategy* is an abstract class for containing entirely the mapping algorithm as a class method.

*AbstractOrchestrator* implements the common functionality for orchestrator's in different layers.

*ProcessorError* can signal unfulfilled requirements.

*AbstractMappingDataProcessor* is an abstract class to implement pre and post processing functions right before/after the mapping.

*ProcessorSkipper* implements a non-processing class to skip pre/post processing gracefully.

**Module contents** Contains abstract classes for NFFG mapping.

**class escape.util.mapping.*AbstractMappingStrategy***  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for the mapping strategies.

Follows the Strategy design pattern.

**\_\_init\_\_()**  
Init

**classmethod map (graph, resource)**  
Abstract function for mapping algorithm.

**Warning:** Derived class have to override this function

#### Parameters

- **graph** ([NFFG](#)) – Input graph which need to be mapped
- **resource** ([NFFG](#)) – resource info

**Raise** `NotImplementedError`

**Returns** mapped graph

**Return type** [NFFG](#)

**exception escape.util.mapping.ProcessorError**

Bases: `exceptions.Exception` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception>)

Specific error signaling characteristics (one or more) does not meet the requirements checked and/or defined in a inherited class of [ProcessorError](#).

**class escape.util.mapping.AbstractMappingDataProcessor (layer\_name)**

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for contain and perform validation steps.

**\_\_init\_\_ (layer\_name)**

**pre\_mapping\_exec (input\_graph, resource\_graph)**

Invoked right before the mapping algorithm.

The given attributes are direct reference to the [NFFG](#) objects which are forwarded to the algorithm.

If there is a return value considering True (e.g. True, not-empty container, collection, an object reference etc.) or a kind of specific ValidationError is thrown in the function the mapping process will be skipped and the orchestration process will be aborted.

The Validator instance is created during the initialization of ESCAPEv2 and used the same instance before/after every mapping process to provide a persistent way to cache data between validations.

#### Parameters

- **input\_graph** ([NFFG](#)) – graph representation which need to be mapped
- **resource\_graph** ([NFFG](#)) – resource information

**Returns** need to abort the mapping process

**Return type** bool or None

**post\_mapping\_exec (input\_graph, resource\_graph, result\_graph)**

Invoked right after if the mapping algorithm is completed without an error.

The given attributes are direct reference to the [NFFG](#) objects the mapping algorithm is worked on.

If there is a return value considering True (e.g. True, not-empty container, collection, an object reference etc.) or a kind of specific ValidationError is thrown in the function the orchestration process will be aborted.

#### Parameters

- **input\_graph** ([NFFG](#)) – graph representation which need to be mapped
- **resource\_graph** ([NFFG](#)) – resource information
- **result\_graph** ([NFFG](#)) – result of the mapping process

**Returns** need to abort the mapping process

**Return type** bool or None

**class** escape.util.mapping.**ProcessorSkipper** (*layer\_name*)  
 Bases: *escape.util.mapping.AbstractMappingDataProcessor*

Default class for skipping validation and proceed to mapping algorithm.

**pre\_mapping\_exec** (*input\_graph*, *resource\_graph*)

**post\_mapping\_exec** (*input\_graph*, *resource\_graph*, *result\_graph*)

**class** escape.util.mapping.**PrePostMapNotifier** (*layer\_name*)  
 Bases: *escape.util.mapping.AbstractMappingDataProcessor*

Notifier class for notifying other POX modules about pre/post map event.

For future features, currently AbstractMapper explicitly raise the event.

**pre\_mapping\_exec** (*input\_graph*, *resource\_graph*)

**post\_mapping\_exec** (*input\_graph*, *resource\_graph*, *result\_graph*)

**class** escape.util.mapping.**PreMapEvent** (*input\_graph*, *resource\_graph*)  
 Bases: *pox.lib.revent.revent.Event*

Raised before the request graph is mapped to the (virtual) resources.

Event handlers might modify the request graph, for example, to enforce some decomposition rules.

**\_\_init\_\_** (*input\_graph*, *resource\_graph*)

**sg**

For support backward compatibility.

**class** escape.util.mapping.**PostMapEvent** (*input\_graph*, *resource\_graph*, *result\_graph*)  
 Bases: *pox.lib.revent.revent.Event*

Raised after the request graph is mapped to the (virtual) resources.

Event handlers might modify the mapped request graph.

**\_\_init\_\_** (*input\_graph*, *resource\_graph*, *result\_graph*)

**class** escape.util.mapping.**AbstractMapper** (*layer\_name*, *strategy=None*, *threaded=None*)  
 Bases: *pox.lib.revent.revent.EventMixin*

Abstract class for graph mapping function.

Inherited from :class`EventMixin` to implement internal event-based communication.

If the Strategy class is not set as DEFAULT\_STRATEGY the it try to search in the CONFIG with the name STRATEGY under the given Layer name.

Contain common functions and initialization.

**DEFAULT\_STRATEGY = None**

**\_\_init\_\_** (*layer\_name*, *strategy=None*, *threaded=None*)

Initialize Mapper class.

Set given strategy class and threaded value or check in *CONFIG*.

If no valid value is found for arguments set the default params defined in *\_default*.

**Warning:** Strategy classes must be a subclass of AbstractMappingStrategy

#### Parameters

- **layer\_name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the layer which initialize this class. This value is used to search the layer configuration in *CONFIG*

- **strategy** (*AbstractMappingStrategy*) – strategy class (optional)
- **threaded** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – run mapping algorithm in separate Python thread instead of in the coop microtask environment (optional)

**Returns** None

#### **\_perform\_mapping** (*input\_graph, resource\_view*)

Abstract function for wrapping optional steps connected to initiate mapping algorithm.

Implemented function call the mapping algorithm.

**Warning:** Derived class have to override this function

#### **Parameters**

- **input\_graph** (*NFFG*) – graph representation which need to be mapped
- **resource\_view** (*AbstractVirtualizer*) – resource information

**Raise** NotImplementedError

**Returns** mapped graph

**Return type** *NFFG*

#### **\_orchestrate** (*input\_graph, resource\_view*)

Abstract function for wrapping optional steps connected to orchestration.

Implemented function call the mapping algorithm.

If a derived class of *AbstractMappingDataProcessor* is set in the global config under the name “PROCESSOR” then the this class performs pre/post mapping steps.

After the pre/post-processor steps the relevant Mapping event will be raised on the main API class of the layer!

**Warning:** Derived class have to override this function

Follows the Template Method design pattern.

#### **Parameters**

- **input\_graph** (*NFFG*) – graph representation which need to be mapped
- **resource\_view** (*AbstractVirtualizer*) – resource information

**Raise** NotImplementedError

**Returns** mapped graph

**Return type** *NFFG*

#### **\_start\_mapping** (*graph, resource*)

Run mapping algorithm in a separate Python thread.

#### **Parameters**

- **graph** (*NFFG*) – Network Function Forwarding Graph
- **resource** (*NFFG*) – global resource

**Returns** None

#### **\_mapping\_finished** (*nffg*)

Called from a separate thread when the mapping process is finished.

**Warning:** Derived class have to override this function

**Parameters** `nffg` (`NFFG`) – generated NF-FG

**Returns** None

```
class escape.util.mapping.AbstractOrchestrator(layer_API, mapper=None, strategy=None)
```

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for common and generic Orchestrator functions.

If the mapper class is not set as `DEFAULT_MAPPER` the it try to search in the CONFIG with the name `MAPPER` under the given Layer name.

`DEFAULT_MAPPER = None`

`__init__(layer_API, mapper=None, strategy=None)`

Init.

**Parameters**

- `layer_API` (`AbstractAPI`) – reference os the actual layer performing the orchestration
- `mapper` (`AbstractMapper`) – additional mapper class (optional)
- `strategy` (`AbstractMappingStrategy`) – override strategy class for the used Mapper (optional)

**Returns** None

***misc.py* module** Contains miscellaneous helper functions.

Singleton

SimpleStandaloneHelper

`schedule_as_coop_task()` helps invoking a function in POX's cooperative microtask environment.

`call_as_coop_task()` hides POC core functionality and schedule a function in the coop microtask environment directly.

`enum()` is a helper function to generate Pythonic enumeration.

`quit_with_error()` is a helper function to terminate POX.

`SimpleStandaloneHelper` is a helper class for mimic a minimal layer API as a dependency for other layer APIs to handles events.

`Singleton` is a metaclass to implements singleton object.

**Module contents** Contains miscellaneous helper functions.

`escape.util.misc.schedule_as_coop_task(func)`

Decorator functions for running functions in an asynchronous way as a microtask in recoco's cooperative multitasking context (in which POX was written).

**Parameters** `func` (`func`) – decorated function

**Returns** decorator function

**Return type** func

`escape.util.misc.schedule_delayed_as_coop_task(delay=0)`

Decorator functions for running functions delayed in recoco's cooperative multitasking context.

**Parameters** `delay` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – delay in sec  
(default: 1s)

**Returns** decorator function

**Return type** func

`escape.util.misc.call_as_coop_task(func, *args, **kwargs)`

Schedule a coop microtask and run the given function with parameters in it.

Use POX core logic directly.

**Parameters**

- `func` (`func`) – function need to run
- `args` (`tuple` (<https://docs.python.org/2.7/library/functions.html#tuple>)) – nameless arguments
- `kwargs` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – named arguments

**Returns** None

`escape.util.misc.call_delayed_as_coop_task(func, delay=0, *args, **kwargs)`

Schedule a coop microtask with a given time.

Use POX core logic directly.

**Parameters**

- `delay` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – delay of time
- `func` (`func`) – function need to run
- `args` (`tuple` (<https://docs.python.org/2.7/library/functions.html#tuple>)) – nameless arguments
- `kwargs` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – named arguments

**Returns** None

`escape.util.misc.run_silent(cmd)`

Run the given shell command silent.

It's advisable to give the command with a raw string literal e.g.: `r'ps aux'`.

**Parameters** `cmd` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – command

**Returns** return code of the subprocess call

**Return type** `int` (<https://docs.python.org/2.7/library/functions.html#int>)

`escape.util.misc.run_cmd(cmd)`

Run a shell command and return the output.

It's advisable to give the command with a raw string literal e.g.: `r'ps aux'`.

**Parameters** `cmd` ([str](https://docs.python.org/2.7/library/functions.html#str)) – command

**Returns** output of the command

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

`escape.util.misc.enum(*sequential, **named)`

Helper function to define enumeration. E.g.:

```
Numbers = enum(ONE=1, TWO=2, THREE='three')
Numbers = enum('ZERO', 'ONE', 'TWO')
Numbers.ONE
1
Numbers.reversed[2]
'TWO'
```

### Parameters

- `sequential` ([list](https://docs.python.org/2.7/library/functions.html#list)) (<https://docs.python.org/2.7/library/functions.html#list>) – support automatic enumeration
- `named` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) (<https://docs.python.org/2.7/library/stdtypes.html#dict>) – support definition with unique keys

**Returns** Enum object

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

`escape.util.misc.quit_with_error(msg, logger=None, exception=False)`

Helper function for quitting in case of an error.

### Parameters

- `msg` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – error message
- `logger` (str or `logging.Logger` (<https://docs.python.org/2.7/library/logging.html#logging.Logger>)) – logger name or logger object (default: core)

**Returns** None

`class escape.util.misc.SimpleStandaloneHelper(container, cover_name)`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Helper class for layer APIs to catch events and handle these in separate handler functions.

`__init__(container, cover_name)`  
Init.

### Parameters

- `container` – Container class reference
- `cover_name` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – Container's name for logging

**Type** EventMixin

`_register_listeners()`  
Register event listeners.

If a listener is explicitly defined in the class use this function otherwise use the common logger function

**Returns** None

`_log_event(event)`  
Log given event.

**Parameters** `event` (*Event*) – Event object which need to be logged

**Returns** None

```
class escape.util.misc.Singleton
Bases: type (https://docs.python.org/2.7/library/functions.html#type)
Metaclass for classes need to be created only once.
Realize Singleton design pattern in a pythonic way.

_instances = {<class 'escape.util.config.ESCAPEConfig'>: <escape.util.config.ESCAPEConfig object at 0x415b790>}
__call__(*args)

escape.util.misc.deprecated(func)
This is a decorator which can be used to mark functions as deprecated. It will result in a warning being emitted when the function is used.

escape.util.misc.remove_junks(log=<logging.Logger object at 0x415b690>)
escape.util.misc.get_ifaces()
Return the list of all defined interface. Rely on ‘ifconfig’ command.

Returns list of interfaces
Return type list (https://docs.python.org/2.7/library/functions.html#list)
```

**netconf.py module** Implement the supporting classes for communication over NETCONF.

Requirements:

```
$ sudo apt-get install python-setuptools python-paramiko python-lxml \
  python-libxml2 python-libxslt1 libxml2 libxslt1-dev
$ sudo pip install ncclient
```

### AbstractNETCONFAdapter

*AbstractNETCONFAdapter* contains the main function for communication over NETCONF such as managing SSH channel, handling configuration, assemble RPC request and parse RPC reply.

**Module contents** Implement the supporting classes for communication over NETCONF.

```
class escape.util.netconf.AbstractNETCONFAdapter(server, port, username, password,
timeout=10, debug=False)
Bases: object (https://docs.python.org/2.7/library/functions.html#object)
Abstract class for various Adapters rely on NETCONF protocol (RFC 4741 (https://tools.ietf.org/html/rfc4741.html)).
```

Contains basic functions for managing connection and invoking RPC calls. Configuration management can be handled by the external [ncclient.manager.Manager](#) (<http://ncclient.readthedocs.org/en/latest/manager.html#ncclient.manager.Manager>) class exposed by the manager property.

Follows the Adapter design pattern.

**NETCONF\_NAMESPACE** = ‘urn:ietf:params:xml:ns:netconf:base:1.0’

**RPC\_NAMESPACE** = None

```
__init__(server, port, username, password, timeout=10, debug=False)
Initialize connection parameters.
```

**Parameters**

- **server** ([str](https://docs.python.org/2.7/library/functions.html#str)) – server address
- **port** ([int](https://docs.python.org/2.7/library/functions.html#int)) – port number
- **username** ([str](https://docs.python.org/2.7/library/functions.html#str)) – username
- **password** ([str](https://docs.python.org/2.7/library/functions.html#str)) – password
- **timeout** ([int](https://docs.python.org/2.7/library/functions.html#int)) – connection timeout (default=30)
- **debug** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – print DEBUG infos, RPC messages etc. (default: False)

**Returns** None**connected****Returns** Return connection state**Return type** [bool](https://docs.python.org/2.7/library/functions.html#bool) (<https://docs.python.org/2.7/library/functions.html#bool>)**connection\_data****Returns** Return connection data in (server, port, username) tuples**Return type** [tuple](https://docs.python.org/2.7/library/functions.html#tuple) (<https://docs.python.org/2.7/library/functions.html#tuple>)**manager****Returns** Return the connection manager (wrapper for NETCONF commands)**Return type** [ncclient.manager.Manager](http://ncclient.readthedocs.org/en/latest/manager.html#ncclient.manager.Manager) (<http://ncclient.readthedocs.org/en/latest/manager.html#ncclient.manager.Manager>)**connect ()**

This function will connect to the netconf server.

**Returns** Also returns the NETCONF connection manager**Return type** [ncclient.manager.Manager](http://ncclient.readthedocs.org/en/latest/manager.html#ncclient.manager.Manager) (<http://ncclient.readthedocs.org/en/latest/manager.html#ncclient.manager.Manager>)**disconnect ()**

This function will close the connection.

**Returns** None**get\_config (source='running', to\_file=False)**This function will download the configuration of the NETCONF agent in an XML format. If source is None then the running config will be downloaded. Other configurations are netconf specific (**RFC 6241** (<https://tools.ietf.org/html/rfc6241.html>)) - running, candidate, startup.**Parameters**

- **source** ([str](https://docs.python.org/2.7/library/functions.html#str)) – NETCONF specific configuration source (default: running)
- **to\_file** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – save config to file

**Returns** None**get (expr='/')**

This process works as yangcli's GET function. A lot of information can be got from the running NETCONF agent. If an xpath-based expression is also set, the results can be filtered. The results are not printed out in a file, it's only printed to stdout.

**Parameters** **expr** ([str](https://docs.python.org/2.7/library/functions.html#str)) – xpath-based expression**Returns** result in XML

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**\_create\_rpc\_request** (`rpc_name`, `**params`)

This function is devoted to create a raw RPC request message in XML format. Any further additional rpc-input can be passed towards, if netconf agent has this input list, called ‘options’. Switches is used for connectVNF rpc in order to set the switches where the vnf should be connected.

**Parameters**

- **rpc\_name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – rpc name
- **options** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – additional RPC input in the specific <options> tag
- **switches** (`list` (<https://docs.python.org/2.7/library/functions.html#list>)) – set the switches where the vnf should be connected
- **params** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – input params for the RPC using param’s name as XML tag name

**Returns** raw RPC message in XML format (lxml library)

**Return type** `lxml.etree.ElementTree`

**\_parse\_rpc\_response** (`data=None`)

Parse raw XML response and return params in dictionary. If data is given it is parsed instead of the last response and the result will not be saved.

**Parameters** `data` (`lxml.etree.ElementTree`) – raw data (uses last reply by default)

**Returns** return parsed params

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**\_invoke\_rpc** (`request_data`)

This function is devoted to call an RPC, and parses the rpc-reply message (if needed) and returns every important parts of it in as a dictionary. Any further additional rpc-input can be passed towards, if netconf agent has this input list, called ‘options’. Switches is used for connectVNF rpc in order to set the switches where the vnf should be connected.

**Parameters** `request_data` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – data for RPC request body

**Returns** raw RPC response

**Return type** `lxml.etree.ElementTree`

**call\_RPC** (`rpc_name`, `no_rpc_error=False`, `**params`)

Call an RPC given by `rpc_name`. If `no_rpc_error` is set returns with a dict instead of raising `RPCError`.

**Parameters**

- **rpc\_name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – RPC name
- **no\_rpc\_error** (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – return with dict (RPC error) instead of exception

**Returns** RPC reply

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**\_\_enter\_\_()**

Context manager setup action.

Usage:

```
with AbstractNETCONFAdapter() as adapter:
    ...
```

**\_exit\_\_(exc\_type, exc\_val, exc\_tb)**  
Context manager cleanup action

**\_AbstractNETCONFAdapter\_\_parse\_rpc\_params(rpc\_request, params)**

Parse given keyword arguments and generate RPC body in proper XML format. The key value is used as the XML tag name. If the value is another dictionary the XML structure follows the hierarchy. The param values can be only simple types and dictionary for simplicity. Conversation example:

```
{
    'vnf_type': 'headerDecompressor',
    'options': {
        'name': 'ip',
        'value': 127.0.0.1
    }
}
```

will be generated into:

```
<rpc-call-name>
<vnf_type>headerDecompressor</vnf_type>
<options>
    <name>ip</name>
    <value>127.0.0.1</value>
</options>
</rpc-call-name>
```

### Parameters

- **rpc\_request** (`lxml.etree.ElementTree`) – empty RPC request
- **params** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – RPC call argument given in a dictionary

**Returns** parsed params in XML format (`lxml` library)

**Return type** `lxml.etree.ElementTree`

**\_AbstractNETCONFAdapter\_\_parse\_xml\_response(element, namespace=None)**

This is an inner function, which is devoted to automatically analyze the rpc-reply message and iterate through all the xml elements until the last child is found, and then create a dictionary. Return a dict with the parsed data. If the reply is OK the returned dict contains an *rcp-reply* element with value *OK*.

### Parameters

- **element** (`lxml.etree.ElementTree`) – XML element
- **namespace** – namespace

**Type** str

**Returns** parsed XML data

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**\_AbstractNETCONFAdapter\_\_remove\_namespace(xml\_element, namespace=None)**

Own function to remove the ncclient's namespace prefix, because it causes “definition not found error” if OWN modules and RPCs are being used.

### Parameters

- **xml\_element** (`lxml.etree.ElementTree`) – XML element
- **namespace** (`lxml.etree.ElementTree`) – namespace

**Returns** cleaned XML element

**Return type** `lxml.etree.ElementTree`

**\_AbstractNETCONFAdapter\_\_suppress\_ncclient\_logging (level=None)**

Suppress annoying and detailed logging of *ncclient* package.

**Parameters** **level** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – level of logging (default: WARNING)

**Returns** None

**nffg.py module** Abstract class and implementation for basic operations with a single NF-FG, such as building, parsing, processing NF-FG, helper functions, etc.

**NFFGToolBox**



*AbstractNFFG* represents the common function for an NFFG representation.

*NFFG* is the internal representation of an NFFG.

*NFFGToolBox* contains helper functions for *NFFG* handling and operations.

**Module contents** Abstract class and implementation for basic operations with a single NF-FG, such as building, parsing, processing NF-FG, helper functions, etc.

**class escape.util.nffg.AbstractNFFG**

Bases: **object** (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for managing single NF-FG data structure.

The NF-FG data model is described in YANG. This class provides the interfaces with the high level data manipulation functions.

**add\_nf()**

Add a single NF node to the NF-FG.

**add\_sap()**

Add a single SAP node to the NF-FG.

**add\_infra()**

Add a single infrastructure node to the NF-FG.

**add\_link (src, dst)**

Add a static or dynamic infrastructure link to the NF-FG.

**Parameters**

- **src** – source port
- **dst** – destination port

**add\_smlink** (*src*, *dst*)  
Add an SG link to the NFFG.

**Parameters**

- **src** – source port
- **dst** – destination port

**add\_req** (*src*, *dst*)  
Add a requirement link to the NFFG.

**Parameters**

- **src** – source port
- **dst** – destination port

**add\_node** (*node*)  
Add a single node to the NFFG.

**Parameters** **node** – node object

**del\_node** (*id*)  
Remove a single node from the NFFG.

**Parameters** **id** – id of the node

**add\_edge** (*src*, *dst*, *link*)  
Add an edge to the NFFG.

**Parameters**

- **src** – source port
- **dst** – destination port
- **link** – link object

**del\_edge** (*src*, *dst*)  
Remove an edge from the NFFG.

**Parameters**

- **src** – source port
- **dst** – destination port

**classmethod parse** (*data*)  
General function for parsing data as a new :any::NFFG object and return with its reference.

**Parameters** **data** ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – raw data

**Returns** parsed NFFG as an XML object

**Return type** Virtualizer

**dump** ()  
General function for dumping :any::NFFG according to its format to plain text.

**Returns** plain text representation

**Return type** [str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)

**class** `escape.util.nffg.NFFG` (*id=None*, *name=None*, *version='1.0'*)  
Bases: `escape.util.nffg.AbstractNFFG`

Internal NFFG representation based on networkx.

**DOMAIN\_INTERNAL** = 'INTERNAL'  
**DOMAIN\_REMOTE** = 'REMOTE'  
**DOMAIN\_VIRTUAL** = 'VIRTUAL'

```

DOMAIN_OS = 'OPENSTACK'
DOMAIN_UN = 'UNIVERSAL_NODE'
DOMAIN_SDN = 'SDN'
DOMAIN_DOCKER = 'DOCKER'
TYPE_INFRA_SDN_SW = 'SDN-SWITCH'
TYPE_INFRA_EE = 'EE'
TYPE_INFRA_STATIC_EE = 'STATIC'
TYPE_INFRA_BISBIS = 'BiSBiS'
TYPE_INFRA = 'INFRA'
TYPE_NF = 'NF'
TYPE_SAP = 'SAP'
TYPE_LINK_STATIC = 'STATIC'
TYPE_LINK_DYNAMIC = 'DYNAMIC'
TYPE_LINK_SG = 'SG'
TYPE_LINK_REQUIREMENT = 'REQUIREMENT'
OPERATION_ADD = 'ADD'
OPERATION_DEL = 'DELETE'
OPERATION_MOD = 'MODIFIED'
OPERATION_MOV = 'MOVED'

__init__(id=None, name=None, version='1.0')
    Init

```

#### Parameters

- **id** (*str or int*) – optional NF-FG identifier (generated by default)
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – optional NF-FG name (generated by default)
- **version** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – optional version (default: 1.0)

#### Returns

**network** = None

Type networkx.MultiDiGraph

**nfs**

**saps**

**infras**

**links**

**sg\_hops**

**reqs**

**\_\_str\_\_()**

**\_\_repr\_\_()**

**\_\_contains\_\_(item)**

Return True if n is a node, False otherwise.

**`__iter__(data=False)`**

Return an iterator over the nodes.

**Parameters** `data` ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – If True return a two-tuple of node and node data dictionary

**Returns** An iterator over nodes.

**`__len__()`**

Return the number of nodes.

**`__getitem__(item)`**

Return the object given by the id: item.

**Parameters** `item` – node id

**Returns** node object

**`add_node(node)`**

Add a Node to the structure.

**Parameters** `node` ([Node](#)) – a Node object

**Returns** None

**`del_node(node)`**

Remove the node from the structure.

**Parameters** `node` (str or [Node](#) or :any‘Port’) – node id or node object or a port object of the node

**Returns** the actual node is found and removed or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**`add_edge(src, dst, link)`**

Add an Edge to the structure.

**Parameters**

- `src` (str or [Node](#) or :any‘Port’) – source node id or Node object or a Port object
- `dst` (str or [Node](#) or :any‘Port’) – destination node id or Node object or a Port object
- `link` ([Link](#)) – edge data object

**Returns** None

**`del_edge(src, dst, id=None)`**

Remove the edge(s) between two nodes.

**Parameters**

- `src` (str or [Node](#) or :any‘Port’) – source node id or Node object or a Port object
- `dst` (str or [Node](#) or :any‘Port’) – destination node id or Node object or a Port object
- `id` (str or int) – unique id of the edge (otherwise remove all)

**Returns** the actual node is found and removed or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**`add_nf(nf=None, id=None, name=None, func_type=None, dep_type=None, cpu=None, mem=None, storage=None, delay=None, bandwidth=None)`**

Add a Network Function to the structure.

**Parameters**

- `nf` ([NodeNF](#)) – add this explicit NF object instead of create one
- `id` (str or int) – optional id
- `name` (str) (<https://docs.python.org/2.7/library/functions.html#str>) – optional name

- **func\_type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – functional type (default: “None”)
- **dep\_type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – deployment type (default: “None”)
- **cpu** (*str or int*) – CPU resource
- **mem** (*str or int*) – memory resource
- **storage** (*str or int*) – storage resource
- **delay** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – delay property of the Node
- **bandwidth** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – bandwidth property of the Node

**Returns** newly created node

**Return type** *NodeNF*

**add\_sap** (*sap=None, id=None, name=None*)

Add a Service Access Point to the structure.

**Parameters**

- **sap** (*NodeSAP*) – add this explicit SAP object instead of create one
- **id** (*str or int*) – optional id
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – optional name

**Returns** newly created node

**Return type** *NodeSAP*

**add\_infra** (*infra=None, id=None, name=None, domain=None, infra\_type=None, cpu=None, mem=None, storage=None, delay=None, bandwidth=None*)

Add an Infrastructure Node to the structure.

**Parameters**

- **infra** (*NodeInfra*) – add this explicit Infra object instead of create one
- **id** (*str or int*) – optional id
- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – optional name
- **domain** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – domain of the Infrastructure Node (default: None)
- **infra\_type** (*int or str*) – type of the Infrastructure Node (default: 0)
- **cpu** (*str or int*) – CPU resource
- **mem** (*str or int*) – memory resource
- **storage** (*str or int*) – storage resource
- **delay** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – delay property of the Node
- **bandwidth** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – bandwidth property of the Node

**Returns** newly created node

**Return type** *NodeInfra*

**add\_link** (*src\_port, dst\_port, link=None, id=None, dynamic=False, backward=False, delay=None, bandwidth=None*)

Add a Link to the structure.

**Parameters**

- **link** (*EdgeLink*) – add this explicit Link object instead of create one
- **src\_port** (*Port*) – source port
- **dst\_port** (*Port*) – destination port
- **id** (*str or int*) – optional link id
- **backward** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – the link is a backward link compared to an another Link
- **delay** (*str or int*) – delay resource
- **dynamic** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – set the link dynamic (default: False)
- **bandwidth** (*str or int*) – bandwidth resource

**Returns** newly created edge**Return type** *EdgeLink*

**add\_undirected\_link** (*port1*, *port2*, *p1p2id=None*, *p2p1id=None*, *dynamic=False*, *delay=None*, *bandwidth=None*)

Add two Links to the structure, in both directions.

**Parameters**

- **port1** (*Port*) – source port
- **port2** (*Port*) – destination port
- **p1p2id** (*str or int*) – optional link id from port1 to port2
- **p2p1id** (*str or int*) – optional link id from port2 to port1
- **delay** (*str or int*) – delay resource of both links
- **dynamic** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – set the link dynamic (default: False)
- **bandwidth** (*str or int*) – bandwidth resource of both links

**Returns** newly created edge tuple in (p1->p2, p2->p1)**Return type** :any:(*EdgeLink*, *EdgeLink*)

**add\_smlink** (*src\_port*, *dst\_port*, *hop=None*, *id=None*, *flowclass=None*)

Add a SD next hop edge to the structure.

**Parameters**

- **hop** (*EdgeSGLink*) – add this explicit SG Link object instead of create one
- **src\_port** (*Port*) – source port
- **dst\_port** (*Port*) – destination port
- **id** (*str or int*) – optional link id
- **flowclass** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – flowclass of SG next hop link

**Returns** newly created edge**Return type** *EdgeSGLink*

**add\_req** (*src\_port*, *dst\_port*, *req=None*, *id=None*, *delay=None*, *bandwidth=None*, *sg\_path=None*)

Add a requirement edge to the structure.

**Parameters**

- **req** (*EdgeReq*) – add this explicit Requirement Link object instead of create one

- **src\_port** ([Port](#)) – source port
- **dst\_port** ([Port](#)) – destination port
- **id** (*str or int*) – optional link id
- **delay** (*str or int or float*) – delay resource
- **bandwidth** (*str or int or float*) – bandwidth resource
- **sg\_path** (*list or tuple*) – list of ids of sg\_links represents end-to-end requirement

**Returns** newly created edge

**Return type** [EdgeReq](#)

#### **dump()**

Convert the NF-FG structure to a NFFGModel format and return the plain text representation.

**Returns** text representation

**Return type** [str](#) (<https://docs.python.org/2.7/library/functions.html#str>)

#### **classmethod parse (raw\_data)**

Read the given JSON object structure and try to convert to an NF-FG representation as an [NFFG](#)

**Parameters** **raw\_data** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – raw NF-FG description as a string

**Returns** the parsed NF-FG representation

**Return type** [NFFG](#)

#### **duplicate\_static\_links()**

Extend the NFFG model with backward links for STATIC links to fit for the orchestration algorithm.

STATIC links: infra-infra, infra-sap

**Returns** NF-FG with the duplicated links for function chaining

**Return type** [NFFG](#)

#### **merge\_duplicated\_links()**

Detect duplicated STATIC links which both are connected to the same Port/Node and have switched source/destination direction to fit for the simplified NFFG dumping.

Only leaves one of the links, but that's not defined which one.

**Returns** NF-FG with the filtered links for function chaining

**Return type** [NFFG](#)

#### **infra\_neighbors (node\_id)**

Return an iterator for the Infra nodes which are neighbours of the given node.

**Parameters** **node\_id** ([NodeInfra](#)) – infra node

**Returns** iterator for the list of Infra nodes

#### **running\_nfs (infra\_id)**

Return an iterator for the NodeNFs which are mapped to the given Infra node.

**Params** **infra\_id** infra node identifier

**Returns** iterator for the currently running NodeNFs

#### **clear\_links (link\_type)**

Remove every specific Link from the NFFG defined by given type.

**Parameters** **link\_type** – link type defined in [NFFG](#)

**Returns** None

**clear\_nodes (node\_type)**

Remove every specific Node from the NFFG defined by given type.

**Parameters** **node\_type** – node type defined in [NFFG](#)

**Returns** None

**copy ()**

Return the deep copy of the NFFG object.

**Returns** deep copy

**Return type** [NFFG](#)

**generate\_id ()**

Generate a unique id from object memory address.

**Returns** generated id

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

**class escape.util.nffg.NFFGToolBox**

Bases: [object](#) (<https://docs.python.org/2.7/library/functions.html#object>)

Helper functions for NFFG handling.

**static merge\_domains (base, nffg)**

Merge the given nffg into the base NFFG.

**Parameters**

- **base** ([NFFG](#)) – base NFFG object
- **nffg** ([NFFG](#)) – updating information

**Returns** the update base NFFG

**Return type** [NFFG](#)

**static split\_domains (nffg)**

Split NFFG object into separate parts based on DOMAIN attribute.

**Parameters** **nffg** ([NFFG](#)) – global resource view (DoV)

**Returns** splitted parts as list ov domain name, domain part tuples

**Return type** tuple (<https://docs.python.org/2.7/library/functions.html#tuple>)

**static install\_domain (virtualizer, nffg)**

Install NFFG part or complete NFFG into given Virtualizer.

**Parameters**

- **virtualizer** – Virtualizer object based on ETH's XML/Yang version.
- **nffg** – splitted NFFG (not necessarily in valid syntax)

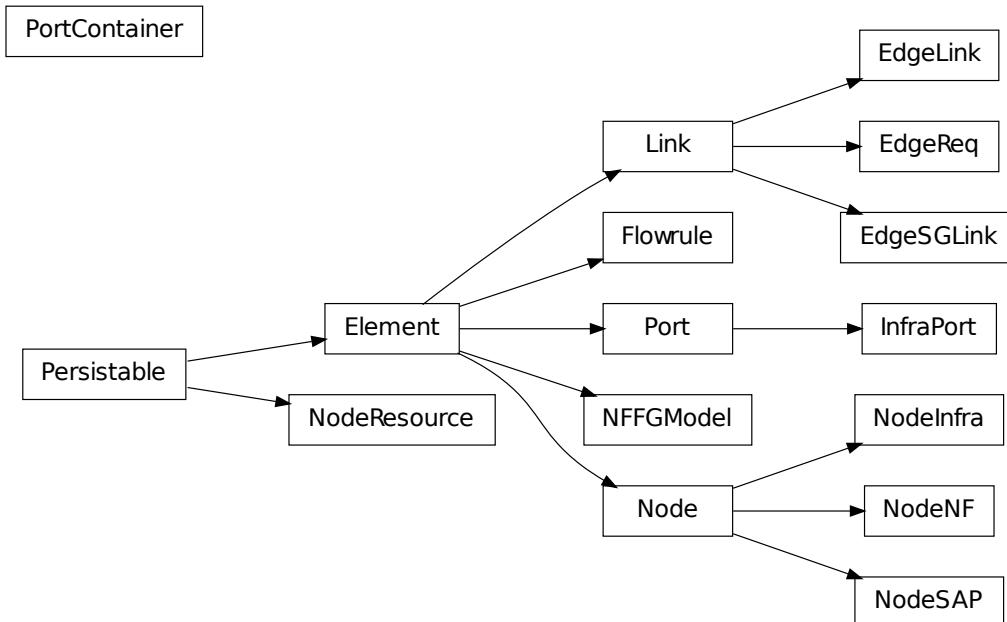
**Returns** modified Virtualizer object

**static \_get\_output\_port\_of\_TAG\_action (TAG, port)****static \_find\_static\_link (nffg, port, outbound=True)****static \_is\_port\_finishing\_flow (TAG, port)****static get\_TAGS\_of\_starting\_flows (port)****static retrieve\_mapped\_path (TAG, nffg, starting\_port)**

Finds the list of links, where the traffic tagged with the given TAG is routed. starting\_port is the first port where the tag is put onto the traffic (the outbound dynamic port of the starting VNF of the flow). Returns the list of link objects and the corresponding bandwidth value. TODO (?): add default 'None' parameter value for starting\_port , when the function should find where the given TAG is put on the traffic

```
static generate_all_TAGS_of_NFFG(nffg)
```

**nffg\_elements.py module** Element classes for NFFG based on nffg.yang.



`NFFGModel` represents the main container class.

`Persistable` ensures the basic parse/dump functionality.

`Element` represents the common functions for elements.

`Node` represents the common functions for Node elements.

`Link` represents the common functions for Edge elements.

`NodeResource` represents the resource attributes of a Node.

`Flowrule` represents the attributes of a flowrule.

`Port` represents a port of a Node.

`InfraPort` extends the port capabilities for the Infrastructure Node.

`NodeNF` defines the NF type of Node.

`NodeSAP` defines the SAP type of Node.

`NodeInfra` defines the Infrastructure type of Node.

`EdgeLink` defines the dynamic and static connections between Nodes.

`EdgeSGLink` defines the connection between SG elements.

`EdgeReq` defines the requirements between SG elements.

**Module contents** Classes for handling the elements of the NF-FG data structure

**class** `escape.util.nffg_elements.Persistable`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Define general persist function for the whole NFFG structure.

**`persist()`**

Common function to persist the actual element into a plain text format.

**Returns** generated object structure fit to JSON

**Return type** `object` (<https://docs.python.org/2.7/library/functions.html#object>)

**`load(data, *args, **kwargs)`**

Common function to fill self with data from JSON data.

**Parameters** `data` – object structure in JSON

**Returns** self

**`classmethod parse(data, *args, **kwargs)`**

Common function to parse the given JSON object structure as the actual NF-FG entity type and return a newly created object.

**Parameters** `data` ([object](https://docs.python.org/2.7/library/functions.html#object) (<https://docs.python.org/2.7/library/functions.html#object>)) – raw JSON object structure

**Returns** parsed data as the entity type

**Return type** `Persistable`

**`class escape.util.nffg_elements.Element(id=None, type='ELEMENT', operation=None)`**

Bases: `escape.util.nffg_elements.Persistable`

Main base class for NF-FG elements with unique id.

Contains the common functionality.

**ADD = 'ADD'**

**DEL = 'DELETE'**

**MOD = 'MODIFIED'**

**MOV = 'MOVED'**

**`__init__(id=None, type='ELEMENT', operation=None)`**

Init.

**Parameters**

- **`id`** (*str or int*) – optional identification (generated by default)
- **`type`** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – explicit object type both for nodes and edges

**Returns** None

**`persist()`****`load(data, *args, **kwargs)`****`copy()`****`__getitem__(item)`****`__setitem__(key, value)`****`__contains__(item)`****`get(item, default=None)`****`setdefault(key, default=None)`****`clear()`****`update(dict2)`**

```
class escape.util.nffg_elements.PortContainer (container=None)
Bases: object (https://docs.python.org/2.7/library/functions.html#object)
Basic container class for ports.

Implements a Container-like behavior for getting a Port with id: cont = PortContainer() ...
cont[“port_id”]

__init__ (container=None)
__getitem__ (id)
__iter__ ()
__len__ ()
__contains__ (item)
append (item)
remove (item)
clear ()
__str__ ()
__repr__ ()

class escape.util.nffg_elements.Node (type, id=None, name=None)
Bases: escape.util.nffg\_elements.Element

Base class for different types of nodes in the NF-FG.

INFRA = ‘INFRA’
SAP = ‘SAP’
NF = ‘NF’

__init__ (type, id=None, name=None)
Init.

Parameters

- type (str (https://docs.python.org/2.7/library/functions.html#str)) – node type
- id (str or int) – optional id
- name (str (https://docs.python.org/2.7/library/functions.html#str)) – optional name

Returns None

short_name
add_port (id=None, properties=None)
Add a port with the given params to the Node.

Parameters

- id (str or int) – optional id
- properties (str or iterable\(str\)) – supported properties of the port (one or more as list)

Returns newly created and stored Port object

Return type Port

del_port (id)
Remove the port with the given id from the Node.

Parameters id (int or str) – port id

Returns the actual Port is found and removed or not
```

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

```
persist()
load(data, *args, **kwargs)
__repr__()
__str__()

class escape.util.nffg_elements.Link(src=None, dst=None, type=None, id=None)
Bases: escape.util.nffg_elements.Element

Base class for different types of edges in the NF-FG.

STATIC = 'STATIC'
DYNAMIC = 'DYNAMIC'
SG = 'SG'
REQUIREMENT = 'REQUIREMENT'

__init__ (src=None, dst=None, type=None, id=None)
Init.
```

#### Parameters

- **src** (*Port*) – source port
- **dst** (*Port*) – destination port
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – link type
- **id** (*str or int*) – optional id

#### Returns

None

```
persist()
load(data, container=None, *args, **kwargs)
__repr__()

class escape.util.nffg_elements.NodeResource(cpu=None, mem=None, storage=None,
                                             delay=None, bandwidth=None)
Bases: escape.util.nffg_elements.Persistable

Class for storing resource information for Nodes.

__init__ (cpu=None, mem=None, storage=None, delay=None, bandwidth=None)
Init.
```

#### Parameters

- **cpu** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – CPU resource
- **mem** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – memory resource
- **storage** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – storage resource
- **delay** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – delay property of the Node
- **bandwidth** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – bandwidth property of the Node

#### Returns

None

```
persist()
load(data, *args, **kwargs)
```

```
__getitem__(item)
__setitem__(key, value)
__repr__()
__str__()

class escape.util.nffg_elements.Flowrule(id=None,      match=''',      action=''',      band-
                                         width=None)
Bases: escape.util.nffg_elements.Element

Class for storing a flowrule.

__init__(id=None, match=''', action=''', bandwidth=None)
    Init.
```

**Parameters**

- **match** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – matching rule
- **action** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – forwarding action

**Returns** None

```
persist()
load(data, *args, **kwargs)
__repr__()
__str__()
```

```
class escape.util.nffg_elements.Port(node, properties=None, id=None)
Bases: escape.util.nffg_elements.Element
```

Class for storing a port of an NF.

**TYPE = ‘PORT’**

```
__init__(node, properties=None, id=None)
    Init.
```

**Parameters**

- **node** ([Node](#)) – container node
- **id** ([str or int](#)) – optional id
- **properties** ([str or iterable\(str\)](#)) – supported properties of the port

**Returns** None

**node**

**add\_property** (property, value=None)

Add a property or list of properties to the port. If value is not None, then property is used as a key.

**Parameters**

- **property** ([str or list or tuple](#)) – property
- **value** ([str](#) (<https://docs.python.org/2.7/library/functions.html#str>)) – optional property value

**Returns** the Port object to allow function chaining**Return type** [Port](#)

**del\_property** (property=None)

Remove the property from the Port. If no property is given remove all the properties from the Port.

**Parameters** `property` ([str](https://docs.python.org/2.7/library/functions.html#str)) – property

**Returns** None

**get\_property** (`property`)  
Return the value of the property

**Parameters** `property` ([str](https://docs.python.org/2.7/library/functions.html#str)) – property

**Returns** the value of the property

**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

**persist()**

**load** (`data, *args, **kwargs`)

**\_\_repr\_\_()**

**class** `escape.util.nffg_elements.InfraPort` (`node, properties=None, id=None`)  
Bases: `escape.util.nffg_elements.Port`

Class for storing a port of Infra Node and handles flowrules.

**\_\_init\_\_** (`node, properties=None, id=None`)  
Init.

**Parameters**

- `node` ([Node](#)) – container node
- `id` ([str or int](#)) – optional id
- `properties` ([str or iterable\(str\)](#)) – supported properties of the port

**Returns** None

**add\_flowrule** (`match, action, bandwidth=None, id=None`)  
Add a flowrule with the given params to the port of an Infrastructure Node.

**Parameters**

- `match` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – matching rule
- `action` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – forwarding action
- `bandwidth` ([int](https://docs.python.org/2.7/library/functions.html#int)) (<https://docs.python.org/2.7/library/functions.html#int>) – bandwidth value
- `id` ([str or int](#)) – specific id of the flowrule

**Returns** newly created and stored flowrule

**Return type** `Flowrule`

**del\_flowrule** (`id=None, match=None, action=None`)  
Remove the flowrule with the given id or all flowrules which match the given action/match parameters.

**Parameters**

- `id` ([int or str](#)) – flowrule id
- `match` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – matching rule
- `action` ([str](https://docs.python.org/2.7/library/functions.html#str)) (<https://docs.python.org/2.7/library/functions.html#str>) – forwarding action

**Returns** the actual FlowRule is found and removed or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

```
persist()
load(data, *args, **kwargs)
class escape.util.nffg_elements.NodeNF (id=None, name=None, func_type=None, dep_type=None, res=None)
Bases: escape.util.nffg_elements.Node

Network Function (NF) nodes in the graph.

__init__(id=None, name=None, func_type=None, dep_type=None, res=None)
    Init.

Parameters

- func_type (str (https://docs.python.org/2.7/library/functions.html#str)) – functional type (default: “None”)
- dep_type (str (https://docs.python.org/2.7/library/functions.html#str)) – deployment type (default: “None”)
- res (NodeResource) – optional NF resources

Returns None

persist()
load(data, *args, **kwargs)
__str__()

class escape.util.nffg_elements.NodeSAP (id=None, name=None, domain=None)
Bases: escape.util.nffg_elements.Node

Class for SAP nodes in the NF-FG.

__init__(id=None, name=None, domain=None)
__str__()
__repr__()
persist()
load(data, *args, **kwargs)
class escape.util.nffg_elements.NodeInfra (id=None, name=None, domain=None, infra_type=None, supported=None, res=None)
Bases: escape.util.nffg_elements.Node

Class for infrastructure nodes in the NF-FG.

TYPE_BISBIS = ‘BiSBiS’
TYPE_EE = ‘EE’
TYPE_STATIC_EE = ‘STATIC’
TYPE_SDN_SWITCH = ‘SDN-SWITCH’
DOMAIN_VIRTUAL = ‘VIRTUAL’
DOMAIN_INTERNAL = ‘INTERNAL’
DOMAIN_REMOTE = ‘REMOTE’
DOMAIN_OS = ‘OPENSTACK’
DOMAIN_UN = ‘UNIVERSAL_NODE’
DOMAIN_SDN = ‘SDN’
DOMAIN_DOCKER = ‘DOCKER’
```

---

**\_\_init\_\_(id=None, name=None, domain=None, infra\_type=None, supported=None, res=None)**  
Init.

#### Parameters

- **domain** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – domain of the Infrastructure Node
- **infra\_type** (*int or str*) – type of the Infrastructure Node
- **supported** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – list of supported functional types
- **res** (*NodeResource*) – optional Infra resources

**Returns** None

**add\_port(id=None, properties=None)**

Add a port with the given params to the Infrastructure Node.

#### Parameters

- **id** (*str or int*) – optional id
- **properties** (*str or iterable(str)*) – supported properties of the port (one or more as list)

**Returns** newly created and stored Port object

**Return type** *Port*

**add\_supported\_type(functional\_type)**

Add a supported functional type or list of types to the Infrastructure Node.

**Parameters** **functional\_type** (*str or list or tuple*) – the functional type

**Returns** the Node object to allow function chaining

**Return type** *NodeInfra*

**del\_supported\_type(functional\_type=None)**

Remove the given functional type from the Infrastructure Node. If no type is given then all supported type will be removed.

**Parameters** **functional\_type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – the functional type

**Returns** None

**persist()**

**load(data, \*args, \*\*kwargs)**

**\_\_str\_\_()**

**\_\_repr\_\_()**

**class escape.util.nffg\_elements.EdgeLink(src=None, dst=None, type=None, id=None, backward=False, delay=None, bandwidth=None)**

Bases: *escape.util.nffg\_elements.Link*

Class for static and dynamic links in the NF-FG.

Represent a static or dynamic link.

**\_\_init\_\_(src=None, dst=None, type=None, id=None, backward=False, delay=None, bandwidth=None)**

Init.

#### Parameters

- **src** (*Port*) – source port

- **dst** (*Port*) – destination port
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – type of the link  
(default: Link.STATIC)
- **id** (*str or int*) – optional link id
- **backward** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – the link is a backward link compared to an another Link
- **delay** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – delay resource
- **bandwidth** (*float* (<https://docs.python.org/2.7/library/functions.html#float>)) – bandwidth resource

**Returns** None

**persist** ()

**load** (*data, container=None, \*args, \*\*kwargs*)

**\_\_str\_\_** ()

**\_\_repr\_\_** ()

**class** *escape.util.nffg\_elements.EdgeSGLink* (*src=None, dst=None, id=None, flowclass=None*)  
Bases: *escape.util.nffg\_elements.Link*

Class for links of SG.

Represent an edge between SG elements.

**\_\_init\_\_** (*src=None, dst=None, id=None, flowclass=None*)

Init.

#### Parameters

- **src** (*Port*) – source port
- **dst** (*Port*) – destination port
- **id** (*str or int*) – optional id
- **flowclass** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – flowclass of SG next hop link a.k.a a match

**Returns** None

**persist** ()

**load** (*data, container=None, \*args, \*\*kwargs*)

**class** *escape.util.nffg\_elements.EdgeReq* (*src=None, dst=None, id=None, delay=None, bandwidth=None, sg\_path=None*)  
Bases: *escape.util.nffg\_elements.Link*

Class for constraint of networking parameters between SG elements.

Class for requirements between arbitrary NF modes.

**\_\_init\_\_** (*src=None, dst=None, id=None, delay=None, bandwidth=None, sg\_path=None*)  
Init.

#### Parameters

- **src** (*Port*) – source port
- **dst** (*Port*) – destination port
- **id** (*str or int*) – optional id

- **delay** ([float](https://docs.python.org/2.7/library/functions.html#float) (<https://docs.python.org/2.7/library/functions.html#float>)) – delay resource
- **bandwidth** ([float](https://docs.python.org/2.7/library/functions.html#float) (<https://docs.python.org/2.7/library/functions.html#float>)) – bandwidth resource
- **sg\_path** (*list or tuple*) – list of ids of sg\_links represents end-to-end requirement

**Returns** None

**persist()**

**load** (*data, container=None, \*args, \*\*kwargs*)

**class** `escape.util.nffg_elements.NFFGModel` (*id=None, name=None, version=None*)

Bases: `escape.util.nffg_elements.Element`

Wrapper class for a single NF-FG.

Network Function Forwarding Graph (NF-FG) data model.

**VERSION = ‘1.0’**

**NAMESPACE = ‘http://csikor.tmit.bme.hu/netconf/unify/nffg’**

**PREFIX = ‘nffg’**

**ORGANIZATION = ‘BME-TMIT’**

**DESCRIPTION = ‘Network Function Forwarding Graph (NF-FG) data model’**

**TYPE = ‘NFFG’**

**\_\_init\_\_** (*id=None, name=None, version=None*)

Init

#### Parameters

- **id** (*str or int*) – optional NF-FG identifier (generated by default)
- **name** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – optional NF-FG name
- **version** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – optional version (default: 1.0)

**Returns** None

**nodes**

Return all the node in the Container as a list.

**Returns** nodes

**Return type** `list` (<https://docs.python.org/2.7/library/functions.html#list>)

**edges**

Return all the edges in the Container as a list.

**Returns** edges

**Return type** `list` (<https://docs.python.org/2.7/library/functions.html#list>)

**get\_port** (*node\_id, port\_id*)

Return the Port reference according to the given Node and Port ids.

#### Parameters

- **node\_id** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – node id
- **port\_id** ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – port id

**Returns** port object

**Return type** `Port`

**add\_nf (\*\*kwargs)**

Create and store a NF Node with the given parameters.

**Returns** the created NF

**Return type** *NodeNF*

**del\_nf (id)**

Remove the NF Node with the given id.

**Parameters**

- **id** – NF id
- **id** – str

**Returns** the actual Node is found and removed or not

**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)

**add\_sap (\*\*kwargs)**

Create and store a SAP Node with the given parameters.

**Returns** the created SAP

**Return type** *NodeSAP*

**del\_sap (id)**

Remove the SAP Node with the given id.

**Parameters**

- **id** – SAP id
- **id** – str

**Returns** the actual Node is found and removed or not

**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)

**add\_infra (\*\*kwargs)**

Create and store an Infrastructure Node with the given parameters.

**Returns** the created Infra

**Return type** *NodeInfra*

**del\_infra (id)**

Remove Infrastructure Node with the given id.

**Parameters**

- **id** – Infra id
- **id** – str

**Returns** the actual Node is found and removed or not

**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)

**add\_link (src, dst, \*\*kwargs)**

Create and store a Link Edge with the given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the created edge

**Return type** *EdgeLink*

**del\_link (src, dst)**

Remove Link Edge with given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the actual Edge is found and removed or not**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)**add\_sg\_hop** (*src*, *dst*, *\*\*kwargs*)

Create and store an SG next hop Edge with the given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the created edge**Return type** *EdgeSGLink***del\_sg\_hop** (*src*, *dst*)

Remove SG next hop Edge with given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the actual Edge is found and removed or not**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)**add\_req** (*src*, *dst*, *\*\*kwargs*)

Create and store a Requirement Edge with the given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the created edge**Return type** *EdgeReq***del\_req** (*src*, *dst*)

Remove Requirement Edge with given src and dst nodes.

**Parameters**

- **src** (*Node*) – source node
- **dst** (*Node*) – destination node

**Returns** the actual Edge is found and removed or not**Return type** *bool* (<https://docs.python.org/2.7/library/functions.html#bool>)**persist()****load** (*raw\_data*, *\*args*, *\*\*kwargs*)Read the given JSON object structure and try to convert to an NF-FG representation as an *NFFGModel*.**Parameters** **raw\_data** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – raw date in JSON**Returns** the constructed NF-FG representation**Return type** *NFFGModel*

**dump()**

Dump the container in plain text based on JSON structure.

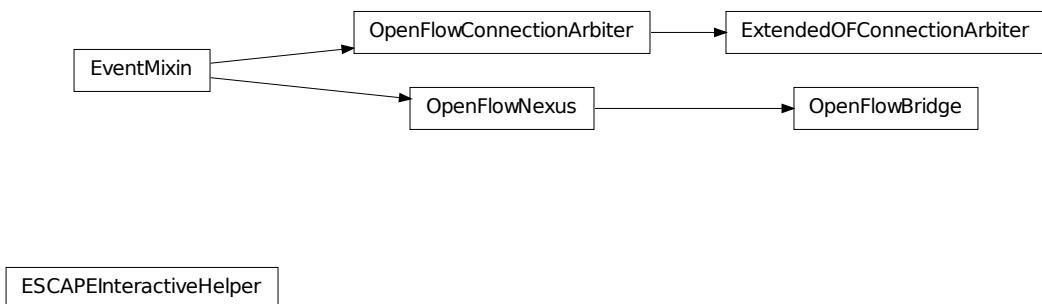
**Returns** NF-FG representation as plain text

**Return type** str (<https://docs.python.org/2.7/library/functions.html#str>)

```
escape.util.nffg_elements.test_parse_load()
```

```
escape.util.nffg_elements.test_networkx_mod()
```

**pox\_extension.py module** Override and extend internal POX components to achieve ESCAPE-desired behaviour.



*OpenFlowBridge* is a special version of OpenFlow event originator class.

*ExtendedOFConnectionArbiter* dispatches incoming OpenFlow connections to fit ESCAPEv2.

*ESCAPEInteractiveHelper* contains helper function for debugging.

**Module contents** Override and extend internal POX components to achieve ESCAPE-desired behaviour.

**class escape.util.pox\_extension.OpenFlowBridge**

Bases: pox.openflow.OpenFlowNexus

Own class for listening OpenFlow event originated by one of the contained Connection and sending OpenFlow messages according to DPID.

Purpose of the class mostly fits the Bride design pattern.

**clear\_flows\_on\_connect = False**

**class escape.util.pox\_extension.ExtendedOFConnectionArbiter (default=False)**

Bases: pox.openflow.OpenFlowConnectionArbiter

Extended connection arbiter class for dispatching incoming OpenFlow Connection between registered OF event originators (OpenFlowNexus) according to the connection's listening address.

**\_core\_name = 'OpenFlowConnectionArbiter'**

**\_\_init\_\_ (default=False)**

Init.

**Parameters** **default** (OpenFlowNexus) – inherited param

**add\_connection\_listener (address, nexus)**

Helper function to register connection listeners a.k.a. OpenFlowNexus.

**Parameters**

- **address** (*tuple*) (<https://docs.python.org/2.7/library/functions.html#tuple>) – listened socket name in form of (address, port)

- **nexus** (*OpenFlowBridge*) – registered object

**Returns** registered listener

**Return type** *OpenFlowBridge*

**classmethod activate()**  
Register this component into `pox.core` and replace already registered Arbiter.

**Returns** registered component

**Return type** *ExtendedOFConnectionArbiter*

**getNexus(connection)**  
Return registered connection listener or default `core.openflow`.

Fires ConnectionIn event.

**Parameters** `connection` (`Connection`) – incoming connection object

**Returns** OpenFlow event originator object

**Return type** *OpenFlowNexus*

**class escape.util.pox\_extension.ESCAPEInteractiveHelper**  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Extended Interactive class which add ESCAPE specific debug functions to POX's py module.

**\_\_repr\_\_()**  
return with defined helper functions.

**static init()**  
Register an ESCPAEInteractiveHelper into POX's core.

**ping()**  
Call the ping() function of the OpenStackRESTAdapter.

**get\_config()**  
Call the get\_config() function of the OpenStackRESTAdapter.

**edit\_config()**  
Call the edit\_config() function of OpenStackRESTAdapter with the default config.

**config()**  
Dump running config (CONFIG)

**Returns** None

## 8.2 Topmost POX modules for UNIFY's layers/sublayers

### 8.2.1 The *unify.py* top module

Basic POX module for ESCAPE

Initiate appropriate APIs

Follows POX module conventions

**unify.\_start\_components(event)**  
Initiate and run POX with ESCAPE components.

**Parameters** `event` (`GoingUpEvent`) – POX's going up event

**Returns** None

**unify.launch(sg\_file='', config=None, gui=False, agent=False, rosapi=False, full=False, debug=True, cfor=False, topo=None)**  
Launch function called by POX core when core is up.

## Parameters

- **sg\_file** ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str) – Path of the input Service graph (optional)
- **config** ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str) – additional config file with different name
- **gui** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – Signal for initiate GUI (optional)
- **agent** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – Do not start the service layer (optional)
- **rosapi** –
- **full** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – Initiate Infrastructure Layer also
- **debug** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – run in debug mode (optional)
- **cfor** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – start Cf-Or REST API (optional)
- **topo** ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str) – Path of the initial topology graph (optional)

**Returns** None

## Submodules

### The `service.py` main module

Basic POX module for ESCAPE Service (Graph Adaptation) sublayer

Initiate appropriate API class which implements U-SI reference point

Follows POX module conventions

```
service._start_layer(event)
    Initiate and run Service module.
```

**Parameters** **event** (*GoingUpEvent*) – POX's going up event

**Returns** None

```
service.launch(sg_file='', gui=False, standalone=False)
```

Launch function called by POX core when core is up.

#### Parameters

- **sg\_file** ([str](https://docs.python.org/2.7/library/functions.html#str)) (https://docs.python.org/2.7/library/functions.html#str) – Path of the input Service graph (optional)
- **gui** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – Initiate built-in GUI (optional)
- **standalone** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) (https://docs.python.org/2.7/library/functions.html#bool) – Run layer without dependency checking (optional)

**Returns** None

## Service related classes

**`escape.service` package** Subpackage for classes related mostly to Service (Graph) Adaptation sublayer

## Submodules

**`element_mgmt.py` module** Contains classes relevant to element management.



`AbstractElementManager` is an abstract class for element managers.

`ClickManager` represent the interface to Click elements.

**Module contents** Contains classes relevant to element management.

**class** `escape.service.element_mgmt.AbstractElementManager`  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for element management components (EM).

**Warning:** Not implemented yet!

`__init__()`  
Init

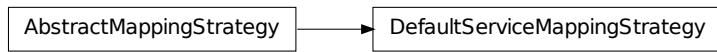
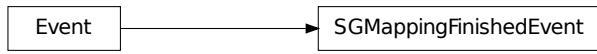
**class** `escape.service.element_mgmt.ClickManager`  
Bases: `escape.service.element_mgmt.AbstractElementManager`

Manager class for specific VNF management based on Clicky.

**Warning:** Not implemented yet!

`__init__()`  
Init.

**`sas_mapping.py` module** Contains classes which implement SG mapping functionality.



`DefaultServiceMappingStrategy` implements a default mapping algorithm which map given SG on a single Bis-Bis.

`SGMappingFinishedEvent` can signal end of service graph mapping.

`ServiceGraphMapper` perform the supplementary tasks for SG mapping.

**Module contents** Contains classes which implement SG mapping functionality.

**class** `escape.service.sas_mapping.DefaultServiceMappingStrategy`  
Bases: `escape.util.mapping.AbstractMappingStrategy`

Mapping class which maps given Service Graph into a single BiS-BiS.

`__init__()`  
Init.

**classmethod** `map(graph, resource)`

Default mapping algorithm which maps given Service Graph on one BiS-BiS.

**Parameters**

- `graph` (`NFFG`) – Service Graph
- `resource` (`NFFG`) – virtual resource

**Returns** Network Function Forwarding Graph

**Return type** `NFFG`

**class** `escape.service.sas_mapping.SGMappingFinishedEvent(nffg)`

Bases: `pox.lib.revent.revent.Event`

Event for signaling the end of SG mapping.

`__init__(nffg)`  
Init.

**Parameters** `nffg` (`NFFG`) – NF-FG need to be initiated

**class** `escape.service.sas_mapping.ServiceGraphMapper(strategy=None)`

Bases: `escape.util.mapping.AbstractMapper`

Helper class for mapping Service Graph to NF-FG.

`_eventMixin_events = set([<class 'escape.service.sas_mapping.SGMappingFinishedEvent'>])`

**DEFAULT\_STRATEGY**

alias of `DefaultServiceMappingStrategy`

`__init__(strategy=None)`  
Init Service mapper.

**Returns** None

`_perform_mapping(input_graph, resource_view)`

Orchestrates mapping of given service graph on given virtual resource.

**Parameters**

- `input_graph` (`NFFG`) – Service Graph
- `resource_view` – virtual resource view
- `resource_view` – `AbstractVirtualizer`

**Returns** Network Function Forwarding Graph

**Return type** `NFFG`

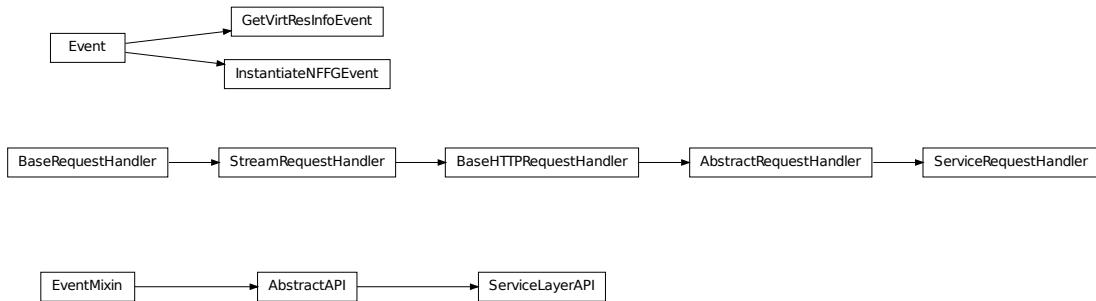
`_mapping_finished(nffg)`

Called from a separate thread when the mapping process is finished.

**Parameters** `nffg` (`NFFG`) – generated NF-FG

**Returns** None

**sas\_API.py module** Implements the platform and POX dependent logic for the Service Adaptation Sublayer.



`InstantiateNFFGEEvent` can send NF-FG to the lower layer.

`GetVirtResInfoEvent` can request virtual resource info from lower layer.

`ServiceRequestHandler` implement the specific RESTful API functionality thereby realizes the UNIFY's U - SI API.

`ServiceLayerAPI` represents the SAS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Service Adaptation Sublayer.

**class** `escape.service.sas_API.InstantiateNFFGEEvent (nffg)`

Bases: `pox.lib.revent.revent.Event`

Event for passing NFFG (mapped SG) to Orchestration layer.

**\_\_init\_\_** (`nffg`)  
Init.

**Parameters** `nffg` (`NFFG`) – NF-FG need to be initiated

**class** `escape.service.sas_API.GetVirtResInfoEvent (sid)`

Bases: `pox.lib.revent.revent.Event`

Event for requesting virtual resource info from Orchestration layer.

**\_\_init\_\_** (`sid`)  
Init.

**Parameters** `sid` (`int` (<https://docs.python.org/2.7/library/functions.html#int>)) – Service layer ID

**class** `escape.service.sas_API.ServiceRequestHandler (request, client_address, server)`

Bases: `escape.util.api.AbstractRequestHandler`

Request Handler for Service Adaptation SubLayer.

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronization is needed to take care manually: use relevant helper function of core object: `callLater/raiseLater` or use `schedule_as_coop_task` decorator defined in `util.misc` on the called function.

```

request_perm = {'POST': ('ping', 'result', 'sg', 'topology'), 'GET': ('ping', 'version', 'operations', 'topology')}
bounded_layer = 'service'
  
```

```
log = <logging.Logger object at 0x4ccca10>

result()
    Return the result of a request given by the id.

sg()
    Main API function for Service Graph initiation
    Bounded to POST HTTP verb

topology()
    Provide internal topology description

class escape.service.sas_API.ServiceLayerAPI (standalone=False, **kwargs)
    Bases: escape.util.api.AbstractAPI

    Entry point for Service Adaptation Sublayer.

    Maintain the contact with other UNIFY layers.

    Implement the U - S1 reference point.

    _core_name = 'service'
    LAYER_ID = 'ESCAPE-service'
    dependencies = ('orchestration',)
    __init__(standalone=False, **kwargs)
```

**See also:***AbstractAPI.\_\_init\_\_()***initialize()****See also:***AbstractAPI.initialize()***shutdown(event)****See also:***AbstractAPI.shutdown()***\_initiate\_rest\_api()**

Initialize and set up REST API in a different thread.

**Returns** None**\_initiate\_gui()**

Initiate and set up GUI.

**\_handle\_SGMappingFinishedEvent(event)**Handle SGMappingFinishedEvent and proceed with *NFFG* instantiation.**Parameters** **event** (*SGMappingFinishedEvent*) – event object**Returns** None**api\_sas\_sg\_request(\*args, \*\*kwargs)**

Initiate service graph in a cooperative micro-task.

**Parameters** **service\_nffg** (*NFFG*) – service graph instance**Returns** None**api\_sas\_sg\_request\_delayed(\*args, \*\*kwargs)**

Initiate service graph in a cooperative micro-task.

**Parameters** `service_nffg` (*NFFG*) – service graph instance  
**Returns** None

**api\_sas\_get\_topology()**  
 Return with the topology description.  
**Returns** topology description requested from the layer's Virtualizer  
**Return type** *NFFG*

**get\_result(*id*)**  
 Return the state of a request given by *id*.  
**Parameters** `id` (*str or int*) – request id  
**Returns** state  
**Return type** `str` (<https://docs.python.org/2.7/library/functions.html#str>)

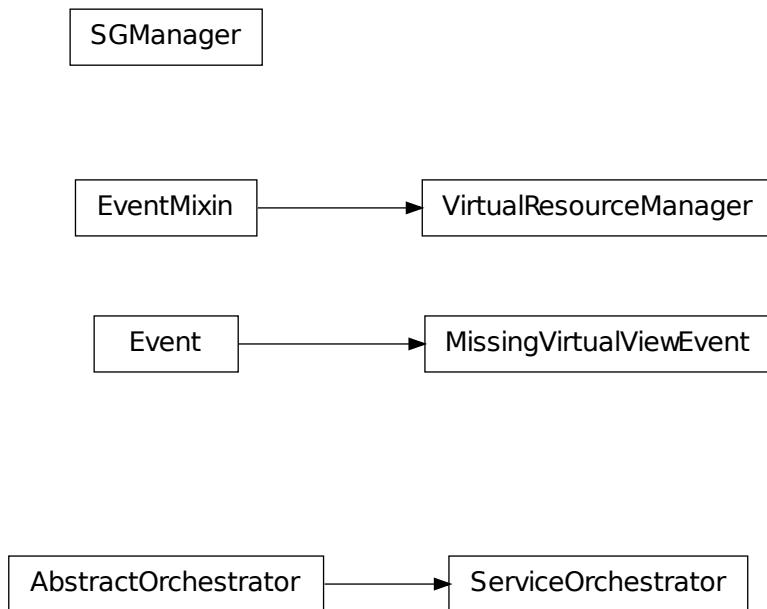
**\_instantiate\_NFFG(*nffg*)**  
 Send NFFG to Resource Orchestration Sublayer in an implementation-specific way.  
 General function which is used from microtask and Python thread also.  
**Parameters** `nffg` (*NFFG*) – mapped Service Graph  
**Returns** None

**\_handle\_MissingVirtualViewEvent(*event*)**  
 Request virtual resource info from Orchestration layer (UNIFY SI - Or API).  
 Invoked when a MissingVirtualViewEvent raised.  
 Service layer is identified with the sid value automatically.  
**Parameters** `event` (*MissingVirtualViewEvent*) – event object  
**Returns** None

**\_handle\_VirtResInfoEvent(*event*)**  
 Save requested virtual resource info as an *AbstractVirtualizer*.  
**Parameters** `event` (*VirtResInfoEvent*) – event object  
**Returns** None

**\_handle\_InstantiationFinishedEvent(*event*)**  
**\_ServiceLayerAPI\_proceed\_sg\_request(*service\_nffg*)**  
 Initiate a Service Graph (UNIFY U-SI API).  
**Parameters** `service_nffg` (*NFFG*) – service graph instance  
**Returns** None

**sas\_orchestration.py module** Contains classes relevant to Service Adaptation Sublayer functionality.



*ServiceOrchestrator* orchestrates SG mapping and centralize layer logic.

*SGManager* stores and handles Service Graphs.

*MissingVirtualViewEvent* can signal missing virtual info.

*VirtualResourceManager* contains the functionality tided to the layer's virtual view and virtual resources.

**Module contents** Contains classes relevant to Service Adaptation Sublayer functionality.

**class escape.service.sas\_orchestration.MissingVirtualViewEvent**

Bases: `pox.lib.revent.revent.Event`

Event for signaling missing virtual resource view

**class escape.service.sas\_orchestration.ServiceOrchestrator(layer\_API)**

Bases: `escape.util.mapping.AbstractOrchestrator`

Main class for the actual Service Graph processing.

**DEFAULT\_MAPPER**

alias of `ServiceGraphMapper`

**\_\_init\_\_(layer\_API)**

Initialize main Service Layer components.

**Parameters** `layer_API` (`ServiceLayerAPI`) – layer API instance

**Returns** None

**initiate\_service\_graph(sg)**

Main function for initiating Service Graphs.

**Parameters** `sg` (`NFFG`) – service graph stored in NFFG instance

**Returns** NF-FG description

**Return type** `NFFG`

```
class escape.service.sas_orchestration.SGManager
Bases: object (https://docs.python.org/2.7/library/functions.html#object)
Store, handle and organize Service Graphs.
Currently it just stores SGs in one central place.

__init__()
Init.

save(sg)
Save SG in a dict.

Parameters sg (NFFG) – Service Graph
Returns computed id of given Service Graph
Return type int (https://docs.python.org/2.7/library/functions.html#int)

get(graph_id)
Return service graph with given id.

Parameters graph_id (int (https://docs.python.org/2.7/library/functions.html#int)) –
graph ID
Returns stored Service Graph
Return type NFFG

_generate_id(sg)
Try to generate a unique id for SG.

Parameters sg (NFFG) – SG

class escape.service.sas_orchestration.VirtualResourceManager
Bases: pox.lib.revent.revent.EventMixin
Support Service Graph mapping, follow the used virtual resources according to the Service Graph(s) in effect.

Handles object derived from :class:`AbstractVirtualizer` and requested from lower layer.

_eventMixin_events = set([<class 'escape.service.sas_orchestration.MissingVirtualViewEvent'>])

__init__()
Initialize virtual resource manager.

Returns None

virtual_view
Return resource info of actual layer as an NFFG instance.
If it isn't exist requires it from Orchestration layer.

Returns resource info as a Virtualizer
Return type AbstractVirtualizer
```

### The `orchestration.py` main module

Basic POX module for ESCAPE Resource Orchestration Sublayer (ROS)

Initiate appropriate API class which implements Sl-Or reference point

Follows POX module conventions

`orchestration._start_layer(event)`

Initiate and run Orchestration module.

**Parameters** **event** (`GoingUpEvent`) – POX's going up event

**Returns** None

`orchestration.launch(nffg_file=''', standalone=False, agent=False, rosapi=False, cfor=False)`  
Launch function called by POX core when core is up.

**Parameters**

- **nffg\_file** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – Path of the NF-FG graph (optional)
- **standalone** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – Run layer without dependency checking (optional)
- **agent** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – start a REST API and act like an agent
- **agent** – start a REST API for the Cf-Or interface

**Returns** None

## Orchestration related classes

`escape.orchest` package Subpackage for classes related to UNIFY's Resource Orchestration Sublayer (ROS)

### Submodules

`nfib_mgmt.py` module Contains the class for managing NFIB.

#### NFIBManager

`NFIBManager` manages the handling of Network Function Information Base.

**Module contents** Contains the class for managing NFIB.

`class escape.orchest.nfib_mgmt.NFIBManager`  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Manage the handling of Network Function Information Base.

Use neo4j implementation for storing and querying NFs and NF decompositions.

`__init__()`  
Init.

`addNode(node)`  
Add new node to the DB.

**Parameters** `node` (*dict* (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – node to be added to the DB

**Returns** success of addition

**Return type** Boolean

`addClickNF(nf)`  
Add new click-based NF to the DB

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – nf to be added to the DB

**Returns** success of addition

**Return type** Boolean

**addVMNF** (`nf`)

**static clickCompile** (`nf`)

Compile source of the click-based NF

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – the click-based NF

**Returns** success of compilation

**Return type** Boolean

**removeNF** (`nf_id`)

Remove an NF and all its decompositions from the DB.

**Parameters** `nf_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the NF to be removed from the DB

**Returns** success of removal

**Return type** Boolean

**updateNF** (`nf`)

Update the information of a NF.

**Parameters** `nf` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – the information for the NF to be updated

**Returns** success of the update

**Return type** Boolean

**getNF** (`nf_id`)

Get the information for the NF with id equal to nf\_id.

**Parameters** `nf_id` ([string](https://docs.python.org/2.7/library/string.html#module-string)) – the id of the NF to get the information for

**Returns** the information of NF with id equal to nf\_id

**Return type** [dict](https://docs.python.org/2.7/library/stdtypes.html#dict) ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict))

**addRelationship** (`relationship`)

Add relationship between two existing nodes

**Parameters** `relationship` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – relationship to be added between two nodes

**Returns** success of the addition

**Return type** Boolean

**removeRelationship** (`relationship`)

Remove the relationship between two nodes in the DB.

**Parameters** `relationship` ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – the relationship to be removed

**Returns** the success of the removal

**Return type** Boolean

**addDecomp** (`nf_id, decomp_id, decomp`)

Add new decomposition for a high-level NF.

**Parameters**

- **nf\_id** (*string* (<https://docs.python.org/2.7/library/string.html#module-string>)) – the id of the NF for which a decomposition is added
- **decomp\_id** (*string* (<https://docs.python.org/2.7/library/string.html#module-string>)) – the id of the new decomposition
- **decomp** (*Networkx.DiGraph*) – the decomposition to be added to the DB

**Returns** success of the addition

**Return type** Boolean

**removeDecomp** (*decomp\_id*)

Remove a decomposition from the DB.

**Parameters** **decomp\_id** (*string* (<https://docs.python.org/2.7/library/string.html#module-string>)) – the id of the decomposition to be removed from the DB

**Returns** the success of the removal

**Return type** Boolean

**getSingleDecomp** (*decomp\_id*)

Get a decomposition with id *decomp\_id*.

: param *decomp\_id*: the id of the decomposition to be returned : type *decomp\_id*: str : return: decomposition with id equal to *decomp\_id* : rtype: tuple of networkx.DiGraph and Relationships

**getDecomps** (*nffg*)

Get all decompositions for a given *nffg*.

: param *nffg*: the *nffg* for which the decompositions should be returned : type *nffg*: nffg : return: all the decompositions for the given *nffg* : rtype: dict

**removeGraphDB** ()

Remove all nodes and relationships from the DB.

**Returns** None

**initialize** ()

Initialize NFIB with test data.

**\_NFIBManager\_\_initialize** ()

Initialize NFIB with test data.

**\_NFIBManager\_\_suppress\_neo4j\_logging** (*level=None*)

Suppress annoying and detailed logging of *py2neo* and *httpstream* packages.

**Parameters** **level** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – level of logging (default: WARNING)

**Returns** None

**policy\_enforcement.py module** Contains functionality related to policy enforcement.

PolicyEnforcementMetaClass

PolicyEnforcementError

PolicyEnforcement

*PolicyEnforcementError* represents a violation during the policy checking process.

*PolicyEnforcementMetaClass* contains the main general logic which handles the Virtualizers and enforce policies.

*PolicyEnforcement* implements the actual enforcement logic.

**Module contents** Contains functionality related to policy enforcement.

**exception** escape.orchest.policy\_enforcement.**PolicyEnforcementError**

Bases: `exceptions.RuntimeError` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.RuntimeError>)

Exception class to signal policy enforcement error.

**class** escape.orchest.policy\_enforcement.**PolicyEnforcementMetaClass**

Bases: `type` (<https://docs.python.org/2.7/library/functions.html#type>)

Meta class for handling policy enforcement in the context of classes inherited from *AbstractVirtualizer*.

If the *PolicyEnforcement* class contains a function which name matches one in the actual Virtualizer then PolicyEnforcement's function will be called first.

**Warning:** Therefore the function names must be identical!

---

**Note:** If policy checking fails a *PolicyEnforcementError* should be raised and handled in a higher layer..

---

To use policy checking set the following class attribute:

`__metaclass__ = PolicyEnforcementMetaClass`

**static \_\_new\_\_(mcs, name, bases, attrs)**

Magic function called before subordinated class even created

#### Parameters

- **name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – given class name

- **bases** ([tuple](https://docs.python.org/2.7/library/functions.html#tuple)) – bases of the class
- **attrs** ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – given attributes

**Returns** inferred class instance

**Return type** *AbstractVirtualizer*

**classmethod** **get\_wrapper** (*mcs, orig\_func, hooks*)

Return a decorator function which do the policy enforcement check.

**Parameters**

- **orig\_func** (*func*) – original function
- **hooks** ([tuple](https://docs.python.org/2.7/library/functions.html#tuple)) ([tuple](https://docs.python.org/2.7/library/functions.html#tuple)) – tuple of pre and post checking functions

**Raise** PolicyEnforcementError

**Returns** decorator function

**Return type** func

**class** escape.orchest.policy\_enforcement.**PolicyEnforcement**

Bases: [object](https://docs.python.org/2.7/library/functions.html#object) (<https://docs.python.org/2.7/library/functions.html#object>)

Proxy class for policy checking.

Contains the policy checking function.

Binding is based on function name (checking function have to exist in this class and its name have to stand for the *pre\_* or *post\_* prefix and the name of the checked function).

**Warning:** Every PRE policy checking function is classmethod and need to have two parameter for nameless (args) and named(kwparams) params:

Example:

```
def pre_sanity_check (cls, args, kwargs):
```

**Warning:** Every POST policy checking function is classmethod and need to have three parameter for nameless (args), named (kwparams) params and return value:

Example:

```
def post_sanity_check (cls, args, kwargs, ret_value):
```

**Note:** The first element of args is the supervised Virtualizer ('self' param in the original function)

**\_\_init\_\_()**

Init

**classmethod** **pre\_sanity\_check** (*args, kwargs*)

Implements the the sanity check before virtualizer's sanity check is called.

**Parameters**

- **args** ([tuple](https://docs.python.org/2.7/library/functions.html#tuple)) ([tuple](https://docs.python.org/2.7/library/functions.html#tuple)) – original nameless arguments
- **kwargs** ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) ([dict](https://docs.python.org/2.7/library/stdtypes.html#dict)) – original named arguments

**Returns** None

**classmethod post\_sanity\_check (args, kwargs, ret\_value)**

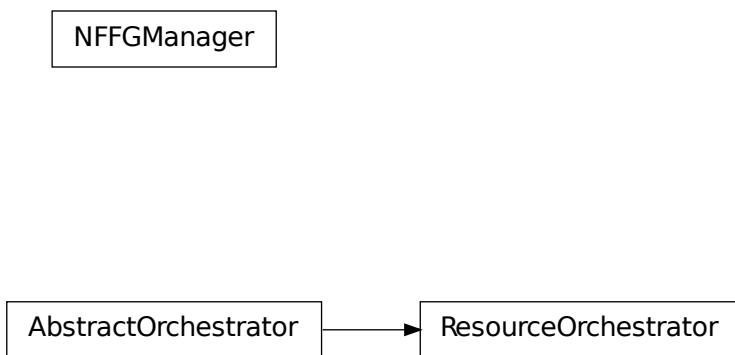
Implements the the sanity check after virtualizer's sanity check is called.

**Parameters**

- **args** (*tuple* (<https://docs.python.org/2.7/library/functions.html#tuple>)) – original nameless arguments
- **kwargs** (*dict* (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – original named arguments
- **ret\_value** – return value of Virtualizer's policy check function

**Returns** None

**ros\_orchestration.py module** Contains classes relevant to Resource Orchestration Sublayer functionality.



*ResourceOrchestrator* orchestrates *NFFG* mapping and centralize layer logic.

*NFFGManager* stores and handles Network Function Forwarding Graphs.

**Module contents** Contains classes relevant to Resource Orchestration Sublayer functionality.

**class escape.orchest.ros\_orchestration.ResourceOrchestrator (layer\_API)**  
Bases: *escape.util.mapping.AbstractOrchestrator*

Main class for the handling of the ROS-level mapping functions.

**DEFAULT\_MAPPER**

alias of *ResourceOrchestrationMapper*

**\_\_init\_\_ (layer\_API)**

Initialize main Resource Orchestration Layer components.

**Parameters** **layer\_API** (*ResourceOrchestrationAPI*) – layer API instance

**Returns** None

**instantiate\_nffg (nffg)**

Main API function for NF-FG instantiation.

**Parameters** **nffg** (*NFFG*) – NFFG instance

**Returns** mapped NFFG instance

**Return type** *NFFG*

```
class escape.orchest.ros_orchestration.NFFGManager
Bases: object (https://docs.python.org/2.7/library/functions.html#object)
Store, handle and organize Network Function Forwarding Graphs.

__init__()
Init.

save (nffg)
Save NF-FG in a dict.

    Parameters nffg (NFFG) – Network Function Forwarding Graph

    Returns generated ID of given NF-FG

    Return type int (https://docs.python.org/2.7/library/functions.html#int)

_generate_id (nffg)
Try to generate a unique id for NFFG.

    Parameters nffg (NFFG) – NFFG

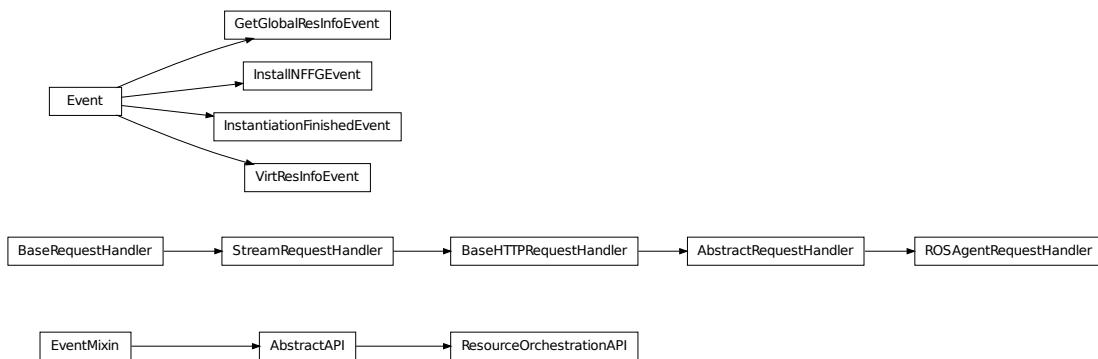
get (nffg_id)
Return NF-FG with given id.

    Parameters nffg_id (int (https://docs.python.org/2.7/library/functions.html#int)) – ID of NF-FG

    Returns NF-FG instance

    Return type NFFG
```

**ros\_API.py module** Implements the platform and POX dependent logic for the Resource Orchestration Sublayer.



*InstallNFFGEvent* can send mapped NF-FG to the lower layer.

*VirtResInfoEvent* can send back virtual resource info requested from upper layer.

*GetGlobalResInfoEvent* can request global resource info from lower layer.

*InstantiationFinishedEvent* can signal info about NFFG instantiation.

*ROSAgentRequestHandler* implements the REST-API functions for agent mode.

*ResourceOrchestrationAPI* represents the ROS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Resource Orchestration Sublayer.

```
class escape.orchest.ros_API.InstallNFFGEvent (mapped_nffg)
Bases: pox.lib.revent.revent.Event

Event for passing mapped NFFG to Controller Adaptation Sublayer.
```

```
__init__(mapped_nffg)
    Init

    Parameters mapped_nffg (NFFG) – NF-FG graph need to be installed

class escape.orchest.ros_API.VirtResInfoEvent (virtualizer)
    Bases: pox.lib.revent.revent.Event

    Event for sending back requested Virtual view an a specific Virtualizer.

__init__(virtualizer)
    Init

    Parameters virtualizer (AbstractVirtualizer) – virtual resource info

class escape.orchest.ros_API.GetGlobalResInfoEvent
    Bases: pox.lib.revent.revent.Event

    Event for requesting DomainVirtualizer from CAS.

class escape.orchest.ros_API.InstantiationFinishedEvent (id, result, error=None)
    Bases: pox.lib.revent.revent.Event

    Event for signalling end of mapping process finished with success.

__init__(id, result, error=None)

class escape.orchest.ros_API.CfOrRequestHandler (request, client_address, server)
    Bases: escape.util.api.AbstractRequestHandler

    Request Handler for the Cf-OR interface.
```

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually: use relevant helper function of core object: *callLater/raiseLater* or use *schedule\_as\_coop\_task* decorator defined in util.misc on the called function.

Contains handler functions for REST-API.

```
request_perm = {'POST': ('ping', 'get_config', 'edit_config'), 'GET': ('ping', 'version', 'operations', 'get_config')}
bounded_layer = 'orchestration'
static_prefix = 'cfor'

log = <logging.Logger object at 0x5e85550>
rpc_mapper = {'edit-config': 'edit_config', 'get-config': 'get_config'}

__init__(request, client_address, server)
    Init.

get_config()
    Response configuration.

edit_config()
    Receive configuration and initiate orchestration.

class escape.orchest.ros_API.ROSAgentRequestHandler (request, client_address, server)
    Bases: escape.util.api.AbstractRequestHandler

    Request Handler for agent behaviour in Resource Orchestration SubLayer.
```

**Warning:** This class is out of the context of the recoco's co-operative thread context! While you don't need to worry much about synchronization between recoco tasks, you do need to think about synchronization between recoco task and normal threads. Synchronisation is needed to take care manually: use relevant helper function of core object: *callLater/raiseLater* or use *schedule\_as\_coop\_task* decorator defined in util.misc on the called function.

Contains handler functions for REST-API.

```
request_perm = {'POST': ('ping', 'get_config', 'edit_config'), 'GET': ('ping', 'version', 'operations', 'get_config')}
```

```
bounded_layer = 'orchestration'
```

```
static_prefix = 'escape'
```

```
log = <logging.Logger object at 0x5e85b10>
```

```
rpc_mapper = {'edit-config': 'edit_config', 'get-config': 'get_config'}
```

```
__init__(request, client_address, server)  
    Init.
```

```
get_config()
```

Response configuration.

```
edit_config()
```

Receive configuration and initiate orchestration.

```
_update_REMOTE_ESCAPE_domain(nffg_part)
```

Update domain descriptor of infra: REMOTE -> INTERNAL

**Parameters** **nffg\_part** ([NFFG](#)) – NF-FG need to be updated

**Returns** updated NFFG

**Return type** [NFFG](#)

```
class escape.orchest.ros_API.ResourceOrchestrationAPI(standalone=False,  
                                                 **kwargs)
```

Bases: [escape.util.api.AbstractAPI](#)

Entry point for Resource Orchestration Sublayer (ROS).

Maintain the contact with other UNIFY layers.

Implement the SI - Or reference point.

```
_core_name = 'orchestration'
```

```
dependencies = ('adaptation',)
```

```
__init__(standalone=False, **kwargs)
```

**See also:**

[AbstractAPI.\\_\\_init\\_\\_\(\)](#)

**initialize()**

**See also:**

[AbstractAPI.initialize\(\)](#)

**shutdown(event)**

**See also:**

[AbstractAPI.shutdown\(\)](#)

```
_initiate_ros_api()
```

Initialize and setup REST API in a different thread.

If agent\_mod is set rewrite the received NFFG domain from REMOTE to INTERNAL.

**Returns** None

```
_initiate_cfor_api()
    Initialize and setup REST API in a different thread.

    Returns None

_handle_NFFGMappingFinishedEvent (event)
    Handle NFFGMappingFinishedEvent and proceed with NFFG installation.

    Parameters event (NFFGMappingFinishedEvent) – event object

    Returns None

api_ros_get_config()
    Implementation of REST-API RPC: get-config.

    Returns dump of global view (DoV)

    Return type str (https://docs.python.org/2.7/library/functions.html#str)

api_ros_edit_config (nffg)
    Implementation of REST-API RPC: edit-config

    Parameters nffg (NFFG) – NFFG need to deploy

api_cfor_get_config()
    Implementation of Cf-Or REST-API RPC: get-config.

    Returns dump of a single BiSBiS view based on DoV

    Return type str (https://docs.python.org/2.7/library/functions.html#str)

api_cfor_edit_config (nffg)
    Implementation of Cf-Or REST-API RPC: edit-config

    Parameters nffg (NFFG) – NFFG need to deploy

_handle_InstantiateNFFGEVENT (event)
    Instantiate given NF-FG (UNIFY SI - Or API).

    Parameters event (InstantiateNFFGEVENT) – event object contains NF-FG

    Returns None

_install_NFFG (mapped_nffg)
    Send mapped NFFG to Controller Adaptation Sublayer in an implementation-specific way.

    General function which is used from microtask and Python thread also.

    Parameters mapped_nffg (NFFG) – mapped NF-FG

    Returns None

_handle_GetVirtResInfoEvent (event)
    Generate virtual resource info and send back to SAS.

    Parameters event (GetVirtResInfoEvent) – event object contains service layer id

    Returns None

_handle_MissingGlobalViewEvent (event)
    Request Global infrastructure View from CAS (UNIFY Or - CA API).

    Invoked when a MissingGlobalViewEvent raised.

    Parameters event (MissingGlobalViewEvent) – event object

    Returns None

_handle_GlobalResInfoEvent (event)
    Save requested Global Infrastructure View as the DomainVirtualizer.

    Parameters event (GlobalResInfoEvent) – event object contains resource info

    Returns None
```

**\_handle\_InstallationFinishedEvent (event)**  
Get information from NFFG installation process.

**Parameters** `event` (`InstallationFinishedEvent`) – event object info

**Returns** None

**\_ResourceOrchestrationAPI\_\_proceed\_instantiation (\*args, \*\*kwargs)**  
Helper function to instantiate the NFFG mapping from different source.

**Parameters** `nffg` (`NFFG`) – pre-mapped service request

**Returns** None

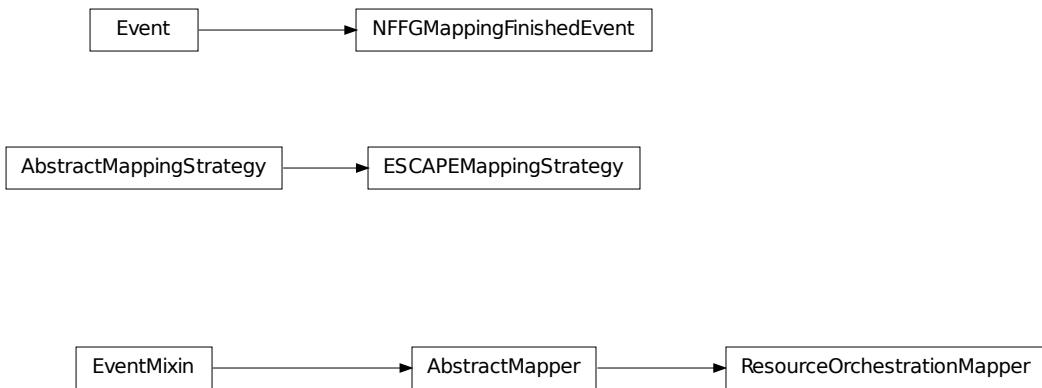
**\_ResourceOrchestrationAPI\_\_update\_nffg (nffg\_part)**  
Update domain descriptor of infra: REMOTE -> INTERNAL

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be updated

**Returns** updated NFFG

**Return type** `NFFG`

**ros\_mapping.py module** Contains classes which implement `NFFG` mapping functionality.



`ESCAPEMappingStrategy` implements a default `NFFG` mapping algorithm of ESCAPEv2.

`NFFGMappingFinishedEvent` can signal the state of NFFG mapping.

`ResourceOrchestrationMapper` perform the supplementary tasks for `NFFG` mapping.

**Module contents** Contains classes which implement `NFFG` mapping functionality.

**class escape.orchest.ros\_mapping.ESCAPEMappingStrategy**  
Bases: `escape.util.mapping.AbstractMappingStrategy`

Implement a strategy to map initial `NFFG` into extended `NFFG`.

**\_\_init\_\_()**  
Init

**classmethod map (graph, resource)**  
Default mapping algorithm of ESCAPEv2.

**Parameters**

- **graph** (`NFFG`) – Network Function forwarding Graph
- **resource** (`NFFG`) – global virtual resource info

**Returns** mapped Network Function Forwarding Graph

**Return type** `NFFG`

```
class escape.orchest.ros_mapping.NFFGMappingFinishedEvent (nffg)
```

Bases: `pox.lib.revent.revent.Event`

Event for signaling the end of NF-FG mapping.

```
__init__(nffg)
```

Init.

**Parameters** `nffg` (`NFFG`) – NF-FG need to be installed

```
class escape.orchest.ros_mapping.ResourceOrchestrationMapper (strategy=None)
```

Bases: `escape.util.mapping.AbstractMapper`

Helper class for mapping NF-FG on global virtual view.

```
_eventMixin_events = set([<class 'escape.orchest.ros_mapping.NFFGMappingFinishedEvent'>])
```

`DEFAULT_STRATEGY`

alias of `ESCAPEMappingStrategy`

```
__init__(strategy=None)
```

Init Resource Orchestrator mapper.

**Returns** None

```
_perform_mapping(input_graph, resource_view)
```

Orchestrates mapping of given NF-FG on given global resource.

**Parameters**

- `input_graph` (`NFFG`) – Network Function Forwarding Graph
- `resource_view` (`DomainVirtualizer`) – global resource view

**Returns** mapped Network Function Forwarding Graph

**Return type** `NFFG`

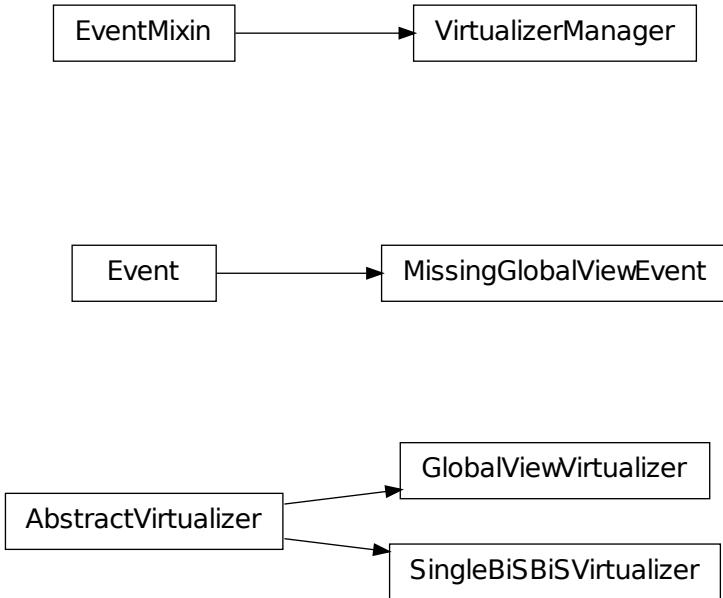
```
_mapping_finished(nffg)
```

Called from a separate thread when the mapping process is finished.

**Parameters** `nffg` (`NFFG`) – mapped NF-FG

**Returns** None

***virtualization\_mgmt.py* module** Contains components relevant to virtualization of resources and views.



`MissingGlobalViewEvent` can signal missing global view.

`AbstractVirtualizer` contains the central logic of Virtualizers.

`GlobalViewVirtualizer` implements a non-filtering/non-virtualizing logic.

`SingleBiSBiSVirtualizer` implement the default, 1-Bis-Bis virtualization logic of the Resource Orchestration Sublayer.

`VirtualizerManager` stores and handles the virtualizers.

**Module contents** Contains components relevant to virtualization of resources and views.

**class escape.orchest.virtualization\_mgmt.MissingGlobalViewEvent**  
Bases: `pox.lib.revent.revent.Event`

Event for signaling missing global resource view.

**class escape.orchest.virtualization\_mgmt.AbstractVirtualizer(id)**  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Abstract class for actual Virtualizers.

Follows the Proxy design pattern.

**\_\_metaclass\_\_**  
alias of `PolicyEnforcementMetaClass`

**\_\_init\_\_(id)**  
Init.

**Parameters** `id` – id of the assigned entity

**Type** `id`: str

**\_\_str\_\_()**

**\_\_repr\_\_()**

**get\_resource\_info()**

Hides object's mechanism and return with a resource object derived from [NFFG](#).

**Warning:** Derived class have to override this function

**Raise** `NotImplementedError`

**Returns** resource info

**Return type** [NFFG](#)

**sanity\_check(\*args, \*\*kwargs)**

Place-holder for sanity check which implemented in `PolicyEnforcement`.

**Parameters** `nffg` ([NFFG](#)) – NFFG instance

**Returns** None

```
class escape.orchest.virtualization_mgmt.GlobalViewVirtualizer(global_view,
                                                               id)
```

Bases: `escape.orchest.virtualization_mgmt.AbstractVirtualizer`

Virtualizer class for experimenting and testing.

No filtering, just offer the whole global resource view.

**\_\_init\_\_(global\_view, id)**

Init.

**Parameters**

- **global\_view** ([DomainVirtualizer](#)) – virtualizer instance represents the global view
- **id** – id of the assigned entity

**Type** id: str

**get\_resource\_info()**

Hides object's mechanism and return with a resource object derived from [NFFG](#).

**Returns** Virtual resource info as an NFFG

**Return type** [NFFG](#)

```
class escape.orchest.virtualization_mgmt.SingleBiSBiSVirtualizer(global_view,
                                                               id)
```

Bases: `escape.orchest.virtualization_mgmt.AbstractVirtualizer`

Actual Virtualizer class for ESCAPEv2.

Default virtualizer class which offer the trivial one BisBis view.

**\_\_init\_\_(global\_view, id)**

Init.

**Parameters**

- **global\_view** ([DomainVirtualizer](#)) – virtualizer instance represents the global view
- **id** – id of the assigned entity

**Type** id: str

**get\_resource\_info()**

Hides object's mechanism and return with a resource object derived from [NFFG](#).

**Returns** Virtual resource info as an NFFG

**Return type** [NFFG](#)

**\_generate\_one\_bisbis()**

Generate trivial virtual topology a.k.a 1 BisBis.

**Returns** 1 Bisbis topo

**Return type** *NFFG*

**class escape.orchest.virtualization\_mgmt.VirtualizerManager**

Bases: `pox.lib.revent.revent.EventMixin`

Store, handle and organize instances of derived classes of `AbstractVirtualizer`.

**\_eventMixin\_events = set([<class ‘escape.orchest.virtualization\_mgmt.MissingGlobalViewEvent’>])**

**TYPES** = {‘SINGLE’: <class ‘escape.orchest.virtualization\_mgmt.SingleBiSBiSVirtualizer’>, ‘GLOBAL’: <class ‘escape.orchest.virtualization\_mgmt.GlobalViewVirtualizer’>}

**\_init\_\_()**

Initialize virtualizer manager.

**Returns** None

**dov**

Getter method for the DomainVirtualizer.

Request DoV from Adaptation if it hasn’t set yet.

Use: `virtualizerManager.dov`.

**Returns** Domain Virtualizer (DoV)

**Return type** *DomainVirtualizer*

**get\_virtual\_view(virtualizer\_id, type=None, cls=None)**

Return the Virtual View as a derived class of `AbstractVirtualizer`.

**Parameters**

- **virtualizer\_id** (*int or str*) – unique id of the requested Virtual view
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – type of the Virtualizer predefined in this class
- **cls** (*AbstractVirtualizer*) – specific Virtualizer class if type is not given

**Returns** virtual view

**Return type** *AbstractVirtualizer*

**\_generate\_single\_view(id)**

Generate a Single BiSBiS virtualizer, store and return with it.

**Parameters** **id** (*int or str*) – unique virtualizer id

**Returns** generated Virtualizer

**Return type** *SingleBiSBiSVirtualizer*

**\_generate\_global\_view(id)**

Generate a Global View virtualizer, store and return with it.

**Parameters** **id** (*int or str*) – unique virtualizer id

**Returns** generated Virtualizer

**Return type** *GlobalViewVirtualizer*

**The *adaptation.py* main module**

Basic POX module for ESCAPE Controller Adaptation Sublayer (CAS)

Initiate appropriate API class which implements Or-Ca reference point

Follows POX module conventions

`adaptation._start_layer(event)`  
Initiate and run Adaptation module.

**Parameters** `event` (*GoingUpEvent*) – POX’s going up event

**Returns** None

`adaptation.launch(mapped_nffg='', with_infr=False, standalone=False)`  
Launch function called by POX core when core is up.

**Parameters**

- `mapped_nffg` (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – Path of the mapped NF-FG graph (optional)
- `with_infr` (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – Set Infrastructure as a dependency
- `standalone` (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – Run layer without dependency checking (optional)

**Returns** None

### Adaptation related classes

`escape.adapt` package Sublayer for classes related to UNIFY’s Controller Adaptation Sublayer (CAS)

#### Submodules

`adaptation.py` module Contains classes relevant to the main adaptation function of the Controller Adaptation Sublayer.

DomainResourceManager

ControllerAdapter

ComponentConfigurator

AbstractVirtualizer → DomainVirtualizer

*ComponentConfigurator* creates, initializes, stores and manages different adaptation components, i.e. derived classes of *AbstractDomainManager* and *AbstractESCAPEAdapter*.

*ControllerAdapter* implements the centralized functionality of high-level adaptation and installation of *NFFG*.

*DomainVirtualizer* implements the standard virtualization/generalization logic of the Resource Orchestration Sublayer.

*DomainResourceManager* stores and manages the global Virtualizer.

**Module contents** Contains classes relevant to the main adaptation function of the Controller Adaptation Sublayer

**class** `escape.adapt.adaptation.ComponentConfigurator (ca, lazy_load=True)`

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Initialize, configure and store DomainManager objects. Use global config to create managers and adapters.

Follows Component Configurator design pattern.

**\_\_init\_\_ (ca, lazy\_load=True)**

For domain adapters the configurator checks the CONFIG first.

**Warning:** Adapter classes must be subclass of *AbstractESCAPEAdapter*

---

**Note:** Arbitrary domain adapters is searched in `escape.adapt.domain_adapters`

---

### Parameters

- **ca** (*ControllerAdapter*) – ControllerAdapter instance
- **lazy\_load** (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – load adapters only at first reference (default: True)

**get\_mgr (domain\_name)**

Return the DomainManager with given name and create+start if needed.

**Parameters** `domain_name (str (https://docs.python.org/2.7/library/functions.html#str))` – name of domain manager

**Returns** None

**start\_mgr (domain\_name, autostart=True)**

Create, initialize and start a DomainManager with given name and start the manager by default.

### Parameters

- **domain\_name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager
- **autostart** (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – also start the domain manager (default: True)

**Returns** domain manager

**Return type** *AbstractDomainManager*

**stop\_mgr (domain\_name)**

Stop and derefer a DomainManager with given name and remove from the repository also.

**Parameters** `domain_name (str (https://docs.python.org/2.7/library/functions.html#str))` – name of domain manager

**Returns** None

**is\_started**(*domain\_name*)

Return with the value the given domain manager is started or not.

**Parameters** *domain\_name* ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – name of domain manager

**Returns** is loaded or not

**Return type** [bool](https://docs.python.org/2.7/library/functions.html#bool) (<https://docs.python.org/2.7/library/functions.html#bool>)

**components**

Return the dict of initiated Domain managers.

**Returns** container of initiated DomainManagers

**Return type** [dict](https://docs.python.org/2.7/library/stdtypes.html#dict) (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**initiated****\_\_iter\_\_**()

Return with an iterator over the (domain\_name, DomainManager) items.

**\_\_getitem\_\_**(*item*)

Return with the DomainManager given by name: item.

**Parameters** *item* ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – component name

**Returns** component

**Return type** [AbstractDomainManager](#)

**load\_component**(*component\_name*)

Load given component (DomainAdapter/DomainManager) from config. Initiate the given component class, pass the additional attributes, register the event listeners and return with the newly created object.

**Parameters** *component\_name* ([str](https://docs.python.org/2.7/library/functions.html#str) (<https://docs.python.org/2.7/library/functions.html#str>)) – component's name

**Returns** initiated component

**Return type** [AbstractESCAPEAdapter](#) or [AbstractDomainManager](#)

**load\_default\_mgrs**()

Initiate and start default DomainManagers defined in CONFIG.

**Returns** None

**load\_internal\_mgr**()

Initiate the DomainManager for the internal domain.

**Returns** None

**clear\_initiated\_mgrs**()

Clear initiated DomainManagers based on the first received config.

**Returns** None

**stop\_initiated\_mgrs**()

Stop initiated DomainManagers.

**Returns** None

**class** `escape.adapt.adaptation.ControllerAdapter`(*layer\_API*, *with\_infr=False*)

Bases: [object](https://docs.python.org/2.7/library/functions.html#object) (<https://docs.python.org/2.7/library/functions.html#object>)

Higher-level class for [NFFG](#) adaptation between multiple domains.

**DOMAIN\_MAPPING** = {‘OPENSTACK’: ‘OPENSTACK’, ‘SDN’: ‘SDN’, ‘INTERNAL’: ‘INTERNAL’, ‘UNIVERSAL’: ‘UNIVERSAL’}

**\_\_init\_\_(layer\_API, with\_infr=False)**  
Initialize Controller adapter.

For domain components the ControllerAdapter checks the CONFIG first.

#### Parameters

- **layer\_API** (*ControllerAdaptationAPI*) – layer API instance
- **with\_infr** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – using emulated infrastructure (default: False)

**shutdown()**

Shutdown ControllerAdapter, related components and stop DomainManagers.

#### Returns

**install\_nffg(mapped\_nffg)**

Start NF-FG installation.

Process given *NFFG*, slice information self.\_\_global\_nffg on domains and invoke DomainManagers to install domain specific parts.

**Parameters** **mapped\_nffg** (*NFFG*) – mapped NF-FG instance which need to be installed

**Returns** None or internal domain NFFG part

**\_handle\_DomainChangedEvent(event)**

Handle DomainChangedEvents, process changes and store relevant information in DomainResourceManager.

**\_split\_into\_domains(nffg)**

Split given *NFFG* into separate parts self.\_\_global\_nffg on original domains.

**Warning:** Not implemented yet!

**Parameters** **nffg** (*NFFG*) – mapped NFFG object

**Returns** sliced parts as a list of (domain\_name, nffg\_part) tuples

**Return type** **list** (<https://docs.python.org/2.7/library/functions.html#list>)

**update\_dov(nffg\_part)**

Update the global view with installed Nfs/Flowrules.

**class escape.adapt.adaptation.DomainVirtualizer(domainResManager,  
global\_res=None)**

Bases: *escape.orchest.virtualization\_mgmt.AbstractVirtualizer*

Specific Virtualizer class for global domain virtualization.

Implement the same interface as *AbstractVirtualizer*

Use *NFFG* format to store the global infrastructure info.

**\_\_init\_\_(domainResManager, global\_res=None)**

Init.

#### Parameters

- **domainResManager** (*DomainResourceManager*) – domain resource manager
- **global\_res** (*NFFG*) – initial global resource (optional)

**Returns** None

**name**

**\_\_str\_\_()**

**\_\_repr\_\_()**

**get\_resource\_info()**

Return the global resource info represented this class.

**Returns** global resource info

**Return type** *NFFG*

**set\_domain\_as\_global\_view(domain, nffg)**

Set the copy of given NFFG as the global view of DoV.

**Parameters** *nffg* (*NFFG*) – NFFG instance intended to use as the global view

**Returns** None

**merge\_domain\_into\_dov(domain, nffg)**

Add a newly detected domain to DoV.

Based on the feature: escape.util.nffg.NFFGToolBox#merge\_domains

**update\_global\_view(global\_nffg)**

Update the merged Global view with the given probably modified global view.

**Parameters** *global\_nffg* (*NFFG*) – updated global view which replace the stored one

**update\_domain\_view(domain, nffg)**

Update the existing domain in the merged Global view.

**class escape.adapt.adaptation.DomainResourceManager**

Bases: *object* (<https://docs.python.org/2.7/library/functions.html#object>)

Handle and store the global resources view.

**\_\_init\_\_()**

Init.

**get\_global\_view()**

Getter for *DomainVirtualizer*.

**Returns** global infrastructure view as the Domain Virtualizer

**Return type** *DomainVirtualizer*

**update\_domain\_resource(domain, nffg)**

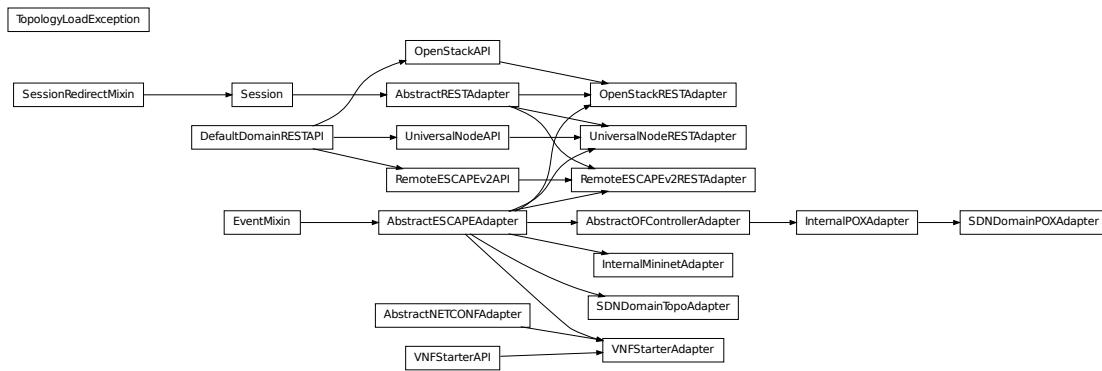
Update the global view data with the specific domain info.

**Parameters**

- **domain** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – domain name
- **nffg** (*NFFG*) – infrastructure info collected from the domain

**Returns** None

**adapters.py module** Contains Adapter classes which contains protocol and technology specific details for the connections between ESCAPEv2 and other different domains.



*InternalPOXAdapter* implements the OF controller functionality for the Mininet-based emulated topology.

*SDNDomainPOXAdapter* implements the OF controller functionality for the external SDN/OpenFlow switches.

*InternalMininetAdapter* implements Mininet related functionality transparently e.g. start/stop/clean topology built from an :any:`'NFFG'`.

*SDNDomainTopoAdapter* implements SDN topology related functions.

*VNFStarterAdapter* is a helper/wrapper class for vnf\_starter NETCONF module.

*RemoteESCAPEv2RESTAdapter* is a wrapper class for REST-based communication with an another ESCAPE instance started in agent mode.

*OpenStackRESTAdapter* is a wrapper class for OpenStack-REST-like API functions.

*UniversalNodeRESTAdapter* is a wrapper class for REST-like communication with the Universal Node domain.

**Module contents** Contains Adapter classes which contains protocol and technology specific details for the connections between ESCAPEv2 and other different domains.

#### **exception** escape.adapt.adapters.**TopologyLoadException**

Bases: `exceptions.Exception` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception>)

Exception class for topology errors.

**class** escape.adapt.adapters.**InternalPOXAdapter** (*name=None*, *address='127.0.0.1'*, *port=6653*, *keepalive=False*)

Bases: `escape.util.domain.AbstractOFControllerAdapter`

Adapter class to handle communication with internal POX OpenFlow controller.

Can be used to define a controller (based on POX) for other external domains.

**name = 'INTERNAL-POX'**

**infra\_to\_dpid = {}**

**saps = {}**

**\_\_init\_\_** (*name=None*, *address='127.0.0.1'*, *port=6653*, *keepalive=False*)

Initialize attributes, register specific connection Arbiter if needed and set up listening of OpenFlow events.

#### **Parameters**

- **name** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name used to register component into `pox.core`
- **address** (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – socket address (default: 127.0.0.1)

- **port** ([int](https://docs.python.org/2.7/library/functions.html#int)) – socket port (default: 6633)

**check\_domain\_reachable()**

Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** [bool](https://docs.python.org/2.7/library/functions.html#bool) (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource()**

Return with the topology description as an [NFFG](#).

**Returns** the emulated topology description

**Return type** [NFFG](#)

**\_handle\_ConnectionUp(event)**

Handle incoming OpenFlow connections.

**\_handle\_ConnectionDown(event)**

Handle disconnected device.

**\_identify\_ovs\_device(connection)**

Identify the representing Node of the OVS switch according to the given connection and extend the dpid-infra binding dictionary.

The discovery algorithm takes the advantage of the naming convention of Mininet for interfaces in an OVS switch e.g.: EE1, EE1-eth1, EE1-eth2, etc.

**Parameters** **connection** (`pox.openflow.of_01.Connection`) – inner Connection class of POX

**Returns** None

```
class escape.adapt.adapters.SDNDomainPOXAdapter(name=None, address='0.0.0.0',
                                                port=6653, keepalive=False)
```

Bases: [escape.adapt.adapters.InternalPOXAdapter](#)

Adapter class to handle communication with external SDN switches.

**name = 'SDN-POX'**

**infra\_to\_dpid = {‘MT2’: 365441792307142, ‘MT1’: 365441792306724}**

**dpid\_to\_infra = {365441792306724: ‘MT1’, 365441792307142: ‘MT2’}**

**\_\_init\_\_(name=None, address='0.0.0.0', port=6653, keepalive=False)**

**get\_topology\_resource()**

**check\_domain\_reachable()**

```
class escape.adapt.adapters.InternalMininetAdapter(net=None)
```

Bases: [escape.util.domain.AbstractESCAPEAdapter](#)

Adapter class to handle communication with Mininet domain.

Implement VNF managing API using direct access to the `mininet.net.Mininet` object.

**\_eventMixin\_events = set([<class ‘escape.util.domain.DomainChangedEvent’>])**

**name = ‘MININET’**

**\_\_init\_\_(net=None)**

Init.

**Parameters** **net** (`ESCAPENetworkBridge`) – set pre-defined network (optional)

**get\_mn\_wrapper()**

Return the specific wrapper for `mininet.net.Mininet` object represents the emulated network.

**Returns** emulated network wrapper

**Return type** `ESCAPENetworkBridge`

**check\_domain\_reachable()**  
Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource()**  
Return with the topology description as an `NFFG`.

**Returns** the emulated topology description

**Return type** `NFFG`

**get\_agent\_connection\_params(`ee_name`)**  
Return the connection parameters for the agent of the switch given by the `switch_name`.

**Parameters** `ee_name` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the container Node

**Returns** connection params

**Return type** `dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**class escape.adapt.adapters.SDNDomainTopoAdapter(`path=None`)**  
Bases: `escape.util.domain.AbstractESCAPEAdapter`

Adapter class to return the topology description of the SDN domain.

Currently it just read the static description from file, and not discover it.

**name = ‘SDN-TOPO’**

**\_\_init\_\_(`path=None`)**

**check\_domain\_reachable()**  
Checker function for domain. Naively return True.

**Returns** the domain is detected or not

**Return type** `bool` (<https://docs.python.org/2.7/library/functions.html#bool>)

**get\_topology\_resource()**  
Return with the topology description as an `NFFG` parsed from file.

**Returns** the static topology description

**Return type** `NFFG`

**\_SDNDomainTopoAdapter\_\_init\_from\_CONFIG(`path=None`)**  
Load a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name SDN-TOPO.

**Parameters** `path` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – additional file path

**Returns** None

**class escape.adapt.adapters.VNFStarterAdapter(\*\*kwargs)**  
Bases: `escape.util.netconf.AbstractNETCONFAdapter`, `escape.util.domain.AbstractESCAPEAdapter`, `escape.util.domain.VNFStarterAPI`

This class is devoted to provide NETCONF specific functions for vnf\_starter module. Documentation is transferred from `vnf_starter.yang`.

This class is devoted to start and stop CLICK-based VNFs that will be connected to a mininet switch.

Follows the MixIn design pattern approach to support NETCONF functionality.

**RPC\_NAMESPACE = u’http://csikor.tmit.bme.hu/netconf/unify/vnf\_starter’**

```
name = 'VNFStarter'
```

```
__init__(**kwargs)
```

Init.

#### Parameters

- **server** ([str](https://docs.python.org/2.7/library/functions.html#str)) – server address
- **port** ([int](https://docs.python.org/2.7/library/functions.html#int)) – port number
- **username** ([str](https://docs.python.org/2.7/library/functions.html#str)) – username
- **password** ([str](https://docs.python.org/2.7/library/functions.html#str)) – password
- **timeout** ([int](https://docs.python.org/2.7/library/functions.html#int)) – connection timeout (default=30)

#### Returns

```
check_domain_reachable()
```

Checker function for domain polling.

**Returns** the domain is detected or not

**Return type** [bool](https://docs.python.org/2.7/library/functions.html#bool)

```
get_topology_resource()
```

Return with the topology description as an [NFFG](#).

**Returns** the emulated topology description

**Return type** [NFFG](#)

```
update_connection_params(**kwargs)
```

Update connection params.

**Returns** only updated params

**Return type** [dict](https://docs.python.org/2.7/library/stdtypes.html#dict)

```
_invoke_rpc(request_data)
```

Override parent function to catch and log exceptions gracefully.

```
initiateVNF(vnf_type, vnf_description=None, options=None)
```

This RCP will start a VNF.

0.initiate new VNF (initiate datastructure, generate unique ID)

1.set its arguments (control port, control ip, and VNF type/command)

2.returns the connection data, which from the vnf\_id is the most important

#### Parameters

- **vnf\_type** ([str](https://docs.python.org/2.7/library/functions.html#str)) – pre-defined VNF type (see in vnf\_starter/available\_vnfs)
- **vnf\_description** ([str](https://docs.python.org/2.7/library/functions.html#str)) – Click description if there are no pre-defined type
- **options** ([collections.OrderedDict](https://docs.python.org/2.7/library/collections.html#collections.OrderedDict)) (<https://docs.python.org/2.7/library/collections.html#collections.OrderedDict>) – unlimited list of additional options as name-value pairs

**Returns** RPC reply data

**Return type** [dict](https://docs.python.org/2.7/library/stdtypes.html#dict)

**Raises** RPCError, OperationError, TransportError

```
connectVNF(vnf_id, vnf_port, switch_id)
```

This RPC will practically start and connect the initiated VNF/CLICK to the switch.

0.create virtualEthernet pair(s)

1.connect either end of it (them) to the given switch(es)

This RPC is also used for reconnecting a VNF. In this case, however, if the input fields are not correctly set an error occurs

#### Parameters

- **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)
- **vnf\_port** (*str or int*) – VNF port (mandatory)
- **switch\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – switch ID (mandatory)

**Returns** Returns the connected port(s) with the corresponding switch(es).

**Raises** `RPCError`, `OperationError`, `TransportError`

#### **disconnectVNF** (*vnf\_id*, *vnf\_port*)

This RPC will disconnect the VNF(s)/CLICK(s) from the switch(es).

0.ip link set uny\_0 down

1.ip link set uny\_1 down

2.(if more ports) repeat 1. and 2. with the corresponding data

#### Parameters

- **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)
- **vnf\_port** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF port (mandatory)

**Returns** reply data

**Raises** `RPCError`, `OperationError`, `TransportError`

#### **startVNF** (*vnf\_id*)

This RPC will actually start the VNF/CLICK instance.

**Parameters** **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)

**Returns** reply data

**Raises** `RPCError`, `OperationError`, `TransportError`

#### **stopVNF** (*vnf\_id*)

This RPC will gracefully shut down the VNF/CLICK instance.

0.if disconnect() was not called before, we call it

1.delete virtual ethernet pairs

2.stop (kill) click

3.remove vnf's data from the data structure

**Parameters** **vnf\_id** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – VNF ID (mandatory)

**Returns** reply data

**Raises** `RPCError`, `OperationError`, `TransportError`

**getVNFInfo (vnf\_id=None)**

This RPC will send back all data of all VNFs that have been initiated by this NETCONF Agent. If an input of vnf\_id is set, only that VNF's data will be sent back. Most of the data this RPC replies is used for DEBUG, however 'status' is useful for indicating to upper layers whether a VNF is UP\_AND\_RUNNING.

**Parameters** **vnf\_id** ([str](https://docs.python.org/2.7/library/functions.html#str)) – VNF ID (default: list info about all VNF)

**Returns** reply data

**Raises** [RPCError](#), [OperationError](#), [TransportError](#)

**deployNF (nf\_type, nf\_ports, infra\_id, nf\_desc=None, nf\_opt=None)**

Initiate and start the given NF using the general RPC calls.

**Parameters**

- **nf\_type** ([str](https://docs.python.org/2.7/library/functions.html#str)) – pre-defined NF type (see in vnf\_starter/available\_vnfs)
- **nf\_ports** ([str or int or tuple](https://docs.python.org/2.7/library/functions.html#str)) – NF port number or list of ports (mandatory)
- **infra\_id** ([str](https://docs.python.org/2.7/library/functions.html#str)) – id of the base node (mandatory)
- **nf\_desc** ([str](https://docs.python.org/2.7/library/functions.html#str)) – Click description if there are no pre-defined type
- **nf\_opt** ([collections.OrderedDict](https://docs.python.org/2.7/library/collections.html#collections.OrderedDict)) (<https://docs.python.org/2.7/library/collections.html#collections.OrderedDict>) – unlimited list of additional options as name-value pairs

**Returns** initiated NF description parsed from RPC reply

**Return type** [dict](#) (<https://docs.python.org/2.7/library/stdtypes.html#dict>)

**removeNF (vnf\_id)**

Stop and remove the given NF using the general RPC calls.

**class escape.adapt.adapters.RemoteESCAPEv2RESTAdapter (url)**

Bases: [escape.util.domain.AbstractRESTAdapter](#), [escape.util.domain.AbstractESCAPEAdapter](#), [escape.util.domain.RemoteESCAPEv2API](#)

This class is devoted to provide REST specific functions for remote ESCAPEv2 domain.

**name = ‘ESCAPE-REST’**

**\_\_init\_\_ (url)**

Init.

**Parameters** **url** ([str](https://docs.python.org/2.7/library/functions.html#str)) – remote ESCAPEv2 RESTful API URL

**ping()**

**get\_config()**

**edit\_config (data)**

**check\_domain\_reachable()**

**get\_topology\_resource()**

**class escape.adapt.adapters.OpenStackRESTAdapter (url)**

Bases: [escape.util.domain.AbstractRESTAdapter](#), [escape.util.domain.AbstractESCAPEAdapter](#), [escape.util.domain.OpenStackAPI](#)

This class is devoted to provide REST specific functions for OpenStack domain.

**name = ‘OpenStack-REST’**

```
__init__(url)
    Init.

    Parameters url (str (https://docs.python.org/2.7/library/functions.html#str)) – OpenStack
    RESTful API URL

ping()
get_config()
edit_config(data)
check_domain_reachable()
get_topology_resource()

class escape.adapt.adapters.UniversalNodeRESTAdapter(url)
    Bases: escape.util.domain.AbstractRESTAdapter, escape.util.domain.AbstractESCAPEAdapter
    escape.util.domain.UniversalNodeAPI

This class is devoted to provide REST specific functions for UN domain.

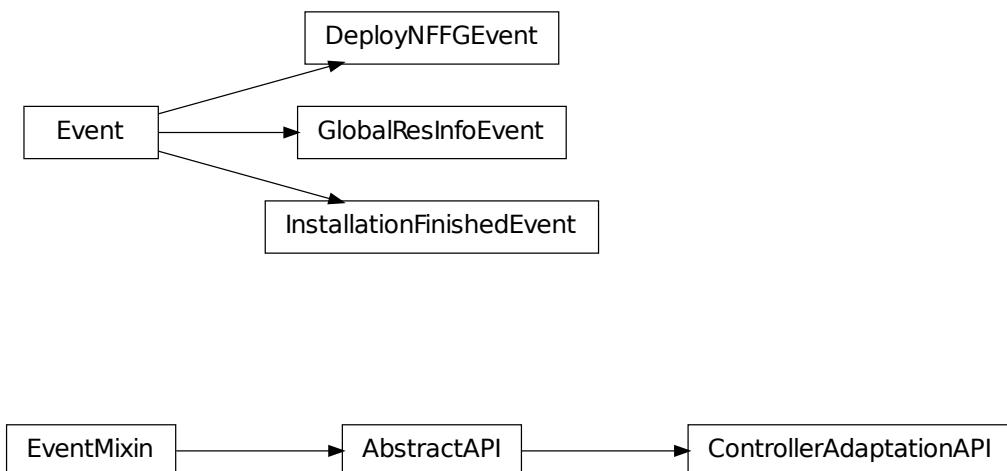
name = 'UN-REST'

__init__(url)
    Init.

    Parameters url (str (https://docs.python.org/2.7/library/functions.html#str)) – Universal
    Node RESTful API URL

ping()
get_config()
edit_config(data)
check_domain_reachable()
get_topology_resource()
```

**cas\_API.py module** Implements the platform and POX dependent logic for the Controller Adaptation Sublayer.



*GlobalResInfoEvent* can send back global resource info requested from upper layer.

*InstallationFinishedEvent* can send back status about the NFFG installation.

`DeployNFFGEvent` can send NFFG to Infrastructure layer for deploying.

`ControllerAdaptationAPI` represents the CAS layer and implement all related functionality.

**Module contents** Implements the platform and POX dependent logic for the Controller Adaptation Sublayer.

**class** `escape.adapt.cas_API.GlobalResInfoEvent (dov)`

Bases: `pox.lib.revent.revent.Event`

Event for sending back requested Global Resource View.

**\_\_init\_\_ (dov)**

Init.

**Parameters** `dov (DomainVirtualizer)` – Domain Virtualizer which handles the Global Infrastructure View.

**class** `escape.adapt.cas_API.InstallationFinishedEvent (id, result, error=None)`

Bases: `pox.lib.revent.revent.Event`

Event for signalling end of mapping process.

**\_\_init\_\_ (id, result, error=None)**

**class** `escape.adapt.cas_API.DeployNFFGEvent (nffg_part)`

Bases: `pox.lib.revent.revent.Event`

Event for passing mapped `NFFG` to internally emulated network based on Mininet for testing.

**\_\_init\_\_ (nffg\_part)**

**class** `escape.adapt.cas_API.ControllerAdaptationAPI (standalone=False, **kwargs)`

Bases: `escape.util.api.AbstractAPI`

Entry point for Controller Adaptation Sublayer (CAS).

Maintain the contact with other UNIFY layers.

Implement the Or - Ca reference point.

**\_core\_name = ‘adaptation’**

**\_\_init\_\_ (standalone=False, \*\*kwargs)**

**See also:**

`AbstractAPI.__init__()`

`initialize()`

**See also:**

`AbstractAPI.initialize()`

`shutdown (event)`

**See also:**

`AbstractAPI.shutdown()`

**\_handle\_InstallNFFGEvent (event)**

Install mapped NF-FG (UNIFY Or - Ca API).

**Parameters** `event (InstallNFFGEvent)` – event object contains mapped NF-FG

**Returns** None

**\_handle\_GetGlobalResInfoEvent (event)**

Generate global resource info and send back to ROS.

**Parameters** `event` (`GetGlobalResInfoEvent`) – event object  
**Returns** None

**\_handle\_DeployEvent** (`event`)  
 Receive processed NF-FG from domain adapter(s) and forward to Infrastructure

**Parameters** `event` (`DeployNFFGEvent`) – event object  
**Returns** None

**\_handle\_DeploymentFinishedEvent** (`event`)  
 Receive successful NF-FG deployment event and propagate upwards

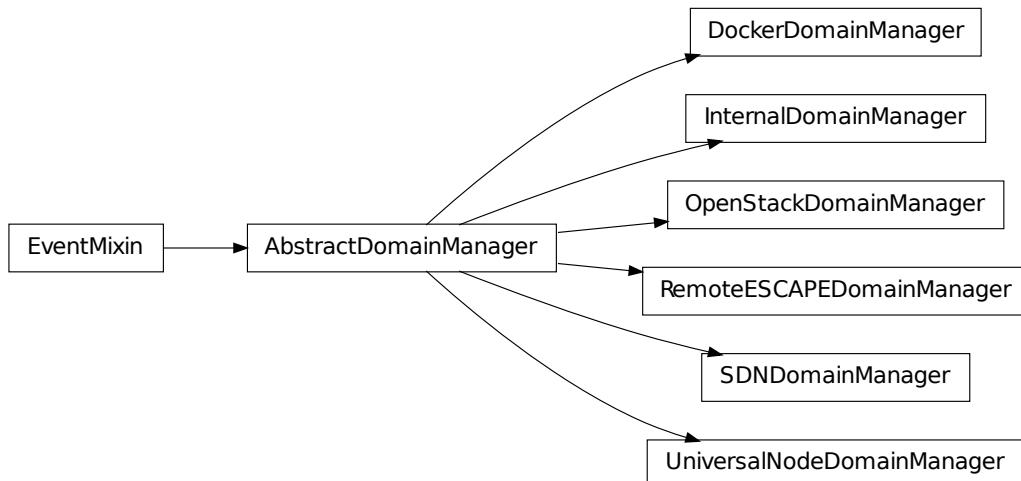
**Parameters** `event` (`DeploymentFinishedEvent`) – event object  
**Returns** None

**\_ControllerAdaptationAPI\_proceed\_installation** (\*args, \*\*kwargs)  
 Helper function to instantiate the NFFG mapping from different source.

**Parameters** `mapped_nffg` (`NFFG`) – pre-mapped service request  
**Returns** None

**managers.py module** Contains Manager classes which contains the higher-level logic for complete domain management.

Uses Adapter classes for ensuring protocol-specific connections with entities in the particular domain.



`InternalDomainManager` represent the top class for interacting with the emulated infrastructure.

`RemoteESCAPEDomainManager` ensures the connection with a different ESCAPE instance started in agent mode.

`OpenStackDomainManager` implements the related functionality for managing the OpenStack-based domain.

`UniversalNodeDomainManager` implements the related functionality for managing the domain based on the Universal Node conception.

`DockerDomainManager` is a placeholder class for managing Docker-based network entities.

`SDNDomainManager` interacts and handles legacy OpenFlow 1.0 switches aggregated into a separate domain.

**Module contents** Contains Manager classes which contains the higher-level logic for complete domain management. Uses Adapter classes for ensuring protocol-specific connections with entities in the particular domain.

**class** `escape.adapt.managers.InternalDomainManager(**kwargs)`

Bases: `escape.util.domain.AbstractDomainManager`

Manager class to handle communication with internally emulated network.

---

**Note:** Uses `InternalMininetAdapter` for managing the emulated network and `InternalPOXAdapter` for controlling the network.

---

**name = 'INTERNAL'**

**\_\_init\_\_(\*\*kwargs)**

Init

**init(configurator, \*\*kwargs)**

Initialize Internal domain manager.

#### Parameters

- **configurator** (`ComponentConfigurator`) – component configurator for configuring adapters
- **kwargs** (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – optional parameters

**Returns** None

**finit()**

Stop polling and release dependent components.

**Returns** None

**controller\_name**

**\_setup\_sap\_hostnames()**

Setup hostnames in /etc/hosts for SAPs.

**Returns** None

**\_collect\_SAP\_infos()**

Collect necessary information from SAPs for traffic steering.

**Returns** None

**install\_nffg(nffg\_part)**

Install an `NFFG` related to the internal domain.

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be deployed

**Returns** None

**clear\_domain()**

Infrastructure Layer has already been stopped and probably cleared.

Skip cleanup process here.

**Returns** None

**\_delete\_nfs()**

Stop and delete deployed NFs.

**Returns** None

**\_deploy\_nfs(nffg\_part)**

Install the NFs mapped in the given NFFG.

If an NF is already defined in the topology and it's state is up and running then the actual NF's initiation will be skipped!

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be deployed

**Returns** None

`_delete_flowrules(nffg_part)`

Delete all flowrules from the first (default) table of all infras.

`_deploy_flowrules(nffg_part)`

Install the flowrules given in the NFFG.

If a flowrule is already defined it will be updated.

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be deployed

**Returns** None

`class escape.adapt.managers.SDNDomainManager(**kwargs)`

Bases: `escape.util.domain.AbstractDomainManager`

Manager class to handle communication with POX-controlled SDN domain.

**Note:** Uses `InternalPOXAdapter` for controlling the network.

`name = 'SDN'`

`__init__(**kwargs)`

Init

`init(configurator, **kwargs)`

Initialize SDN domain manager.

**Returns** None

`fini()`

Stop polling and release dependent components.

**Returns** None

`controller_name`

`install_nffg(nffg_part)`

Install an `NFFG` related to the SDN domain.

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be deployed

**Returns** None

`_delete_flowrules(nffg_part)`

Delete all flowrules from the first (default) table of all infras.

**Returns** None

`_deploy_flowrules(nffg_part)`

Install the flowrules given in the NFFG.

If a flowrule is already defined it will be updated.

**Parameters** `nffg_part` (`NFFG`) – NF-FG need to be deployed

**Returns** None

`clear_domain()`

Delete all flowrule in the registered SDN/OF switches.

**Returns** None

---

```
class escape.adapt.managers.RemoteESCAPEDomainManager (**kwargs)
Bases: escape.util.domain.AbstractDomainManager
```

Manager class to handle communication with other ESCAPEv2 processes started in agent-mode through a REST-API which is provided by the Resource Orchestration Sublayer.

---

**Note:** Uses RemoteESCAPEv2RESTAdapter for communicate with the remote domain.

**name** = ‘REMOTE-ESCAPE’

**\_\_init\_\_** (\*\*kwargs)

Init

**init** (configurator, \*\*kwargs)

Initialize Internal domain manager.

**Returns** None

**finit** ()

Stop polling and release dependent components.

**Returns** None

**install\_nffg** (nffg\_part)

Install an [NFFG](#) related to the internal domain.

**Parameters** **nffg\_part** ([NFFG](#)) – NF-FG need to be deployed

**Returns** None

**clear\_domain** ()

Reset remote domain based on the original (first response) topology.

**Returns** None

---

```
class escape.adapt.managers.OpenStackDomainManager (**kwargs)
```

Bases: escape.util.domain.AbstractDomainManager

Manager class to handle communication with OpenStack domain.

---

**Note:** Uses OpenStackRESTAdapter for communicate with the remote domain.

**name** = ‘OPENSTACK’

**\_\_init\_\_** (\*\*kwargs)

Init.

**init** (configurator, \*\*kwargs)

Initialize OpenStack domain manager.

**Returns** None

**finit** ()

Stop polling and release dependent components.

**Returns** None

**install\_nffg** (nffg\_part)

**clear\_domain** ()

Reset remote domain based on the original (first response) topology.

**Returns** None

---

```
class escape.adapt.managers.UniversalNodeDomainManager (**kwargs)
```

Bases: escape.util.domain.AbstractDomainManager

Manager class to handle communication with Universal Node (UN) domain.

**Note:** Uses UniversalNodeRESTAdapter for communicate with the remote domain.

**name** = ‘UN’

**\_\_init\_\_(\*\*kwargs)**  
Init.

**init(configurator, \*\*kwargs)**  
Initialize OpenStack domain manager.

**Returns** None

**finit()**

Stop polling and release dependent components.

**Returns** None

**install\_nffg(nffg\_part)**

**clear\_domain()**

Reset remote domain based on the original (first response) topology.

**Returns** None

**class escape.adapt.managers.DockerDomainManager(\*\*kwargs)**

Bases: *escape.util.domain.AbstractDomainManager*

Adapter class to handle communication component in a Docker domain.

**Warning:** Not implemented yet!

**name** = ‘DOCKER’

**\_\_init\_\_(\*\*kwargs)**  
Init

**install\_nffg(nffg\_part)**

**clear\_domain()**

### The *infrastructure.py* main module

Basic POX module for ESCAPE Infrastructure Layer

Initiate appropriate API class which emulate Co-Rm reference point

Follows POX module conventions

**infrastructure.\_start\_layer(event)**

Initiate and run Infrastructure module.

**Parameters** **event** (*GoingUpEvent*) – POX’s going up event

**Returns** None

**infrastructure.launch(standalone=False, topo=None)**

Launch function called by POX core when core is up.

**Parameters**

- **standalone** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – Run layer without dependency checking (optional)
- **topo** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – Load the topology description from file (optional)

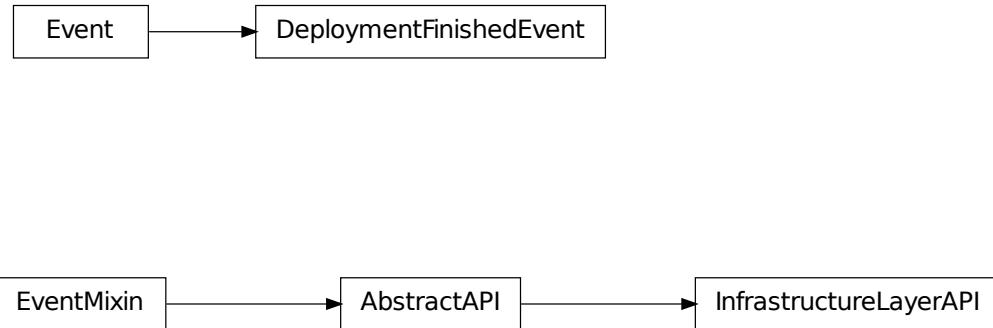
**Returns** None

## Infrastructure related classes

`escape.infr` package Sublayer for classes related to UNIFY's Infrastructure Layer (IL)

### Submodules

`il_API.py` module Emulate UNIFY's Infrastructure Layer for testing purposes based on Mininet.



`DeploymentFinishedEvent` can send status info about NFFG deployment.

`InfrastructureLayerAPI` represents the IL layer and implement all related functionality.

**Module contents** Emulate UNIFY's Infrastructure Layer for testing purposes based on Mininet.

```

class escape.infr.il_API.DeploymentFinishedEvent (success, error=None)
  Bases: pox.lib.revent.revent.Event

  Event for signaling NF-FG deployment

  __init__ (success, error=None)

class escape.infr.il_API.InfrastructureLayerAPI (standalone=False, **kwargs)
  Bases: escape.util.api.AbstractAPI

  Entry point for Infrastructure Layer (IL).

  Maintain the contact with other UNIFY layers.

  Implement a specific part of the Co - Rm reference point.

  _core_name = 'infrastructure'

  _eventMixin_events = set([<class 'escape.infr.il_API.DeploymentFinishedEvent'>])

  __init__ (standalone=False, **kwargs)
  
```

See also:

`AbstractAPI.__init__()`

`initialize()`

See also:

`AbstractAPI.initialize()`

**shutdown** (*event*)

See also:

`AbstractAPI.shutdown ()`

**\_handle\_ComponentRegistered** (*event*)

Wait for controller (internal POX module)

**Parameters** `event` (`ComponentRegistered`) – registered component event

**Returns** None

**\_handle\_DeployNFFGEvent** (\**args*, \*\**kwargs*)

Install mapped NFFG part into the emulated network.

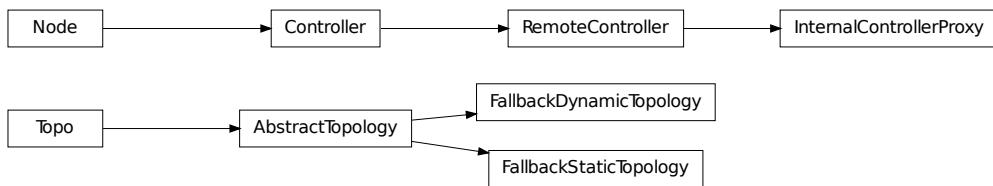
:param event:event object :return: `DeployNFFGEvent`

**topology.py module** Wrapper module for handling emulated test topology based on Mininet.

`TopologyBuilderException`

`ESCAPENetworkBuilder`

`ESCAPENetworkBridge`



`AbstractTopology` can represent an emulated topology for the high-level API.

`FallbackStaticTopology` represents the static fallback topology.

`FallbackDynamicTopology` represents the dynamic fallback topology.

`InternalControllerProxy` represents the connection between the internal controller and the emulated network.

`ESCAPENetworkBridge` represents the emulated topology in high level.

`TopologyBuilderException` can signal various error related to the topology emulation.

`ESCAPENetworkBuilder` can construct an `ESCAPENetworkBridge` object.

**Module contents** Wrapper module for handling emulated test topology based on Mininet.

**class** `escape.infr.topology.AbstractTopology` (*hopts=None*, *sopts=None*, *lopts=None*, *eopts=None*)  
 Bases: `mininet.topo.Topo`

Abstract class for representing emulated topology.

Have the functions to build a ESCAPE-specific topology.

Can be used to define reusable topology similar to Mininet's high-level API. Reusable, convenient and pre-defined way to define a topology, but less flexible and powerful.

```
default_host_opts = None
default_switch_opts = None
default_link_opts = None
default_EE_opts = None
TYPE = None
```

**\_\_init\_\_(hopts=None, sopts=None, lopts=None, eopts=None)**

**construct(builder=None)**

Base class for construct the topology.

**static get\_topo\_desc()**

Return the NFFG object represents the specific, constructed topology

```
class escape.infr.topology.FallbackStaticTopology(hopts=None, sopts=None,
lopts=None, eopts=None)
```

Bases: *escape.infr.topology.AbstractTopology*

Topology class for testing purposes and serve as a fallback topology.

Use the static way for topology compilation.

**TYPE = 'STATIC'**

**construct(builder=None)**

**static get\_topo\_desc()**

```
class escape.infr.topology.FallbackDynamicTopology(hopts=None, sopts=None,
lopts=None, eopts=None)
```

Bases: *escape.infr.topology.AbstractTopology*

Topology class for testing purposes and serve as a fallback topology.

Use the dynamic way for topology compilation.

**TYPE = 'DYNAMIC'**

**construct(builder=None)**

Set a topology with NETCONF capability for mostly testing.

**Returns** None

**static get\_topo\_desc()**

```
class escape.infr.topology.InternalControllerProxy(name='InternalPOXController',
ip='127.0.0.1', port=6653,
**kwargs)
```

Bases: *mininet.node.RemoteController*

Controller class for emulated Mininet network. Making connection with internal controller initiated by InternalPOXAdapter.

```
__init__(name='InternalPOXController', ip='127.0.0.1', port=6653, **kwargs)
```

Init.

#### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the controller (default: InternalPOXController)

- **ip** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – IP address (default: 127.0.0.1)
- **port** (*int* (<https://docs.python.org/2.7/library/functions.html#int>)) – port number (default 6633)

**checkListening()**

Check the controller port is open.

**class** `escape.infr.topology.ESCAPENetworkBridge` (*network=None, topo\_desc=None*)  
Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Internal class for representing the emulated topology.

Represents a container class for network elements such as switches, nodes, execution environments, links etc. Contains network management functions similar to Mininet's mid-level API extended with ESCAPEv2 related capabilities

Separate the interface using internally from original Mininet object to implement loose coupling and avoid changes caused by Mininet API changes e.g. 2.1.0 -> 2.2.0.

Follows Bridge design pattern.

**\_\_init\_\_** (*network=None, topo\_desc=None*)

Initialize Mininet implementation with proper attributes. Use network as the hided Mininet topology if it's given.

**Parameters**

- **topo\_desc** (*NFFG*) – static topology description e.g. the related NFFG
- **network** (`mininet.net.MininetWithControlNet`) – use this specific Mininet object for init (default: None)

**Returns** None**network**

Internal network representation.

**Returns** network representation**Return type** `mininet.net.MininetWithControlNet`**runXTerms()**

Start an xterm to every SAP if it's enabled in the global config. SAP are stored as hosts in the Mininet class.

**Returns** None**start\_network()**

Start network.

**stop\_network()**

Stop network.

**cleanup()**

Clean up junk which might be left over from old runs.

**..seealso::** `mininet.clean.cleanup()`

**get\_agent\_to\_switch** (*switch\_name*)

Return the agent to which the given switch is tided..

**Parameters** **switch\_name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the switch

**Returns** the agent**Return type** `mininet.node.NetconfAgent`

**exception** `escape.infr.topology.TopologyBuilderException`

Bases: `exceptions.Exception` (<https://docs.python.org/2.7/library/exceptions.html#exceptions.Exception>)

Exception class for topology errors.

**class** `escape.infr.topology.ESCAPENetworkBuilder` (`net=None, opts=None, fallback=True, run_dry=True`)

Bases: `object` (<https://docs.python.org/2.7/library/functions.html#object>)

Builder class for topology.

Update the network object based on the parameters if it's given or create an empty instance.

Always return with an `ESCAPENetworkBridge` instance which offer a generic interface for created `mininet.net.Mininet` object and hide implementation's nature.

Follows Builder design pattern.

`default_opts = {'listenPort': None, 'autoSetMacs': False, 'inNamespace': False, 'autoStaticArp': True, 'controller': None}`

`DEFAULT_NFFG_FORMAT = 'NFFG'`

`TYPE_EE_LOCAL = 'LOCAL'`

`TYPE_EE_REMOTE = 'REMOTE'`

`dpidBase = 1`

`dpidLen = 16`

`__init__(net=None, opts=None, fallback=True, run_dry=True)`

Initialize NetworkBuilder.

If the topology definition is not found, an exception will be raised or an empty `mininet.net.Mininet` topology will be created if `run_dry` is set.

#### Parameters

- `net` (`mininet.net.Mininet`) – update given Mininet object instead of creating a new one
- `opts` (`dict` (<https://docs.python.org/2.7/library/stdtypes.html#dict>)) – update default options with the given opts
- `fallback` (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – search for fallback topology (default: True)
- `run_dry` (`bool` (<https://docs.python.org/2.7/library/functions.html#bool>)) – do not raise an Exception and return with bare Mininet obj.

#### Returns

`None`

`get_network()`

Return the bridge to the constructed network.

**Returns** object representing the emulated network

**Return type** `ESCAPENetworkBridge`

`create_static_EE(name, cls=None, **params)`

Create and add a new EE to Mininet in the static way.

This function is for only backward compatibility.

**Warning:** Not tested yet!

#### Parameters

- `name` (`str` (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the Execution Environment
- `cls` (`mininet.node.EE`) – custom EE class/constructor (optional)

- **cores** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – Specify (real) cores that our cgroup can run on (optional)
- **frac** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – Set overall CPU fraction for this EE (optional)
- **vlanif** (*list* (<https://docs.python.org/2.7/library/functions.html#list>)) – set vlan interfaces (optional)

**Returns** newly created EE object

**Return type** mininet.node.EE

**create\_NETCONF\_EE** (*name*, *type='LOCAL'*, *\*\*params*)

Create and add a new EE to Mininet network.

The type of EE can be {local|remote} NETCONF-based.

#### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of the EE: switch: name, agent: agt\_+’name’
- **type** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – type of EE {local|remote}
- **opts** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – additional options for the switch in EE
- **dpid** – remote switch DPID (remote only)
- **username** – NETCONF username (remote only)
- **passwd** – NETCONF password (remote only)
- **ip** – control Interface for the agent (optional)
- **agentPort** – port to listen on for NETCONF connections, (else set automatically)
- **minPort** – first VNF control port which can be used (else set automatically)
- **cPort** – number of VNF control ports (and VNFs) which can be used ( default: 10)

**Returns** tuple of newly created mininet.node.Agent and mininet.node.Switch object

**Return type** tuple (<https://docs.python.org/2.7/library/functions.html#tuple>)

**create\_Switch** (*name*, *cls=None*, *\*\*params*)

Create and add a new OF switch instance to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Switch class type.

#### Parameters

- **name** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – name of switch
- **cls** (mininet.node.Switch) – custom switch class/constructor (optional)
- **dpid** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – DPID for switch (default: derived from name)
- **opts** (*str* (<https://docs.python.org/2.7/library/functions.html#str>)) – additional switch options
- **listenPort** (*int* (<https://docs.python.org/2.7/library/functions.html#int>)) – custom listening port (optional)
- **inNamespace** (*bool* (<https://docs.python.org/2.7/library/functions.html#bool>)) – override the switch spawn in namespace (optional)

- **of\_ver** ([int](https://docs.python.org/2.7/library/functions.html#int)) – override OpenFlow version (optional)
- **ip** – set IP address for the switch (optional)

**Returns** newly created Switch object

**Return type** mininet.node.Switch

**create\_Controller** (*name, controller=None, \*\*params*)

Create and add a new OF controller to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Controller class type.

**Warning:** Should not call this function and use the default InternalControllerProxy!

#### Parameters

- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) – name of controller
- **controller** ([mininet.node.Controller](#)) – custom controller class/constructor (optional)
- **inNamespace** ([bool](https://docs.python.org/2.7/library/functions.html#bool)) – override the controller spawn in namespace (optional)

**Returns** newly created Controller object

**Return type** mininet.node.Controller

**create\_SAP** (*name, cls=None, \*\*params*)

Create and add a new SAP to Mininet network.

Additional parameters are keyword arguments depend on and forwarded to the initiated Host class type.

#### Parameters

- **name** ([str](https://docs.python.org/2.7/library/functions.html#str)) – name of SAP
- **cls** ([mininet.node.Host](#)) – custom hosts class/constructor (optional)

**Returns** newly created Host object as the SAP

**Return type** mininet.node.Host

**bind\_inter\_domain\_SAPs** (*nffg*)

Search for inter-domain SAPs in given [NFFG](#), create them as a switch port and bind them to a physical interface given in sap.domain attribute.

**Parameters** **nffg** ([NFFG](#)) – topology description

**Returns** None

**\_ESCAPENetworkBuilder\_\_get\_new\_dpid()**

Generate a new DPID and return the valid format for Mininet/OVS.

**Returns** new DPID

**Return type** str ([str](https://docs.python.org/2.7/library/functions.html#str))

**\_ESCAPENetworkBuilder\_\_init\_from\_AbstractTopology** (*topo\_class*)

Build topology from pre-defined Topology class.

**Parameters** **topo\_class** ([AbstractTopology](#)) – topology

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_CONFIG (format='NFFG')**

Build a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name TOPO.

**Parameters** **format** ([str](https://docs.python.org/2.7/library/functions.html#str)) – NF-FG storing format (default: internal NFFG representation)

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_NFFG (nffg)**

Initialize topology from an [NFFG](#) representation.

**Parameters** **nffg** ([NFFG](#)) – topology object structure

**Returns** None

**\_ESCAPENetworkBuilder\_\_init\_from\_file (path, format='NFFG')**

Build a pre-defined topology from an NFFG stored in a file. The file path is searched in CONFIG with the name TOPO.

**Parameters**

- **path** ([str](https://docs.python.org/2.7/library/functions.html#str)) – file path
- **format** ([str](https://docs.python.org/2.7/library/functions.html#str)) – NF-FG storing format (default: internal NFFG representation)

**Returns** None

**create\_Link (src, dst, src\_port=None, dst\_port=None, \*\*params)**

Create an undirected connection between src and dst.

Source and destination ports can be given optionally:

**Parameters**

- **src** – source Node
- **dst** – destination Node
- **src\_port** – source Port (optional)
- **dst\_port** – destination Port (optional)
- **params** – additional link parameters

**Returns****build (topo=None)**

Initialize network.

- 1.If the additional topology is given then using that for init.
- 2.If TOPO is not given, search topology description in CONFIG with the name ‘TOPO’.
- 3.If TOPO not found or an Exception was raised, search for the fallback topo with the name FALBACK-TOPO.
- 4.If FALBACK-TOPO not found raise an exception or run a bare Mininet object if the run\_dry attribute is set

**Parameters** **topo** ([NFFG](#) or [AbstractTopology](#) or None) – optional topology representation

**Returns** object representing the emulated network

**Return type** [ESCAPENetworkBridge](#)

## **Contacts**

---

János Czentye - [janos.czentye@tmit.bme.hu](mailto:janos.czentye@tmit.bme.hu) ([janos.czentye@tmit.bme.hu](mailto:janos.czentye@tmit.bme.hu))

Balázs Sonkoly - [balazs.sonkoly@tmit.bme.hu](mailto:balazs.sonkoly@tmit.bme.hu) ([balazs.sonkoly@tmit.bme.hu](mailto:balazs.sonkoly@tmit.bme.hu))

## **Indices and tables**

---

- genindex
- modindex
- search

**a**

adaptation, 178

**e**

escape, 52  
escape.adapt, 179  
escape.adapt.adaptation, 180  
escape.adapt.adapters, 184  
escape.adapt.cas\_API, 191  
escape.adapt.managers, 193  
escape.infr, 197  
escape.infr.il\_API, 197  
escape.infr.topology, 198  
escape.orchest, 164  
escape.orchest.nfib\_mgmt, 164  
escape.orchest.policy\_enforcement, 167  
escape.orchest.ros\_API, 170  
escape.orchest.ros\_mapping, 174  
escape.orchest.ros\_orchestration, 169  
escape.orchest.virtualization\_mgmt, 176  
escape.service, 156  
escape.service.element\_mgmt, 157  
escape.service.sas\_API, 159  
escape.service.sas\_mapping, 158  
escape.service.sas\_orchestration, 162  
escape.util, 99  
escape.util.api, 99  
escape.util.config, 104  
escape.util.conversion, 109  
escape.util.domain, 115  
escape.util.mapping, 123  
escape.util.misc, 128  
escape.util.netconf, 130  
escape.util.nffg, 134  
escape.util.nffg\_elements, 142  
escape.util.pox\_extension, 154

**i**

infrastructure, 196

**o**

orchestration, 163

**S**

service, 156

**U**

unify, 155

## Symbols

\_AbstractNETCONFAdapter\_\_parse\_rpc\_params() (escape.util.netconf.AbstractNETCONFAdapter method), 133  
\_AbstractNETCONFAdapter\_\_parse\_xml\_response() (escape.util.netconf.AbstractNETCONFAdapter method), 133  
\_AbstractNETCONFAdapter\_\_remove\_namespace() (escape.util.netconf.AbstractNETCONFAdapter method), 133  
\_AbstractNETCONFAdapter\_\_suppress\_ncclient\_logging() (escape.util.netconf.AbstractNETCONFAdapter method), 133  
\_AbstractRESTAdapter\_\_suppress\_requests\_logging() (escape.util.domain.AbstractRESTAdapter method), 122  
\_ControllerAdaptationAPI\_\_proceed\_installation() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 192  
\_ESCAPEConfig\_\_parse\_part() (escape.util.config.ESCAPEConfig method), 109  
\_ESCAPENetworkBuilder\_\_get\_new\_dpid() (escape.infr.topology.ESCAPENetworkBuilder method), 97, 203  
\_ESCAPENetworkBuilder\_\_init\_from\_AbstractTopology() (escape.infr.topology.ESCAPENetworkBuilder method), 97, 203  
\_ESCAPENetworkBuilder\_\_init\_from\_CONFIG() (escape.infr.topology.ESCAPENetworkBuilder method), 98, 203  
\_ESCAPENetworkBuilder\_\_init\_from\_NFFG() (escape.infr.topology.ESCAPENetworkBuilder method), 98, 204  
\_ESCAPENetworkBuilder\_\_init\_from\_file() (escape.infr.topology.ESCAPENetworkBuilder method), 98, 204  
\_NFFGConverter\_\_convert\_flowrule\_action() (escape.util.conversion.NFFGConverter method), 114  
\_NFFGConverter\_\_convert\_flowrule\_match() (escape.util.conversion.NFFGConverter method), 114  
\_NFIBManager\_\_initialize() (escape.orchest.nfib\_mgmt.NFIBManager method), 62, 166  
\_NFIBManager\_\_suppress\_neo4j\_logging() (escape.orchest.nfib\_mgmt.NFIBManager method), 62, 166  
\_ResourceOrchestrationAPI\_\_proceed\_instantiation() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 174  
\_ResourceOrchestrationAPI\_\_update\_nffg() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 174  
\_SDNDomainTopoAdapter\_\_init\_from\_CONFIG() (escape.adapt.adapters.SDNDomainTopoAdapter method), 81, 186  
\_ServiceLayerAPI\_\_proceed\_sg\_request() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
\_Virtualizer3BasedNFFGBuilder\_\_UUID\_NUM (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 113  
\_Virtualizer3BasedNFFGBuilder\_\_add\_connection() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
\_\_call\_\_() (escape.util.misc.Singleton method), 130  
\_\_contains\_\_() (escape.util.nffg.NFFG method), 136  
\_\_contains\_\_() (escape.util.nffg\_elements.Element method), 143  
\_\_contains\_\_() (escape.util.nffg\_elements.PortContainer method), 144  
\_\_delitem\_\_() (escape.util.config.ESCAPEConfig method), 105  
\_\_enter\_\_() (escape.util.netconf.AbstractNETCONFAdapter method), 132

\_\_exit\_\_(escape.util.netconf.AbstractNETCONFAdapter method), 132  
\_\_getitem\_\_(escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
\_\_getitem\_\_(escape.util.config.ESCAPEConfig method), 105  
\_\_getitem\_\_(escape.util.nffg.NFFG method), 137  
\_\_getitem\_\_(escape.util.nffg\_elements.Element method), 143  
\_\_getitem\_\_(escape.util.nffg\_elements.NodeResource method), 145  
\_\_getitem\_\_(escape.util.nffg\_elements.PortContainer method), 144  
\_\_init\_\_(escape.adapt.adaptation.ComponentConfigurator method), 75, 180  
\_\_init\_\_(escape.adapt.adaptation.ControllerAdapter method), 76, 181  
\_\_init\_\_(escape.adapt.adaptation.DomainResourceManager method), 78, 183  
\_\_init\_\_(escape.adapt.adaptation.DomainVirtualizer method), 77, 182  
\_\_init\_\_(escape.adapt.adapters.InternalMininetAdapter method), 80, 185  
\_\_init\_\_(escape.adapt.adapters.InternalPOXAdapter method), 79, 184  
\_\_init\_\_(escape.adapt.adapters.OpenStackRESTAdapter method), 84, 189  
\_\_init\_\_(escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
\_\_init\_\_(escape.adapt.adapters.SDNDomainPOXAdapter method), 80, 185  
\_\_init\_\_(escape.adapt.adapters.SDNDomainTopoAdapter method), 81, 186  
\_\_init\_\_(escape.adapt.adapters.UniversalNodeRESTAdapter method), 84, 190  
\_\_init\_\_(escape.adapt.adapters.VNFStarterAdapter method), 81, 187  
\_\_init\_\_(escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 191  
\_\_init\_\_(escape.adapt.cas\_API.DeployNFFGEvent method), 86, 191  
\_\_init\_\_(escape.adapt.cas\_API.GlobalResInfoEvent method), 85, 191  
\_\_init\_\_(escape.adapt.cas\_API.InstallationFinishedEvent method), 85, 191  
\_\_init\_\_(escape.adapt.managers.DockerDomainManager method), 91, 196  
\_\_init\_\_(escape.adapt.managers.InternalDomainManager method), 87, 193  
\_\_init\_\_(escape.adapt.managers.OpenStackDomainManager method), 90, 195  
\_\_init\_\_(escape.adapt.managers.RemoteESCAPEDomainManager method), 89, 195  
\_\_init\_\_(escape.adapt.managers.SDNDomainManager method), 88, 194  
\_\_init\_\_(escape.adapt.managers.UniversalNodeDomainManager method), 90, 196  
\_\_init\_\_(escape.infr.il\_API.DeploymentFinishedEvent method), 91, 197  
\_\_init\_\_(escape.infr.il\_API.InfrastructureLayerAPI method), 91, 197  
\_\_init\_\_(escape.infr.topology.AbstractTopology method), 93, 199  
\_\_init\_\_(escape.infr.topology.ESCAPENetworkBridge method), 94, 200  
\_\_init\_\_(escape.infr.topology.ESCAPENetworkBuilder method), 95, 201  
\_\_init\_\_(escape.infr.topology.InternalControllerProxy method), 94, 199  
\_\_init\_\_(escape.orchest.nfib\_mgmt.NFIBManager method), 60, 164  
\_\_init\_\_(escape.orchest.policy\_enforcement.PolicyEnforcement method), 64, 168  
\_\_init\_\_(escape.orchest.ros\_API.CfOrRequestHandler method), 67, 171  
\_\_init\_\_(escape.orchest.ros\_API.InstallNFFGEvent method), 66, 170  
\_\_init\_\_(escape.orchest.ros\_API.InstantiationFinishedEvent method), 66, 171  
\_\_init\_\_(escape.orchest.ros\_API.ROSAgentRequestHandler method), 67, 172  
\_\_init\_\_(escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 172  
\_\_init\_\_(escape.orchest.ros\_API.VirtResInfoEvent method), 66, 171  
\_\_init\_\_(escape.orchest.ros\_mapping.ESCAPEMappingStrategy method), 70, 174  
\_\_init\_\_(escape.orchest.ros\_mapping.NFFGMappingFinishedEvent method), 70, 175  
\_\_init\_\_(escape.orchest.ros\_mapping.ResourceOrchestrationMapper method), 70, 175  
\_\_init\_\_(escape.orchest.ros\_orchestration.NFFGManager method), 65, 170  
\_\_init\_\_(escape.orchest.ros\_orchestration.ResourceOrchestrator method), 65, 169  
\_\_init\_\_(escape.orchest.virtualization\_mgmt.AbstractVirtualizer method), 72, 176  
\_\_init\_\_(escape.orchest.virtualization\_mgmt.GlobalViewVirtualizer method), 72, 177  
\_\_init\_\_(escape.orchest.virtualization\_mgmt.SingleBiSBiSVirtualizer method), 73, 177  
\_\_init\_\_(escape.orchest.virtualization\_mgmt.VirtualizerManager method), 73, 178  
\_\_init\_\_(escape.service.element\_mgmt.AbstractElementManager method), 53, 157  
\_\_init\_\_(escape.service.element\_mgmt.ClickManager method), 53, 157  
\_\_init\_\_(escape.service.sas\_API.GetVirtResInfoEvent method), 55, 159  
\_\_init\_\_(escape.service.sas\_API.InstantiateNFFGEvent method), 55, 159  
\_\_init\_\_(escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
\_\_init\_\_(escape.service.sas\_mapping.DefaultServiceMappingStrategy method), 53, 158

`__init__( )` (`escape.service.sas_mapping.SGMappingFinishedEvent` method), 54, 158  
`__init__( )` (`escape.service.sas_mapping.ServiceGraphMapper` method), 54, 158  
`__init__( )` (`escape.service.sas_orchestration.SGManager` method), 59, 163  
`__init__( )` (`escape.service.sas_orchestration.ServiceOrchestrator` method), 58, 162  
`__init__( )` (`escape.service.sas_orchestration.VirtualResourceManager` method), 59, 163  
`__init__( )` (`escape.util.api.AbstractAPI` method), 99  
`__init__( )` (`escape.util.api.RESTError` method), 101  
`__init__( )` (`escape.util.api.RESTServer` method), 101  
`__init__( )` (`escape.util.api.RequestCache` method), 101  
`__init__( )` (`escape.util.config.ESCAPEConfig` method), 104  
`__init__( )` (`escape.util.conversion.NFFGConverter` method), 114  
`__init__( )` (`escape.util.conversion.Virtualizer3BasedNFFGBuilder` method), 110  
`__init__( )` (`escape.util.domain.AbstractDomainManager` method), 116  
`__init__( )` (`escape.util.domain.AbstractESCAPEAdapter` method), 118  
`__init__( )` (`escape.util.domain.AbstractOFControllerAdapter` method), 118  
`__init__( )` (`escape.util.domain.AbstractRESTAdapter` method), 122  
`__init__( )` (`escape.util.domain.DeployEvent` method), 116  
`__init__( )` (`escape.util.domain.DomainChangedEvent` method), 116  
`__init__( )` (`escape.util.domain.VNFStarterAPI` method), 120  
`__init__( )` (`escape.util.mapping.AbstractMapper` method), 125  
`__init__( )` (`escape.util.mapping.AbstractMappingDataProcessor` method), 124  
`__init__( )` (`escape.util.mapping.AbstractMappingStrategy` method), 123  
`__init__( )` (`escape.util.mapping.AbstractOrchestrator` method), 127  
`__init__( )` (`escape.util.mapping.PostMapEvent` method), 125  
`__init__( )` (`escape.util.mapping.PreMapEvent` method), 125  
`__init__( )` (`escape.util.misc.SimpleStandaloneHelper` method), 129  
`__init__( )` (`escape.util.netconf.AbstractNETCONFAdapter` method), 130  
`__init__( )` (`escape.util.nffg.NFFG` method), 136  
`__init__( )` (`escape.util.nffg_elements.EdgeLink` method), 149  
`__init__( )` (`escape.util.nffg_elements.EdgeReq` method), 150  
`__init__( )` (`escape.util.nffg_elements.EdgeSGLink` method), 150  
`__init__( )` (`escape.util.nffg_elements.Element` method), 143  
`__init__( )` (`escape.util.nffg_elements.Flowrule` method), 146  
`__init__( )` (`escape.util.nffg_elements.InfraPort` method), 147  
`__init__( )` (`escape.util.nffg_elements.Link` method), 145  
`__init__( )` (`escape.util.nffg_elements.NFFGModel` method), 151  
`__init__( )` (`escape.util.nffg_elements.Node` method), 144  
`__init__( )` (`escape.util.nffg_elements.NodeInfra` method), 148  
`__init__( )` (`escape.util.nffg_elements.NodeNF` method), 148  
`__init__( )` (`escape.util.nffg_elements.NodeResource` method), 145  
`__init__( )` (`escape.util.nffg_elements.NodeSAP` method), 148  
`__init__( )` (`escape.util.nffg_elements.Port` method), 146  
`__init__( )` (`escape.util.nffg_elements.PortContainer` method), 144  
`__init__( )` (`escape.util.pox_extension.ExtendedOFConnectionArbiter` method), 154  
`__iter__( )` (`escape.adapt.adaptation.ComponentConfigurator` method), 76, 181  
`__iter__( )` (`escape.util.nffg.NFFG` method), 136  
`__iter__( )` (`escape.util.nffg_elements.PortContainer` method), 144  
`__len__( )` (`escape.util.nffg.NFFG` method), 137  
`__len__( )` (`escape.util.nffg_elements.PortContainer` method), 144  
`__metaclass__` (`escape.orchest.virtualization_mgmt.AbstractVirtualizer` attribute), 72, 176  
`__metaclass__` (`escape.util.config.ESCAPEConfig` attribute), 104  
`new__( )` (`escape.orchest.policy_enforcement.PolicyEnforcementMetaClass` static method), 63, 167  
`repr__( )` (`escape.adapt.adaptation.DomainVirtualizer` method), 77, 182  
`repr__( )` (`escape.orchest.virtualization_mgmt.AbstractVirtualizer` method), 72, 176  
`repr__( )` (`escape.util.nffg.NFFG` method), 136  
`repr__( )` (`escape.util.nffg_elements.EdgeLink` method), 150  
`repr__( )` (`escape.util.nffg_elements.Flowrule` method), 146  
`repr__( )` (`escape.util.nffg_elements.Link` method), 145

\_\_repr\_\_() (escape.util.nffg\_elements.Node method), 145  
\_\_repr\_\_() (escape.util.nffg\_elements.NodeInfra method), 149  
\_\_repr\_\_() (escape.util.nffg\_elements.NodeResource method), 146  
\_\_repr\_\_() (escape.util.nffg\_elements.NodeSAP method), 148  
\_\_repr\_\_() (escape.util.nffg\_elements.Port method), 147  
\_\_repr\_\_() (escape.util.nffg\_elements.PortContainer method), 144  
\_\_repr\_\_() (escape.util.pox\_extension.ESCAPEInteractiveHelper method), 155  
\_\_setitem\_\_() (escape.util.config.ESCAPEConfig method), 105  
\_\_setitem\_\_() (escape.util.nffg\_elements.Element method), 143  
\_\_setitem\_\_() (escape.util.nffg\_elements.NodeResource method), 146  
\_\_str\_\_() (escape.adapt.adaptation.DomainVirtualizer method), 77, 182  
\_\_str\_\_() (escape.orchest.virtualization\_mgmt.AbstractVirtualizer method), 72, 176  
\_\_str\_\_() (escape.util.api.AbstractAPI method), 100  
\_\_str\_\_() (escape.util.api.RESTError method), 101  
\_\_str\_\_() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 110  
\_\_str\_\_() (escape.util.nffg.NFFG method), 136  
\_\_str\_\_() (escape.util.nffg\_elements.EdgeLink method), 150  
\_\_str\_\_() (escape.util.nffg\_elements.Flowrule method), 146  
\_\_str\_\_() (escape.util.nffg\_elements.Node method), 145  
\_\_str\_\_() (escape.util.nffg\_elements.NodeInfra method), 149  
\_\_str\_\_() (escape.util.nffg\_elements.NodeNF method), 148  
\_\_str\_\_() (escape.util.nffg\_elements.NodeResource method), 146  
\_\_str\_\_() (escape.util.nffg\_elements.NodeSAP method), 148  
\_\_str\_\_() (escape.util.nffg\_elements.PortContainer method), 144  
\_all\_dependencies\_met() (escape.util.api.AbstractAPI method), 100  
\_collect\_SAP\_infos() (escape.adapt.managers.InternalDomainManager method), 88, 193  
\_core\_name (escape.adapt.cas\_API.ControllerAdaptationAPI attribute), 86, 191  
\_core\_name (escape.infr.il\_API.InfrastructureLayerAPI attribute), 91, 197  
\_core\_name (escape.orchest.ros\_API.ResourceOrchestrationAPI attribute), 68, 172  
\_core\_name (escape.service.sas\_API.ServiceLayerAPI attribute), 56, 160  
\_core\_name (escape.util.api.AbstractAPI attribute), 99  
\_core\_name (escape.util.pox\_extension.ExtendedOFConnectionArbiter attribute), 154  
\_create\_rpc\_request() (escape.util.netconf.AbstractNETCONFAdapter method), 132  
\_delete\_flowrules() (escape.adapt.managers.InternalDomainManager method), 88, 194  
\_delete\_flowrules() (escape.adapt.managers.SDNDomainManager method), 89, 194  
\_delete\_nfs() (escape.adapt.managers.InternalDomainManager method), 88, 193  
\_deploy\_flowrules() (escape.adapt.managers.InternalDomainManager method), 88, 194  
\_deploy\_flowrules() (escape.adapt.managers.SDNDomainManager method), 89, 194  
\_deploy\_nfs() (escape.adapt.managers.InternalDomainManager method), 88, 193  
\_detect\_topology() (escape.util.domain.AbstractDomainManager method), 117  
\_eventMixin\_events (escape.adapt.adapters.InternalMininetAdapter attribute), 80, 185  
\_eventMixin\_events (escape.infr.il\_API.InfrastructureLayerAPI attribute), 91, 197  
\_eventMixin\_events (escape.orchest.ros\_mapping.ResourceOrchestrationMapper attribute), 70, 175  
\_eventMixin\_events (escape.orchest.virtualization\_mgmt.VirtualizerManager attribute), 73, 178  
\_eventMixin\_events (escape.service.sas\_mapping.ServiceGraphMapper attribute), 54, 158  
\_eventMixin\_events (escape.service.sas\_orchestration.VirtualResourceManager attribute), 59, 163  
\_eventMixin\_events (escape.util.domain.AbstractDomainManager attribute), 116  
\_eventMixin\_events (escape.util.domain.AbstractESCAPEAdapter attribute), 118  
\_find\_static\_link() (escape.util.nffg.NFFGToolBox static method), 141  
\_generate\_global\_view() (escape.orchest.virtualization\_mgmt.VirtualizerManager method), 74, 178  
\_generate\_id() (escape.orchest.ros\_orchestration.NFFGManager method), 65, 170  
\_generate\_id() (escape.service.sas\_orchestration.SGManager method), 59, 163  
\_generate\_one\_bisbis() (escape.orchest.virtualization\_mgmt.SingleBiSBiSVirtualizer method), 73, 177  
\_generate\_single\_view() (escape.orchest.virtualization\_mgmt.VirtualizerManager method), 73, 178  
\_get\_body() (escape.util.api.AbstractRequestHandler method), 102  
\_get\_output\_port\_of\_TAG\_action() (escape.util.nffg.NFFGToolBox static method), 141  
\_handle\_ComponentRegistered() (escape.infr.il\_API.InfrastructureLayerAPI method), 92, 198  
\_handle\_ConnectionDown() (escape.adapt.adapters.InternalPOXAdapter method), 79, 185

\_handle\_ConnectionUp() (escape.adapt.adapters.InternalPOXAdapter method), 79, 185  
 \_handle\_DeployEvent() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 192  
 \_handle\_DeployNFFGEvent() (escape.infr.il\_API.InfrastructureLayerAPI method), 92, 198  
 \_handle\_DeploymentFinishedEvent() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 192  
 \_handle\_DomainChangedEvent() (escape.adapt.adaptation.ControllerAdapter method), 77, 182  
 \_handle\_GetGlobalResInfoEvent() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 191  
 \_handle\_GetVirtResInfoEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_handle\_GlobalResInfoEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_handle\_InstallNFFGEvent() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 191  
 \_handle\_InstallationFinishedEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_handle\_InstantiateNFFGEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_handle\_InstantiationFinishedEvent() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
 \_handle\_MissingGlobalViewEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_handle\_MissingVirtualViewEvent() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
 \_handle\_NFFGMappingFinishedEvent() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 173  
 \_handle\_SGMMappingFinishedEvent() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
 \_handle\_VirtResInfoEvent() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
 \_handle\_keepalive\_handler() (escape.util.domain.AbstractOFControllerAdapter class method), 119  
 \_identify\_ovs\_device() (escape.adapt.adapters.InternalPOXAdapter method), 79, 185  
 \_initiate\_cfor\_api() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 172  
 \_initiate\_gui() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
 \_initiate\_rest\_api() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
 \_initiate\_ros\_api() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 172  
 \_install\_NFFG() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 69, 173  
 \_instances (escape.util.misc.Singleton attribute), 130  
 \_instantiate\_NFFG() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
 \_interval (escape.util.domain.AbstractOFControllerAdapter attribute), 118  
 \_invoke\_rpc() (escape.adapt.adapters.VNFStarterAdapter method), 82, 187  
 \_invoke\_rpc() (escape.util.netconf.AbstractNETCONFAdapter method), 132  
 \_is\_port\_finishing\_flow() (escape.util.nffg.NFFGToolBox static method), 141  
 \_log\_event() (escape.util.misc.SimpleStandaloneHelper method), 129  
 \_mapping\_finished() (escape.orchest.ros\_mapping.ResourceOrchestrationMapper method), 71, 175  
 \_mapping\_finished() (escape.service.sas\_mapping.ServiceGraphMapper method), 54, 158  
 \_mapping\_finished() (escape.util.mapping.AbstractMapper method), 126  
 \_parse\_rpc\_response() (escape.util.netconf.AbstractNETCONFAdapter method), 132  
 \_perform\_mapping() (escape.orchest.ros\_mapping.ResourceOrchestrationMapper method), 71, 175  
 \_perform\_mapping() (escape.service.sas\_mapping.ServiceGraphMapper method), 54, 158  
 \_perform\_mapping() (escape.util.mapping.AbstractMapper method), 126  
 \_proceed\_API\_call() (escape.util.api.AbstractRequestHandler method), 103  
 \_process\_url() (escape.util.api.AbstractRequestHandler method), 102  
 \_read\_json\_from\_file() (escape.util.api.AbstractAPI static method), 100  
 \_register\_listeners() (escape.util.misc.SimpleStandaloneHelper method), 129  
 \_send\_json\_response() (escape.util.api.AbstractRequestHandler method), 103  
 \_setup\_sap\_hostnames() (escape.adapt.managers.InternalDomainManager method), 88, 193  
 \_split\_into\_domains() (escape.adapt.adaptation.ControllerAdapter method), 77, 182  
 \_start\_components() (in module unify), 155  
 \_start\_layer() (in module adaptation), 179  
 \_start\_layer() (in module infrastructure), 196  
 \_start\_layer() (in module orchestration), 163  
 \_start\_layer() (in module service), 156  
 \_start\_mapping() (escape.util.mapping.AbstractMapper method), 126  
 \_switch\_timeout (escape.util.domain.AbstractOFControllerAdapter attribute), 118  
 \_update\_REMOTE\_ESCAPE\_domain() (escape.orchest.ros\_API.ROSAgentRequestHandler method), 67, 172

## A

AbstractAPI (class in escape.util.api), 99  
 AbstractDomainManager (class in escape.util.domain), 116  
 AbstractElementManager (class in escape.service.element\_mgmt), 53, 157

AbstractESCAPEAdapter (class in escape.util.domain), 117  
AbstractMapper (class in escape.util.mapping), 125  
AbstractMappingDataProcessor (class in escape.util.mapping), 124  
AbstractMappingStrategy (class in escape.util.mapping), 123  
AbstractNETCONFAdapter (class in escape.util.netconf), 130  
AbstractNFFG (class in escape.util.nffg), 134  
AbstractOFControllerAdapter (class in escape.util.domain), 118  
AbstractOrchestrator (class in escape.util.mapping), 127  
AbstractRequestHandler (class in escape.util.api), 101  
AbstractRESTAdapter (class in escape.util.domain), 122  
AbstractTopology (class in escape.infr.topology), 93, 198  
AbstractVirtualizer (class in escape.orchest.virtualization\_mgmt), 72, 176  
activate() (escape.util.pox\_extension.ExtendedOFConnectionArbiter class method), 155  
adapt\_mapping\_into\_Virtualizer() (escape.util.conversion.NFFGConverter method), 114  
adaptation (module), 178  
ADD (escape.util.nffg\_elements.Element attribute), 143  
add\_cfg() (escape.util.config.ESCAPEConfig method), 104  
add\_connection\_listener() (escape.util.pox\_extension.ExtendedOFConnectionArbiter method), 154  
add\_dependencies() (in module escape), 52  
add\_edge() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 111  
add\_edge() (escape.util.nffg.AbstractNFFG method), 135  
add\_edge() (escape.util.nffg.NFFG method), 137  
add\_flow\_entry() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 112  
add\_flowrule() (escape.util.nffg\_elements.InfraPort method), 147  
add\_infra() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_infra() (escape.util.nffg.AbstractNFFG method), 134  
add\_infra() (escape.util.nffg.NFFG method), 138  
add\_infra() (escape.util.nffg\_elements.NFFGModel method), 152  
add\_infrastructure\_node() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 111  
add\_inter\_infra\_link() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_link() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_link() (escape.util.nffg.AbstractNFFG method), 134  
add\_link() (escape.util.nffg.NFFG method), 138  
add\_link() (escape.util.nffg\_elements.NFFGModel method), 152  
add\_link\_resource() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 112  
add\_nf() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_nf() (escape.util.nffg.AbstractNFFG method), 134  
add\_nf() (escape.util.nffg.NFFG method), 137  
add\_nf() (escape.util.nffg\_elements.NFFGModel method), 151  
add\_nf\_instance() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 112  
add\_node() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 111  
add\_node() (escape.util.nffg.AbstractNFFG method), 135  
add\_node() (escape.util.nffg.NFFG method), 137  
add\_node\_port() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 111  
add\_node\_resource() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 111  
add\_port() (escape.util.nffg\_elements.Node method), 144  
add\_port() (escape.util.nffg\_elements.NodeInfra method), 149  
add\_property() (escape.util.nffg\_elements.Port method), 146  
add\_req() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_req() (escape.util.nffg.AbstractNFFG method), 135  
add\_req() (escape.util.nffg.NFFG method), 139  
add\_req() (escape.util.nffg\_elements.NFFGModel method), 153  
add\_request() (escape.util.api.RequestCache method), 101  
add\_sap() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
add\_sap() (escape.util.nffg.AbstractNFFG method), 134  
add\_sap() (escape.util.nffg.NFFG method), 138  
add\_sap() (escape.util.nffg\_elements.NFFGModel method), 152  
add\_sg\_hop() (escape.util.nffg\_elements.NFFGModel method), 153

add\_slink() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
 add\_slink() (escape.util.nffg.AbstractNFFG method), 134  
 add\_slink() (escape.util.nffg.NFFG method), 139  
 add\_supported\_nf() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 112  
 add\_supported\_type() (escape.util.nffg\_elements.NodeInfra method), 149  
 add\_undirected\_link() (escape.util.nffg.NFFG method), 139  
 addClickNF() (escape.orchest.nfib\_mgmt.NFIBManager method), 60, 164  
 addDecomp() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 165  
 addNode() (escape.orchest.nfib\_mgmt.NFIBManager method), 60, 164  
 addRelationship() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 165  
 addVMNF() (escape.orchest.nfib\_mgmt.NFIBManager method), 60, 165  
 api\_cfor\_edit\_config() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 173  
 api\_cfor\_get\_config() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 173  
 api\_ros\_edit\_config() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 173  
 api\_ros\_get\_config() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 173  
 api\_sas\_get\_topology() (escape.service.sas\_API.ServiceLayerAPI method), 56, 161  
 api\_sas\_sg\_request() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
 api\_sas\_sg\_request\_delayed() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
 append() (escape.util.nffg\_elements.PortContainer method), 144

## B

bind\_inter\_domain\_SAPs() (escape.infr.topology.ESCAPENetworkBuilder method), 97, 203  
 bounded\_layer (escape.orchest.ros\_API.CfOrRequestHandler attribute), 67, 171  
 bounded\_layer (escape.orchest.ros\_API.ROSAgentRequestHandler attribute), 67, 172  
 bounded\_layer (escape.service.sas\_API.ServiceRequestHandler attribute), 55, 159  
 bounded\_layer (escape.util.api.AbstractRequestHandler attribute), 102  
 build() (escape.infr.topology.ESCAPENetworkBuilder method), 98, 204  
 build() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 110

## C

call\_as\_coop\_task() (in module escape.util.misc), 128  
 call\_delayed\_as\_coop\_task() (in module escape.util.misc), 128  
 call\_RPC() (escape.util.netconf.AbstractNETCONFAdapter method), 132  
 CfOrRequestHandler (class in escape.orchest.ros\_API), 66, 171  
 check\_domain\_reachable() (escape.adapt.adapters.InternalMininetAdapter method), 80, 186  
 check\_domain\_reachable() (escape.adapt.adapters.InternalPOXAdapter method), 79, 185  
 check\_domain\_reachable() (escape.adapt.adapters.OpenStackRESTAdapter method), 84, 190  
 check\_domain\_reachable() (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
 check\_domain\_reachable() (escape.adapt.adapters.SDNDomainPOXAdapter method), 80, 185  
 check\_domain\_reachable() (escape.adapt.adapters.SDNDomainTopoAdapter method), 81, 186  
 check\_domain\_reachable() (escape.adapt.adapters.UniversalNodeRESTAdapter method), 85, 190  
 check\_domain\_reachable() (escape.adapt.adapters.VNFStarterAdapter method), 81, 187  
 check\_domain\_reachable() (escape.util.domain.AbstractESCAPEAdapter method), 118  
 check\_domain\_reachable() (escape.util.domain.AbstractOFControllerAdapter method), 119  
 checkListening() (escape.infr.topology.InternalControllerProxy method), 94, 200  
 cleanup() (escape.infr.topology.ESCAPENetworkBridge method), 94, 200  
 clear() (escape.util.nffg\_elements.Element method), 143  
 clear() (escape.util.nffg\_elements.PortContainer method), 144  
 clear\_domain() (escape.adapt.managers.DockerDomainManager method), 91, 196  
 clear\_domain() (escape.adapt.managers.InternalDomainManager method), 88, 193  
 clear\_domain() (escape.adapt.managers.OpenStackDomainManager method), 90, 195  
 clear\_domain() (escape.adapt.managers.RemoteESCAPEDomainManager method), 89, 195  
 clear\_domain() (escape.adapt.managers.SDNDomainManager method), 89, 194  
 clear\_domain() (escape.adapt.managers.UniversalNodeDomainManager method), 90, 196  
 clear\_domain() (escape.util.domain.AbstractDomainManager method), 117  
 clear\_flows\_on\_connect (escape.util.pox\_extension.OpenFlowBridge attribute), 154  
 clear\_initiated\_mgrs() (escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
 clear\_links() (escape.util.nffg.NFFG method), 140

clear\_nodes() (escape.util.nffg.NFFG method), 140  
clickCompile() (escape.orchest.nfib\_mgmt.NFIBManager static method), 60, 165  
ClickManager (class in escape.service.element\_mgmt), 53, 157  
code (escape.util.api.RESTError attribute), 101  
ComponentConfigurator (class in escape.adapt.adaptation), 75, 180  
components (escape.adapt.adaptation.ComponentConfigurator attribute), 75, 181  
config() (escape.util.pox\_extension.ESCAPEInteractiveHelper method), 155  
connect() (escape.util.netconf.AbstractNETCONFAdapter method), 131  
CONNECTED (escape.util.domain.VNFStarterAPI.ConnectedStatus attribute), 120  
connected (escape.util.netconf.AbstractNETCONFAdapter attribute), 131  
connection\_data (escape.util.netconf.AbstractNETCONFAdapter attribute), 131  
CONNECTION\_TIMEOUT (escape.util.domain.AbstractRESTAdapter attribute), 122  
connectVNF() (escape.adapt.adapters.VNFStarterAdapter method), 82, 187  
connectVNF() (escape.util.domain.VNFStarterAPI method), 120  
construct() (escape.infr.topology.AbstractTopology method), 93, 199  
construct() (escape.infr.topology.FallbackDynamicTopology method), 93, 199  
construct() (escape.infr.topology.FallbackStaticTopology method), 93, 199  
controller\_name (escape.adapt.managers.InternalDomainManager attribute), 88, 193  
controller\_name (escape.adapt.managers.SDNDomainManager attribute), 89, 194  
ControllerAdaptationAPI (class in escape.adapt.cas\_API), 86, 191  
ControllerAdapter (class in escape.adapt.adaptation), 76, 181  
copy() (escape.util.nffg.NFFG method), 141  
copy() (escape.util.nffg\_elements.Element method), 143  
create\_Controller() (escape.infr.topology.ESCAPENetworkBuilder method), 97, 203  
create\_Link() (escape.infr.topology.ESCAPENetworkBuilder method), 98, 204  
create\_NETCONF\_EE() (escape.infr.topology.ESCAPENetworkBuilder method), 96, 202  
create\_SAP() (escape.infr.topology.ESCAPENetworkBuilder method), 97, 203  
create\_static\_EE() (escape.infr.topology.ESCAPENetworkBuilder method), 95, 201  
create\_Switch() (escape.infr.topology.ESCAPENetworkBuilder method), 96, 202  
custom\_headers (escape.util.domain.AbstractRESTAdapter attribute), 122

## D

DEFAULT\_CFG (escape.util.config.ESCAPEConfig attribute), 104  
default\_EE\_opts (escape.infr.topology.AbstractTopology attribute), 93, 199  
default\_host\_opts (escape.infr.topology.AbstractTopology attribute), 93, 199  
DEFAULT\_INFRA\_TYPE (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
default\_link\_opts (escape.infr.topology.AbstractTopology attribute), 93, 199  
DEFAULT\_MAPPER (escape.orchest.ros\_orchestration.ResourceOrchestrator attribute), 65, 169  
DEFAULT\_MAPPER (escape.service.sas\_orchestration.ServiceOrchestrator attribute), 58, 162  
DEFAULT\_MAPPER (escape.util.mapping.AbstractOrchestrator attribute), 127  
DEFAULT\_NFFG\_FORMAT (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
DEFAULT\_NODE\_TYPE (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
default\_opts (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
DEFAULT\_STRATEGY (escape.orchest.ros\_mapping.ResourceOrchestrationMapper attribute), 70, 175  
DEFAULT\_STRATEGY (escape.service.sas\_mapping.ServiceGraphMapper attribute), 54, 158  
DEFAULT\_STRATEGY (escape.util.mapping.AbstractMapper attribute), 125  
default\_switch\_opts (escape.infr.topology.AbstractTopology attribute), 93, 199  
DefaultDomainRESTAPI (class in escape.util.domain), 121  
DefaultServiceMappingStrategy (class in escape.service.sas\_mapping), 53, 158  
DEL (escape.util.nffg\_elements.Element attribute), 143  
del\_edge() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113  
del\_edge() (escape.util.nffg.AbstractNFFG method), 135  
del\_edge() (escape.util.nffg.NFFG method), 137  
del\_flowrule() (escape.util.nffg\_elements.InfraPort method), 147  
del\_infra() (escape.util.nffg\_elements.NFFGModel method), 152  
del\_link() (escape.util.nffg\_elements.NFFGModel method), 152  
del\_nf() (escape.util.nffg\_elements.NFFGModel method), 152  
del\_node() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 113

del\_node() (escape.util.nffg.AbstractNFFG method), 135  
del\_node() (escape.util.nffg.NFFG method), 137  
del\_port() (escape.util.nffg\_elements.Node method), 144  
del\_property() (escape.util.nffg\_elements.Port method), 146  
del\_req() (escape.util.nffg\_elements.NFFGModel method), 153  
del\_sap() (escape.util.nffg\_elements.NFFGModel method), 152  
del\_sg\_hop() (escape.util.nffg\_elements.NFFGModel method), 153  
del\_supported\_type() (escape.util.nffg\_elements.NodeInfra method), 149  
delete\_flowrules() (escape.util.domain.AbstractOFControllerAdapter method), 119  
dependencies (escape.orchest.ros\_API.ResourceOrchestrationAPI attribute), 68, 172  
dependencies (escape.service.sas\_API.ServiceLayerAPI attribute), 56, 160  
dependencies (escape.util.api.AbstractAPI attribute), 99  
DeployEvent (class in escape.util.domain), 116  
DeploymentFinishedEvent (class in escape.infr.il\_API), 91, 197  
deployNF() (escape.adapt.adapters.VNFStarterAdapter method), 83, 189  
DeployNFFGEVENT (class in escape.adapt.cas\_API), 85, 191  
deprecated() (in module escape.util.misc), 130  
DESCRIPTION (escape.util.nffg\_elements.NFFGModel attribute), 151  
disconnect() (escape.util.netconf.AbstractNETCONFAdapter method), 131  
DISCONNECTED (escape.util.domain.VNFStarterAPI.ConnectedStatus attribute), 120  
disconnectVNF() (escape.adapt.adapters.VNFStarterAdapter method), 82, 188  
disconnectVNF() (escape.util.domain.VNFStarterAPI method), 120  
do\_CONNECT() (escape.util.api.AbstractRequestHandler method), 102  
do\_DELETE() (escape.util.api.AbstractRequestHandler method), 102  
do\_GET() (escape.util.api.AbstractRequestHandler method), 102  
do\_HEAD() (escape.util.api.AbstractRequestHandler method), 102  
do\_OPTIONS() (escape.util.api.AbstractRequestHandler method), 102  
do\_POST() (escape.util.api.AbstractRequestHandler method), 102  
do\_PUT() (escape.util.api.AbstractRequestHandler method), 102  
do\_TRACE() (escape.util.api.AbstractRequestHandler method), 102  
DockerDomainManager (class in escape.adapt.managers), 90, 196  
DOMAIN\_DOCKER (escape.util.nffg.NFFG attribute), 136  
DOMAIN\_DOCKER (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_INTERNAL (escape.util.nffg.NFFG attribute), 135  
DOMAIN\_INTERNAL (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_MAPPING (escape.adapt.adaptation.ControllerAdapter attribute), 76, 181  
DOMAIN\_OS (escape.util.nffg.NFFG attribute), 135  
DOMAIN\_OS (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_REMOTE (escape.util.nffg.NFFG attribute), 135  
DOMAIN\_REMOTE (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_SDN (escape.util.nffg.NFFG attribute), 136  
DOMAIN\_SDN (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_UN (escape.util.nffg.NFFG attribute), 136  
DOMAIN\_UN (escape.util.nffg\_elements.NodeInfra attribute), 148  
DOMAIN\_VIRTUAL (escape.util.nffg.NFFG attribute), 135  
DOMAIN\_VIRTUAL (escape.util.nffg\_elements.NodeInfra attribute), 148  
DomainChangedEvent (class in escape.util.domain), 115  
DomainResourceManager (class in escape.adapt.adaptation), 78, 183  
DomainVirtualizer (class in escape.adapt.adaptation), 77, 182  
dov (escape.orchest.virtualization\_mgmt.VirtualizerManager attribute), 73, 178  
dpid\_to\_infra (escape.adapt.adapters.SDNDomainPOXAdapter attribute), 80, 185  
dpidBase (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
dpidLen (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
dump() (escape.util.config.ESCAPEConfig method), 105  
dump() (escape.util.conversion.Virtualizer3BasedNFFGBuilder method), 110  
dump() (escape.util.nffg.AbstractNFFG method), 135  
dump() (escape.util.nffg.NFFG method), 140  
dump() (escape.util.nffg\_elements.NFFGModel method), 153

duplicate\_static\_links() (escape.util.nffg.NFFG method), 140  
DYNAMIC (escape.util.nffg\_elements.Link attribute), 145

## E

EdgeLink (class in escape.util.nffg\_elements), 149  
EdgeReq (class in escape.util.nffg\_elements), 150  
edges (escape.util.nffg\_elements.NFFGModel attribute), 151  
EdgeSGLink (class in escape.util.nffg\_elements), 150  
edit\_config() (escape.adapt.adapters.OpenStackRESTAdapter method), 84, 190  
edit\_config() (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
edit\_config() (escape.adapt.adapters.UniversalNodeRESTAdapter method), 85, 190  
edit\_config() (escape.orchest.ros\_API.CfOrRequestHandler method), 67, 171  
edit\_config() (escape.orchest.ros\_API.ROSAgentRequestHandler method), 67, 172  
edit\_config() (escape.util.domain.DefaultDomainRESTAPI method), 121  
edit\_config() (escape.util.pox\_extension.ESCAPEInteractiveHelper method), 155  
Element (class in escape.util.nffg\_elements), 143  
enum() (in module escape.util.misc), 129  
ERROR (escape.util.api.RequestCache attribute), 101  
error\_content\_type (escape.util.api.AbstractRequestHandler attribute), 103  
escape (module), 52  
escape.adapt (module), 74, 179  
escape.adapt.adaptation (module), 75, 180  
escape.adapt.adapters (module), 79, 184  
escape.adapt.cas\_API (module), 85, 191  
escape.adapt.managers (module), 87, 193  
escape.infr (module), 91, 197  
escape.infr.il\_API (module), 91, 197  
escape.infr.topology (module), 93, 198  
escape.orchest (module), 59, 164  
escape.orchest.nfib\_mgmt (module), 60, 164  
escape.orchest.policy\_enforcement (module), 62, 167  
escape.orchest.ros\_API (module), 66, 170  
escape.orchest.ros\_mapping (module), 70, 174  
escape.orchest.ros\_orchestration (module), 65, 169  
escape.orchest.virtualization\_mgmt (module), 72, 176  
escape.service (module), 52, 156  
escape.service.element\_mgmt (module), 53, 157  
escape.service.sas\_API (module), 55, 159  
escape.service.sas\_mapping (module), 53, 158  
escape.service.sas\_orchestration (module), 58, 162  
escape.util (module), 99  
escape.util.api (module), 99  
escape.util.config (module), 104  
escape.util.conversion (module), 109  
escape.util.domain (module), 115  
escape.util.mapping (module), 123  
escape.util.misc (module), 128  
escape.util.netconf (module), 130  
escape.util.nffg (module), 134  
escape.util.nffg\_elements (module), 142  
escape.util.pox\_extension (module), 154  
ESCAPEConfig (class in escape.util.config), 104  
ESCAPEInteractiveHelper (class in escape.util.pox\_extension), 155  
ESCAPEMappingStrategy (class in escape.orchest.ros\_mapping), 70, 174  
ESCAPENetworkBridge (class in escape.infr.topology), 94, 200  
ESCAPENetworkBuilder (class in escape.infr.topology), 95, 201  
ExtendedOFConnectionArbiter (class in escape.util.pox\_extension), 154

## F

FAILED (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 119  
 FallbackDynamicTopology (class in escape.infr.topology), 93, 199  
 FallbackStaticTopology (class in escape.infr.topology), 93, 199  
 filter\_connections() (escape.util.domain.AbstractOFControllerAdapter method), 119  
 finit() (escape.adapt.managers.InternalDomainManager method), 88, 193  
 finit() (escape.adapt.managers.OpenStackDomainManager method), 90, 195  
 finit() (escape.adapt.managers.RemoteESCAPEDomainManager method), 89, 195  
 finit() (escape.adapt.managers.SDNDomainManager method), 89, 194  
 finit() (escape.adapt.managers.UniversalNodeDomainManager method), 90, 196  
 finit() (escape.util.domain.AbstractDomainManager method), 116  
 Flowrule (class in escape.util.nffg\_elements), 146

## G

generate\_all\_TAGS\_of\_NFFG() (escape.util.nffg.NFFGToolBox static method), 141  
 generate\_id() (escape.util.nffg.NFFG method), 141  
 GET (escape.util.domain.AbstractRESTAdapter attribute), 122  
 get() (escape.orchest.ros\_orchestration.NFFGManager method), 65, 170  
 get() (escape.service.sas\_orchestration.SGManager method), 59, 163  
 get() (escape.util.netconf.AbstractNETCONFAdapter method), 131  
 get() (escape.util.nffg\_elements.Element method), 143  
 get\_adapter\_keepalive() (escape.util.config.ESCAPEConfig method), 108  
 get\_agent\_connection\_params() (escape.adapt.adapters.InternalMininetAdapter method), 80, 186  
 get\_agent\_to\_switch() (escape.infr.topology.ESCAPENetworkBridge method), 94, 200  
 get\_api\_virtualizer() (escape.util.config.ESCAPEConfig method), 108  
 get\_cfor\_api\_address() (escape.util.config.ESCAPEConfig method), 108  
 get\_cfor\_api\_class() (escape.util.config.ESCAPEConfig method), 108  
 get\_cfor\_api\_prefix() (escape.util.config.ESCAPEConfig method), 108  
 get\_clean\_after\_shutdown() (escape.util.config.ESCAPEConfig method), 107  
 get\_component() (escape.util.config.ESCAPEConfig method), 106  
 get\_component\_params() (escape.util.config.ESCAPEConfig method), 106  
 get\_config() (escape.adapt.adapters.OpenStackRESTAdapter method), 84, 190  
 get\_config() (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
 get\_config() (escape.adapt.adapters.UniversalNodeRESTAdapter method), 85, 190  
 get\_config() (escape.orchest.ros\_API.CfOrRequestHandler method), 67, 171  
 get\_config() (escape.orchest.ros\_API.ROSAgentRequestHandler method), 67, 172  
 get\_config() (escape.util.domain.DefaultDomainRESTAPI method), 121  
 get\_config() (escape.util.netconf.AbstractNETCONFAdapter method), 131  
 get\_config() (escape.util.pox\_extension.ESCAPEInteractiveHelper method), 155  
 get\_Controller\_params() (escape.util.config.ESCAPEConfig method), 108  
 get\_EE\_params() (escape.util.config.ESCAPEConfig method), 108  
 get\_fallback\_topology() (escape.util.config.ESCAPEConfig method), 107  
 get\_global\_view() (escape.adapt.adaptation.DomainResourceManager method), 78, 183  
 get\_ifaces() (in module escape.util.misc), 130  
 get\_Link\_params() (escape.util.config.ESCAPEConfig method), 109  
 get\_managers() (escape.util.config.ESCAPEConfig method), 106  
 get\_mapper() (escape.util.config.ESCAPEConfig method), 106  
 get\_mapping\_enabled() (escape.util.config.ESCAPEConfig method), 105  
 get\_mapping\_processor() (escape.util.config.ESCAPEConfig method), 106  
 get\_mgr() (escape.adapt.adaptation.ComponentConfigurator method), 75, 180  
 get\_mininet\_topology() (escape.util.config.ESCAPEConfig method), 107  
 get\_mn\_network\_opts() (escape.util.config.ESCAPEConfig method), 107  
 get\_mn\_wrapper() (escape.adapt.adapters.InternalMininetAdapter method), 80, 185  
 get\_network() (escape.infr.topology.ESCAPENetworkBuilder method), 95, 201  
 get\_port() (escape.util.nffg\_elements.NFFGModel method), 151  
 get\_processor\_enabled() (escape.util.config.ESCAPEConfig method), 106  
 get\_project\_root\_dir() (escape.util.config.ESCAPEConfig static method), 105  
 get\_property() (escape.util.nffg\_elements.Port method), 147

get\_resource\_info() (escape.adapt.adaptation.DomainVirtualizer method), 77, 182  
get\_resource\_info() (escape.orchest.virtualization\_mgmt.AbstractVirtualizer method), 72, 176  
get\_resource\_info() (escape.orchest.virtualization\_mgmt.GlobalViewVirtualizer method), 72, 177  
get\_resource\_info() (escape.orchest.virtualization\_mgmt.SingleBiSBiSVirtualizer method), 73, 177  
get\_result() (escape.service.sas\_API.ServiceLayerAPI method), 57, 161  
get\_result() (escape.util.api.RequestCache method), 101  
get\_ros\_agent\_address() (escape.util.config.ESCAPEConfig method), 107  
get\_ros\_agent\_class() (escape.util.config.ESCAPEConfig method), 107  
get\_ros\_agent\_prefix() (escape.util.config.ESCAPEConfig method), 107  
get\_SAP\_params() (escape.util.config.ESCAPEConfig method), 108  
get\_SAP\_xterms() (escape.util.config.ESCAPEConfig method), 108  
get\_sas\_api\_address() (escape.util.config.ESCAPEConfig method), 107  
get\_sas\_api\_class() (escape.util.config.ESCAPEConfig method), 107  
get\_sas\_api\_prefix() (escape.util.config.ESCAPEConfig method), 107  
get\_sdn\_topology() (escape.util.config.ESCAPEConfig method), 107  
get\_strategy() (escape.util.config.ESCAPEConfig method), 105  
get\_Switch\_params() (escape.util.config.ESCAPEConfig method), 108  
get\_TAGS\_of\_starting\_flows() (escape.util.nffg.NFFGToolBox static method), 141  
get\_threaded() (escape.util.config.ESCAPEConfig method), 106  
get\_topo\_desc() (escape.infr.topology.AbstractTopology static method), 93, 199  
get\_topo\_desc() (escape.infr.topology.FallbackDynamicTopology static method), 93, 199  
get\_topo\_desc() (escape.infr.topology.FallbackStaticTopology static method), 93, 199  
get\_topology\_resource() (escape.adapt.adapters.InternalMininetAdapter method), 80, 186  
get\_topology\_resource() (escape.adapt.adapters.InternalPOXAdapter method), 79, 185  
get\_topology\_resource() (escape.adapt.adapters.OpenStackRESTAdapter method), 84, 190  
get\_topology\_resource() (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
get\_topology\_resource() (escape.adapt.adapters.SDNDomainPOXAdapter method), 80, 185  
get\_topology\_resource() (escape.adapt.adapters.SDNDomainTopoAdapter method), 81, 186  
get\_topology\_resource() (escape.adapt.adapters.UniversalNodeRESTAdapter method), 85, 190  
get\_topology\_resource() (escape.adapt.adapters.VNFStarterAdapter method), 81, 187  
get\_topology\_resource() (escape.util.domain.AbstractESCAPEAdapter method), 118  
get\_topology\_resource() (escape.util.domain.AbstractOFControllerAdapter method), 119  
get\_virtual\_view() (escape.orchest.virtualization\_mgmt.VirtualizerManager method), 73, 178  
get\_wrapper() (escape.orchest.policy\_enforcement.PolicyEnforcementMetaClass class method), 63, 168  
getDecomps() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 166  
GetGlobalResInfoEvent (class in escape.orchest.ros\_API), 66, 171  
getNexus() (escape.util.pox\_extension.ExtendedOFCConnectionArbiter method), 155  
getNF() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 165  
getSingleDecomp() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 166  
GetVirtResInfoEvent (class in escape.service.sas\_API), 55, 159  
getVNFInfo() (escape.adapt.adapters.VNFStarterAdapter method), 83, 188  
getVNFInfo() (escape.util.domain.VNFStarterAPI method), 121  
GlobalResInfoEvent (class in escape.adapt.cas\_API), 85, 191  
GlobalViewVirtualizer (class in escape.orchest.virtualization\_mgmt), 72, 177

|

id (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
in\_initiated (escape.util.config.ESCAPEConfig attribute), 104  
IN\_PROGRESS (escape.util.api.RequestCache attribute), 100  
info() (escape.util.domain.AbstractDomainManager method), 117  
INFRA (escape.util.nffg\_elements.Node attribute), 144  
infra\_neighbors() (escape.util.nffg.NFFG method), 140  
infra\_to\_dpid (escape.adapt.adapters.InternalPOXAdapter attribute), 79, 184  
infra\_to\_dpid (escape.adapt.adapters.SDNDomainPOXAdapter attribute), 80, 185  
infra\_to\_dpid (escape.util.domain.AbstractOFControllerAdapter attribute), 118  
InfraPort (class in escape.util.nffg\_elements), 147  
infras (escape.util.nffg.NFFG attribute), 136  
infrastructure (module), 196

InfrastructureLayerAPI (class in escape.infr.il\_API), 91, 197  
init() (escape.adapt.managers.InternalDomainManager method), 87, 193  
init() (escape.adapt.managers.OpenStackDomainManager method), 90, 195  
init() (escape.adapt.managers.RemoteESCAPEDomainManager method), 89, 195  
init() (escape.adapt.managers.SDNDomainManager method), 89, 194  
init() (escape.adapt.managers.UniversalNodeDomainManager method), 90, 196  
init() (escape.util.domain.AbstractDomainManager method), 116  
init() (escape.util.pox\_extension.ESCAPEInteractiveHelper static method), 155  
initialize() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 191  
initialize() (escape.infr.il\_API.InfrastructureLayerAPI method), 92, 197  
initialize() (escape.orchest.nfib\_mgmt.NFIBManager method), 62, 166  
initialize() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 172  
initialize() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
initialize() (escape.util.api.AbstractAPI method), 100  
INITIALIZING (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 119  
initiate\_service\_graph() (escape.service.sas\_orchestration.ServiceOrchestrator method), 58, 162  
initiated (escape.adapt.adaptation.ComponentConfigurator attribute), 76, 181  
INITIATED (escape.util.api.RequestCache attribute), 100  
initiateVNF() (escape.adapt.adapters.VNFStarterAdapter method), 82, 187  
initiateVNF() (escape.util.domain.VNFStarterAPI method), 120  
install\_domain() (escape.util.nffg.NFFGToolBox static method), 141  
install\_flowrule() (escape.util.domain.AbstractOFControllerAdapter method), 119  
install\_nffg() (escape.adapt.adaptation.ControllerAdapter method), 77, 182  
install\_nffg() (escape.adapt.managers.DockerDomainManager method), 91, 196  
install\_nffg() (escape.adapt.managers.InternalDomainManager method), 88, 193  
install\_nffg() (escape.adapt.managers.OpenStackDomainManager method), 90, 195  
install\_nffg() (escape.adapt.managers.RemoteESCAPEDomainManager method), 89, 195  
install\_nffg() (escape.adapt.managers.SDNDomainManager method), 89, 194  
install\_nffg() (escape.adapt.managers.UniversalNodeDomainManager method), 90, 196  
install\_nffg() (escape.util.domain.AbstractDomainManager method), 117  
InstallationFinishedEvent (class in escape.adapt.cas\_API), 85, 191  
InstallNFFGEVENT (class in escape.orchest.ros\_API), 66, 170  
instantiate\_nffg() (escape.orchest.ros\_orchestration.ResourceOrchestrator method), 65, 169  
InstantiateNFFGEVENT (class in escape.service.sas\_API), 55, 159  
InstantiationFinishedEvent (class in escape.orchest.ros\_API), 66, 171  
InternalControllerProxy (class in escape.infr.topology), 93, 199  
InternalDomainManager (class in escape.adapt.managers), 87, 193  
InternalMininetAdapter (class in escape.adapt.adapters), 80, 185  
InternalPOXAdapter (class in escape.adapt.adapters), 79, 184  
is\_layer\_loaded() (escape.util.config.ESCAPEConfig method), 105  
is\_started() (escape.adapt.adaptation.ComponentConfigurator method), 75, 180

## L

launch() (in module adaptation), 179  
launch() (in module infrastructure), 196  
launch() (in module orchestration), 164  
launch() (in module service), 156  
launch() (in module unify), 155  
LAYER\_ID (escape.service.sas\_API.ServiceLayerAPI attribute), 56, 160  
LAYERS (escape.util.config.ESCAPEConfig attribute), 104  
Link (class in escape.util.nffg\_elements), 145  
links (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
links (escape.util.nffg.NFFG attribute), 136  
load() (escape.util.nffg\_elements.EdgeLink method), 150  
load() (escape.util.nffg\_elements.EdgeReq method), 151  
load() (escape.util.nffg\_elements.EdgeSGLink method), 150  
load() (escape.util.nffg\_elements.Element method), 143  
load() (escape.util.nffg\_elements.Flowrule method), 146

load() (escape.util.nffg\_elements.InfraPort method), 148  
 load() (escape.util.nffg\_elements.Link method), 145  
 load() (escape.util.nffg\_elements.NFFGModel method), 153  
 load() (escape.util.nffg\_elements.Node method), 145  
 load() (escape.util.nffg\_elements.NodeInfra method), 149  
 load() (escape.util.nffg\_elements.NodeNF method), 148  
 load() (escape.util.nffg\_elements.NodeResource method), 145  
 load() (escape.util.nffg\_elements.NodeSAP method), 148  
 load() (escape.util.nffg\_elements.Persistable method), 143  
 load() (escape.util.nffg\_elements.Port method), 147  
 load\_component() (escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
 load\_config() (escape.util.config.ESCAPEConfig method), 104  
 load\_default\_mgrs() (escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
 load\_internal\_mgr() (escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
 log (escape.orchest.ros\_API.CfOrRequestHandler attribute), 67, 171  
 log (escape.orchest.ros\_API.ROSAgentRequestHandler attribute), 67, 172  
 log (escape.service.sas\_API.ServiceRequestHandler attribute), 55, 159  
 log (escape.util.api.AbstractRequestHandler attribute), 102  
 log\_error() (escape.util.api.AbstractRequestHandler method), 103  
 log\_full\_message() (escape.util.api.AbstractRequestHandler method), 103  
 log\_message() (escape.util.api.AbstractRequestHandler method), 103

## M

manager (escape.util.netconf.AbstractNETCONFAdapter attribute), 131  
 map() (escape.orchest.ros\_mapping.ESCAPEMappingStrategy class method), 70, 174  
 map() (escape.service.sas\_mapping.DefaultServiceMappingStrategy class method), 53, 158  
 map() (escape.util.mapping.AbstractMappingStrategy class method), 124  
 merge\_domain\_into\_dov() (escape.adapt.adaptation.DomainVirtualizer method), 78, 183  
 merge\_domains() (escape.util.nffg.NFFGToolBox static method), 141  
 merge\_duplicated\_links() (escape.util.nffg.NFFG method), 140  
 MissingGlobalViewEvent (class in escape.orchest.virtualization\_mgmt), 72, 176  
 MissingVirtualViewEvent (class in escape.service.sas\_orchestration), 58, 162  
 MOD (escape.util.nffg\_elements.Element attribute), 143  
 MOV (escape.util.nffg\_elements.Element attribute), 143  
 msg (escape.util.api.RESTError attribute), 101

## N

name (escape.adapt.adaptation.DomainVirtualizer attribute), 77, 182  
 name (escape.adapt.adapters.InternalMininetAdapter attribute), 80, 185  
 name (escape.adapt.adapters.InternalPOXAdapter attribute), 79, 184  
 name (escape.adapt.adapters.OpenStackRESTAdapter attribute), 84, 189  
 name (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter attribute), 84, 189  
 name (escape.adapt.adapters.SDNDomainPOXAdapter attribute), 80, 185  
 name (escape.adapt.adapters.SDNDomainTopoAdapter attribute), 81, 186  
 name (escape.adapt.adapters.UniversalNodeRESTAdapter attribute), 84, 190  
 name (escape.adapt.adapters.VNFStertAdapter attribute), 81, 186  
 name (escape.adapt.managers.DockerDomainManager attribute), 91, 196  
 name (escape.adapt.managers.InternalDomainManager attribute), 87, 193  
 name (escape.adapt.managers.OpenStackDomainManager attribute), 90, 195  
 name (escape.adapt.managers.RemoteESCAPEDomainManager attribute), 89, 195  
 name (escape.adapt.managers.SDNDomainManager attribute), 88, 194  
 name (escape.adapt.managers.UniversalNodeDomainManager attribute), 90, 196  
 name (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
 name (escape.util.domain.AbstractDomainManager attribute), 116  
 name (escape.util.domain.AbstractESCAPEAdapter attribute), 118  
 NAMESPACE (escape.util.nffg\_elements.NFFGModel attribute), 151  
 NETCONF\_NAMESPACE (escape.util.netconf.AbstractNETCONFAdapter attribute), 130  
 network (escape.infr.topology.ESCAPENetworkBridge attribute), 94, 200

network (escape.util.nffg.NFFG attribute), 136  
 NF (escape.util.nffg\_elements.Node attribute), 144  
 NFFG (class in escape.util.nffg), 135  
 NFFGConverter (class in escape.util.conversion), 114  
 NFFGManager (class in escape.orchest.ros\_orchestration), 65, 169  
 NFFGMappingFinishedEvent (class in escape.orchest.ros\_mapping), 70, 175  
 NFFGModel (class in escape.util.nffg\_elements), 151  
 NFFGToolBox (class in escape.util.nffg), 141  
 NFIBManager (class in escape.orchest.nfib\_mgmt), 60, 164  
 nfs (escape.util.nffg.NFFG attribute), 136  
 Node (class in escape.util.nffg\_elements), 144  
 node (escape.util.nffg\_elements.Port attribute), 146  
 NodeInfra (class in escape.util.nffg\_elements), 148  
 NodeNF (class in escape.util.nffg\_elements), 148  
 NodeResource (class in escape.util.nffg\_elements), 145  
 nodes (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
 nodes (escape.util.nffg\_elements.NFFGModel attribute), 151  
 NodeSAP (class in escape.util.nffg\_elements), 148

## O

OpenFlowBridge (class in escape.util.pox\_extension), 154  
 OpenStackAPI (class in escape.util.domain), 121  
 OpenStackDomainManager (class in escape.adapt.managers), 90, 195  
 OpenStackRESTAdapter (class in escape.adapt.adapters), 84, 189  
 OPERATION\_ADD (escape.util.nffg.NFFG attribute), 136  
 OPERATION\_DEL (escape.util.nffg.NFFG attribute), 136  
 OPERATION\_MOD (escape.util.nffg.NFFG attribute), 136  
 OPERATION\_MOV (escape.util.nffg.NFFG attribute), 136  
 operations() (escape.util.api.AbstractRequestHandler method), 104  
 orchestrate() (escape.util.mapping.AbstractMapper method), 126  
 orchestration (module), 163  
 ORGANIZATION (escape.util.nffg\_elements.NFFGModel attribute), 151

## P

parse() (escape.util.conversion.Virtualizer3BasedNFFGBuilder class method), 110  
 parse() (escape.util.nffg.AbstractNFFG class method), 135  
 parse() (escape.util.nffg.NFFG class method), 140  
 parse() (escape.util.nffg\_elements.Persistable class method), 143  
 parse\_from\_Virtualizer3() (escape.util.conversion.NFFGConverter method), 114  
 persist() (escape.util.nffg\_elements.EdgeLink method), 150  
 persist() (escape.util.nffg\_elements.EdgeReq method), 151  
 persist() (escape.util.nffg\_elements.EdgeSGLink method), 150  
 persist() (escape.util.nffg\_elements.Element method), 143  
 persist() (escape.util.nffg\_elements.Flowrule method), 146  
 persist() (escape.util.nffg\_elements.InfraPort method), 147  
 persist() (escape.util.nffg\_elements.Link method), 145  
 persist() (escape.util.nffg\_elements.NFFGModel method), 153  
 persist() (escape.util.nffg\_elements.Node method), 145  
 persist() (escape.util.nffg\_elements.NodeInfra method), 149  
 persist() (escape.util.nffg\_elements.NodeNF method), 148  
 persist() (escape.util.nffg\_elements.NodeResource method), 145  
 persist() (escape.util.nffg\_elements.NodeSAP method), 148  
 persist() (escape.util.nffg\_elements.Persistable method), 142  
 persist() (escape.util.nffg\_elements.Port method), 147  
 Persistable (class in escape.util.nffg\_elements), 142  
 ping() (escape.adapt.adapters.OpenStackRESTAdapter method), 84, 190  
 ping() (escape.adapt.adapters.RemoteESCAPEv2RESTAdapter method), 84, 189  
 ping() (escape.adapt.adapters.UniversalNodeRESTAdapter method), 85, 190

ping() (escape.util.api.AbstractRequestHandler method), 103  
ping() (escape.util.domain.DefaultDomainRESTAPI method), 121  
ping() (escape.util.pox\_extension.ESCAPEInteractiveHelper method), 155  
PolicyEnforcement (class in escape.orchest.policy\_enforcement), 63, 168  
PolicyEnforcementError, 62, 167  
PolicyEnforcementMetaClass (class in escape.orchest.policy\_enforcement), 62, 167  
poll() (escape.util.domain.AbstractDomainManager method), 117  
poll() (escape.util.domain.AbstractESCAPEAdapter method), 118  
POLL\_INTERVAL (escape.util.domain.AbstractDomainManager attribute), 116  
Port (class in escape.util.nffg\_elements), 146  
PORT\_ABSTRACT (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
PORT\_SAP (escape.util.conversion.Virtualizer3BasedNFFGBuilder attribute), 110  
PortContainer (class in escape.util.nffg\_elements), 143  
POST (escape.util.domain.AbstractRESTAdapter attribute), 122  
post\_mapping\_exec() (escape.util.mapping.AbstractMappingDataProcessor method), 124  
post\_mapping\_exec() (escape.util.mapping.PrePostMapNotifier method), 125  
post\_mapping\_exec() (escape.util.mapping.ProcessorSkipper method), 125  
post\_sanity\_check() (escape.orchest.policy\_enforcement.PolicyEnforcement class method), 64, 168  
PostMapEvent (class in escape.util.mapping), 125  
pre\_mapping\_exec() (escape.util.mapping.AbstractMappingDataProcessor method), 124  
pre\_mapping\_exec() (escape.util.mapping.PrePostMapNotifier method), 125  
pre\_mapping\_exec() (escape.util.mapping.ProcessorSkipper method), 125  
pre\_sanity\_check() (escape.orchest.policy\_enforcement.PolicyEnforcement class method), 64, 168  
PREFIX (escape.util.nffg\_elements.NFFGModel attribute), 151  
PreMapEvent (class in escape.util.mapping), 125  
PrePostMapNotifier (class in escape.util.mapping), 125  
ProcessorError, 124  
ProcessorSkipper (class in escape.util.mapping), 125

## Q

quit\_with\_error() (in module escape.util.misc), 129

## R

RemoteESCAPEDomainManager (class in escape.adapt.managers), 89, 194  
RemoteESCAPEv2API (class in escape.util.domain), 122  
RemoteESCAPEv2RESTAdapter (class in escape.adapt.adapters), 84, 189  
remove() (escape.util.nffg\_elements.PortContainer method), 144  
remove\_junks() (in module escape.util.misc), 130  
removeDecomp() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 166  
removeGraphDB() (escape.orchest.nfib\_mgmt.NFIBManager method), 62, 166  
removeNF() (escape.adapt.adapters.VNFStarterAdapter method), 84, 189  
removeNF() (escape.orchest.nfib\_mgmt.NFIBManager method), 60, 165  
removeRelationship() (escape.orchest.nfib\_mgmt.NFIBManager method), 61, 165  
reqs (escape.util.nffg.NFFG attribute), 136  
request\_perm (escape.orchest.ros\_API.CfOrRequestHandler attribute), 67, 171  
request\_perm (escape.orchest.ros\_API.ROSAgentRequestHandler attribute), 67, 172  
request\_perm (escape.service.sas\_API.ServiceRequestHandler attribute), 55, 159  
request\_perm (escape.util.api.AbstractRequestHandler attribute), 102  
RequestCache (class in escape.util.api), 100  
REQUIREMENT (escape.util.nffg\_elements.Link attribute), 145  
reset\_domains\_after\_shutdown() (escape.util.config.ESCAPEConfig method), 107  
ResourceOrchestrationAPI (class in escape.orchest.ros\_API), 67, 172  
ResourceOrchestrationMapper (class in escape.orchest.ros\_mapping), 70, 175  
ResourceOrchestrator (class in escape.orchest.ros\_orchestration), 65, 169  
restart\_polling() (escape.util.domain.AbstractDomainManager method), 117  
RESTError, 101  
RESTServer (class in escape.util.api), 101  
result() (escape.service.sas\_API.ServiceRequestHandler method), 55, 160

resume() (escape.util.domain.AbstractDomainManager method), 117  
retrieve\_mapped\_path() (escape.util.nffg.NFFGToolBox static method), 141  
RFC  
    RFC 4741, 130  
    RFC 6241, 131  
ROSAgentRequestHandler (class in escape.orchest.ros\_API), 67, 171  
rpc\_mapper (escape.orchest.ros\_API.CfOrRequestHandler attribute), 67, 171  
rpc\_mapper (escape.orchest.ros\_API.ROSAgentRequestHandler attribute), 67, 172  
rpc\_mapper (escape.util.api.AbstractRequestHandler attribute), 102  
RPC\_NAMESPACE (escape.adapt.adapters.VNFStarterAdapter attribute), 81, 186  
RPC\_NAMESPACE (escape.util.netconf.AbstractNETCONFAdapter attribute), 130  
run() (escape.util.api.RESTServer method), 101  
run() (escape.util.domain.AbstractDomainManager method), 116  
run\_cmd() (in module escape.util.misc), 128  
run\_silent() (in module escape.util.misc), 128  
running\_nfs() (escape.util.nffg.NFFG method), 140  
runXTerms() (escape.infr.topology.ESCAPENetworkBridge method), 94, 200

## S

s\_CONNECTED (escape.util.domain.VNFStarterAPI.ConnectedStatus attribute), 120  
s\_DISCONNECTED (escape.util.domain.VNFStarterAPI.ConnectedStatus attribute), 120  
s\_FAILED (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 119  
s\_INITIALIZING (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 119  
s\_UP\_AND\_RUNNING (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 120  
sanity\_check() (escape.orchest.virtualization\_mgmt.AbstractVirtualizer method), 72, 177  
SAP (escape.util.nffg\_elements.Node attribute), 144  
saps (escape.adapt.adapters.InternalPOXAdapter attribute), 79, 184  
saps (escape.util.domain.AbstractOFControllerAdapter attribute), 118  
saps (escape.util.nffg.NFFG attribute), 136  
save() (escape.orchest.ros\_orchestration.NFFGManager method), 65, 170  
save() (escape.service.sas\_orchestration.SGManager method), 59, 163  
schedule\_as\_coop\_task() (in module escape.util.misc), 128  
schedule\_delayed\_as\_coop\_task() (in module escape.util.misc), 128  
SDNDomainManager (class in escape.adapt.managers), 88, 194  
SDNDomainPOXAdapter (class in escape.adapt.adapters), 80, 185  
SDNDomainTopoAdapter (class in escape.adapt.adapters), 80, 186  
send\_acknowledge() (escape.util.api.AbstractRequestHandler method), 103  
send\_error() (escape.util.api.AbstractRequestHandler method), 103  
send\_no\_error() (escape.util.domain.AbstractRESTAdapter method), 122  
send\_request() (escape.util.domain.AbstractRESTAdapter method), 122  
send\_REST\_headers() (escape.util.api.AbstractRequestHandler method), 103  
server\_version (escape.util.api.AbstractRequestHandler attribute), 102  
service (module), 156  
ServiceGraphMapper (class in escape.service.sas\_mapping), 54, 158  
ServiceLayerAPI (class in escape.service.sas\_API), 56, 160  
ServiceOrchestrator (class in escape.service.sas\_orchestration), 58, 162  
ServiceRequestHandler (class in escape.service.sas\_API), 55, 159  
set\_domain\_as\_global\_view() (escape.adapt.adaptation.DomainVirtualizer method), 77, 183  
set\_in\_progress() (escape.util.api.RequestCache method), 101  
set\_layer\_loaded() (escape.util.config.ESCAPEConfig method), 105  
set\_result() (escape.util.api.RequestCache method), 101  
setdefault() (escape.util.nffg\_elements.Element method), 143  
sg (escape.util.mapping.PreMapEvent attribute), 125  
SG (escape.util.nffg\_elements.Link attribute), 145  
sg() (escape.service.sas\_API.ServiceRequestHandler method), 55, 160  
sg\_hops (escape.util.nffg.NFFG attribute), 136  
SGManager (class in escape.service.sas\_orchestration), 58, 162  
SGMappingFinishedEvent (class in escape.service.sas\_mapping), 54, 158

short\_name (escape.util.nffg\_elements.Node attribute), 144  
shutdown() (escape.adapt.adaptation.ControllerAdapter method), 76, 182  
shutdown() (escape.adapt.cas\_API.ControllerAdaptationAPI method), 86, 191  
shutdown() (escape.infr.il\_API.InfrastructureLayerAPI method), 92, 197  
shutdown() (escape.orchest.ros\_API.ResourceOrchestrationAPI method), 68, 172  
shutdown() (escape.service.sas\_API.ServiceLayerAPI method), 56, 160  
shutdown() (escape.util.api.AbstractAPI method), 100  
SimpleStandaloneHelper (class in escape.util.misc), 129  
SingleBiSBISSVirtualizer (class in escape.orchest.virtualization\_mgmt), 72, 177  
Singleton (class in escape.util.misc), 129  
split\_domains() (escape.util.nffg.NFFGToolBox static method), 141  
start() (escape.util.api.RESTServer method), 101  
start\_mgr() (escape.adapt.adaptation.ComponentConfigurator method), 75, 180  
start\_network() (escape.infr.topology.ESCAPENetworkBridge method), 94, 200  
start\_polling() (escape.util.domain.AbstractDomainManager method), 117  
start\_polling() (escape.util.domain.AbstractESCAPEAdapter method), 118  
startVNF() (escape.adapt.adapters.VNFStarterAdapter method), 83, 188  
startVNF() (escape.util.domain.VNFStarterAPI method), 120  
STATIC (escape.util.nffg\_elements.Link attribute), 145  
static\_prefix (escape.orchest.ros\_API.CfOrRequestHandler attribute), 67, 171  
static\_prefix (escape.orchest.ros\_API.ROSAgentRequestHandler attribute), 67, 172  
static\_prefix (escape.util.api.AbstractRequestHandler attribute), 102  
stop() (escape.util.api.RESTServer method), 101  
stop\_initiated\_mgrs() (escape.adapt.adaptation.ComponentConfigurator method), 76, 181  
stop\_mgr() (escape.adapt.adaptation.ComponentConfigurator method), 75, 180  
stop\_network() (escape.infr.topology.ESCAPENetworkBridge method), 94, 200  
stop\_polling() (escape.util.domain.AbstractDomainManager method), 117  
stop\_polling() (escape.util.domain.AbstractESCAPEAdapter method), 118  
stopVNF() (escape.adapt.adapters.VNFStarterAdapter method), 83, 188  
stopVNF() (escape.util.domain.VNFStarterAPI method), 121  
SUCCESS (escape.util.api.RequestCache attribute), 100  
suspend() (escape.util.domain.AbstractDomainManager method), 116

## T

test\_networkx\_mod() (in module escape.util.nffg\_elements), 154  
test\_parse\_load() (in module escape.util.nffg\_elements), 154  
test\_topo\_os() (in module escape.util.conversion), 114  
test\_topo\_un() (in module escape.util.conversion), 114  
test\_virtualizer3\_based\_builder() (in module escape.util.conversion), 114  
test\_xml\_based\_builder() (in module escape.util.conversion), 114  
topology() (escape.service.sas\_API.ServiceRequestHandler method), 56, 160  
TopologyBuilderException, 95, 200  
TopologyLoadException, 79, 184  
TYPE (escape.infr.topology.AbstractTopology attribute), 93, 199  
TYPE (escape.infr.topology.FallbackDynamicTopology attribute), 93, 199  
TYPE (escape.infr.topology.FallbackStaticTopology attribute), 93, 199  
TYPE (escape.util.domain.DomainChangedEvent attribute), 115  
TYPE (escape.util.nffg\_elements.NFFGModel attribute), 151  
TYPE (escape.util.nffg\_elements.Port attribute), 146  
TYPE\_BISBIS (escape.util.nffg\_elements.NodeInfra attribute), 148  
TYPE\_EE (escape.util.nffg\_elements.NodeInfra attribute), 148  
TYPE\_EE\_LOCAL (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
TYPE\_EE\_REMOTE (escape.infr.topology.ESCAPENetworkBuilder attribute), 95, 201  
TYPE\_INFRA (escape.util.nffg.NFFG attribute), 136  
TYPE\_INFRA\_BISBIS (escape.util.nffg.NFFG attribute), 136  
TYPE\_INFRA\_EE (escape.util.nffg.NFFG attribute), 136  
TYPE\_INFRA\_SDN\_SW (escape.util.nffg.NFFG attribute), 136  
TYPE\_INFRA\_STATIC\_EE (escape.util.nffg.NFFG attribute), 136

TYPE\_LINK\_DYNAMIC (escape.util.nffg.NFFG attribute), 136  
TYPE\_LINK\_REQUIREMENT (escape.util.nffg.NFFG attribute), 136  
TYPE\_LINK\_SG (escape.util.nffg.NFFG attribute), 136  
TYPE\_LINK\_STATIC (escape.util.nffg.NFFG attribute), 136  
TYPE\_NF (escape.util.nffg.NFFG attribute), 136  
TYPE\_SAP (escape.util.nffg.NFFG attribute), 136  
TYPE\_SDN\_SWITCH (escape.util.nffg\_elements.NodeInfra attribute), 148  
TYPE\_STATIC\_EE (escape.util.nffg\_elements.NodeInfra attribute), 148  
TYPES (escape.orchest.virtualization\_mgmt.VirtualizerManager attribute), 73, 178

## U

unescape\_output\_hack() (escape.util.conversion.NFFGConverter static method), 114  
unify (module), 155  
UniversalNodeAPI (class in escape.util.domain), 121  
UniversalNodeDomainManager (class in escape.adapt.managers), 90, 195  
UniversalNodeRESTAdapter (class in escape.adapt.adapters), 84, 190  
UNKNOWN (escape.util.api.RequestCache attribute), 100  
UP\_AND\_RUNNING (escape.util.domain.VNFStarterAPI.VNFStatus attribute), 120  
update() (escape.util.nffg\_elements.Element method), 143  
update\_connection\_params() (escape.adapt.adapters.VNFStarterAdapter method), 82, 187  
update\_domain\_resource() (escape.adapt.adaptation.DomainResourceManager method), 78, 183  
update\_domain\_view() (escape.adapt.adaptation.DomainVirtualizer method), 78, 183  
update\_dov() (escape.adapt.adaptation.ControllerAdapter method), 77, 182  
update\_global\_view() (escape.adapt.adaptation.DomainVirtualizer method), 78, 183  
update\_local\_resource\_info() (escape.util.domain.AbstractDomainManager method), 117  
updateNF() (escape.orchest.nfib\_mgmt.NFIBManager method), 60, 165  
URL (escape.util.domain.AbstractRESTAdapter attribute), 122

## V

VERSION (escape.util.nffg\_elements.NFFGModel attribute), 151  
version() (escape.util.api.AbstractRequestHandler method), 104  
VirtResInfoEvent (class in escape.orchest.ros\_API), 66, 171  
virtual\_view (escape.service.sas\_orchestration.VirtualResourceManager attribute), 59, 163  
Virtualizer3BasedNFFGBuilder (class in escape.util.conversion), 109  
VirtualizerManager (class in escape.orchest.virtualization\_mgmt), 73, 178  
VirtualResourceManager (class in escape.service.sas\_orchestration), 59, 163  
VNF\_FORWARDER (escape.util.domain.VNFStarterAPI attribute), 119  
VNF\_HEADER\_COMP (escape.util.domain.VNFStarterAPI attribute), 119  
VNF\_HEADER\_DECOMP (escape.util.domain.VNFStarterAPI attribute), 119  
VNFStarterAdapter (class in escape.adapt.adapters), 81, 186  
VNFStarterAPI (class in escape.util.domain), 119  
VNFStarterAPI.ConnectedStatus (class in escape.util.domain), 120  
VNFStarterAPI.VNFStatus (class in escape.util.domain), 119



## D3.4 Annex II: Virtualizer3 Documentation

Release 3.0.0

Róbert Szabó (ETH), Raphael Vicente Rosa (ETH)

This project is co-funded  
by the European Union



# Chapter 1

## Namespace Index

### 1.1 Packages

Here are the packages with brief descriptions (if available):

gen .....	235
gen.baseclasses .....	235
gen.virtualizer3 .....	240

# Chapter 2

## Hierarchical Index

### 2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

object . . . . .	301
gen.baseclasses.Yang . . . . .	313
gen.baseclasses.FilterYang . . . . .	248
gen.baseclasses.Leaf . . . . .	282
gen.baseclasses.BooleanLeaf . . . . .	242
gen.baseclasses.Decimal64Leaf . . . . .	244
gen.baseclasses.IntLeaf . . . . .	279
gen.baseclasses.StringLeaf . . . . .	307
gen.baseclasses.Leafref . . . . .	285
gen.virtualizer3.Port_ref . . . . .	304
gen.baseclasses.ListedYang . . . . .	293
gen.virtualizer3.Flowentry . . . . .	250
gen.virtualizer3.Infra_node . . . . .	277
gen.virtualizer3.Link . . . . .	288
gen.virtualizer3.Node . . . . .	297
gen.virtualizer3.Port . . . . .	302
gen.baseclasses.ListYang . . . . .	295
gen.virtualizer3.FlowtableFlowtable . . . . .	252
gen.virtualizer3.GroupingFlowtable . . . . .	257
gen.virtualizer3.GroupingInfra_node . . . . .	262
gen.virtualizer3.Infra_node . . . . .	277
gen.virtualizer3.GroupingId_name . . . . .	258
gen.virtualizer3.GroupingFlowentry . . . . .	254
gen.virtualizer3.Flowentry . . . . .	250
gen.virtualizer3.GroupingId_name_type . . . . .	260
gen.virtualizer3.GroupingNode . . . . .	268
gen.virtualizer3.GroupingInfra_node . . . . .	262
gen.virtualizer3.Node . . . . .	297
gen.virtualizer3.GroupingLink . . . . .	263
gen.virtualizer3.Link . . . . .	288
gen.virtualizer3.GroupingPort . . . . .	273
gen.virtualizer3.Port . . . . .	302
gen.virtualizer3.Virtualizer . . . . .	309
gen.virtualizer3.GroupingLink_resource . . . . .	265
gen.virtualizer3.Link_resource . . . . .	290
gen.virtualizer3.GroupingLinks . . . . .	267

**Hierarchical Index**

gen.virtualizer3.GroupingNode . . . . .	268
gen.virtualizer3.Virtualizer . . . . .	309
gen.virtualizer3.GroupingNodes . . . . .	271
gen.virtualizer3.Nodes . . . . .	300
gen.virtualizer3.GroupingSoftware_resource . . . . .	275
gen.virtualizer3.Software_resource . . . . .	306
gen.virtualizer3.Infra_nodeCapabilities . . . . .	278
gen.virtualizer3.LinksLinks . . . . .	291
gen.virtualizer3.NodePorts . . . . .	298
gen.virtualizer3.VirtualizerNodes . . . . .	311

# Chapter 3

## Class Index

### 3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

gen.baseclasses.BooleanLeaf	Class defining Leaf with boolean extensions (e.g., True or False) . . . . .	242
gen.baseclasses.Decimal64Leaf	Class defining Leaf with decimal extensions (e.g., dec_range) . . . . .	244
gen.baseclasses.FilterYang		248
gen.virtualizer3.Flowentry		250
gen.virtualizer3.FlowtableFlowtable		252
gen.virtualizer3.GroupingFlowentry		254
gen.virtualizer3.GroupingFlowtable		257
gen.virtualizer3.GroupingId_name		258
gen.virtualizer3.GroupingId_name_type		260
gen.virtualizer3.GroupingInfra_node		262
gen.virtualizer3.GroupingLink		263
gen.virtualizer3.GroupingLink_resource		265
gen.virtualizer3.GroupingLinks		267
gen.virtualizer3.GroupingNode	Any node: infrastructure or NFs . . . . .	268
gen.virtualizer3.GroupingNodes		271
gen.virtualizer3.GroupingPort		273
gen.virtualizer3.GroupingSoftware_resource		275
gen.virtualizer3.Infra_node		277
gen.virtualizer3.Infra_nodeCapabilities		278
gen.baseclasses.IntLeaf	Class defining Leaf with integer extensions (e.g., range) . . . . .	279
gen.baseclasses.Leaf	Class defining Leaf basis with attributes and methods . . . . .	282
gen.baseclasses.Leafref	Class defining Leaf extensions for stringleaf when its data references other instances . . . . .	285
gen.virtualizer3.Link		288
gen.virtualizer3.Link_resource		290
gen.virtualizer3.LinksLinks		291
gen.baseclasses.ListedYang	Class defined for Virtualizer classes inherit when modeled as list . . . . .	293
gen.baseclasses.ListYang	Class to express list as dictionary . . . . .	295
gen.virtualizer3.Node		297
gen.virtualizer3.NodePorts		298
gen.virtualizer3.Nodes		300

**Class Index**

object . . . . .	301
gen.virtualizer3.Port . . . . .	302
gen.virtualizer3.Port_ref . . . . .	304
gen.virtualizer3.Software_resource . . . . .	306
gen.baseclasses.StringLeaf Class defining Leaf with string extensions . . . . .	307
gen.virtualizer3.Virtualizer Container for a single virtualizer . . . . .	309
gen.virtualizer3.VirtualizerNodes . . . . .	311
gen.baseclasses.Yang Class defining the root attributes and methods for all Virtualizer classes . . . . .	313

# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

baseclasses.py . . . . .	319
virtualizer3.py . . . . .	321

# Chapter 5

## Namespace Documentation

### 5.1 gen Namespace Reference

#### Namespaces

- [baseclasses](#)
- [virtualizer3](#)

### 5.2 gen.baseclasses Namespace Reference

#### Classes

- class [BooleanLeaf](#)  
*Class defining Leaf with boolean extensions (e.g., True or False)*
- class [Decimal64Leaf](#)  
*Class defining Leaf with decimal extensions (e.g., dec\_range)*
- class [FilterYang](#)
- class [IntLeaf](#)  
*Class defining Leaf with integer extensions (e.g., range)*
- class [Leaf](#)  
*Class defining Leaf basis with attributes and methods.*
- class [Leafref](#)  
*Class defining Leaf extensions for stringleaf when its data references other instances.*
- class [ListedYang](#)  
*Class defined for Virtualizer classes inherit when modeled as list.*
- class [ListYang](#)  
*Class to express list as dictionary.*
- class [StringLeaf](#)  
*Class defining Leaf with string extensions.*
- class [Yang](#)  
*Class defining the root attributes and methods for all Virtualizer classes.*

#### Functions

- def [\\_\\_init\\_\\_](#)
- def [get\\_type](#) (self)

- Returns class which references elements of \_data OrderedDict :param: - :return: Yang subclass.*
- def `set_type` (self, type)
 

*Sets class which references elements of \_data OrderedDict :param: Yang subclass :return: -.*
  - def `keys` (self)
 

*Returns indices of ListYang dictionary :param: - :return: list.*
  - def `values` (self)
 

*Returns values of ListYang dictionary :param: - :return: list.*
  - def `iterkeys` (self)
 

*Returns iterator of keys of ListYang dictionary :param: - :return: iterator.*
  - def `itervalues` (self)
 

*Returns iterator of values of ListYang dictionary :param: - :return: list.*
  - def `items` (self)
 

*Returns items of ListYang dictionary :param: - :return: list.*
  - def `iteritems` (self)
 

*Returns iterator of items of ListYang dictionary :param: - :return: list.*
  - def `has_key` (self, key)
 

*Returns if key is in ListYang dictionary :param key: string :return: boolean.*
  - def `has_value` (self, value)
 

*Returns if value is in ListYang dictionary values :param value: string or instance :return: boolean.*
  - def `length` (self)
 

*Returns length of ListYang dictionary :param: - :return: int.*
  - def `is_initialized` (self)
 

*Returns if ListYang dictionary contains elements :param: - :return: boolean.*
  - def `add` (self, item)
 

*Add single or a list of items :param item: a single ListedYang or a list of ListedYang derivates :return: item*
  - def `remove` (self, item)
 

*remove a single element from the list based on a key or a ListedYang :param item: key (single or composit) or a ListedYang :return: item*
  - def `__iter__` (self)
 

*Returns iterator of ListYang dict :param: - :return: iterator.*
  - def `next` (self)
 

*Go to next element of ListYang dictionary :param: - :return: -.*
  - def `__getitem__` (self, key)
 

*Returns ListYang value if key in dictionary :param key: string :return: instance.*
  - def `__setitem__` (self, key, value)
 

*Fill ListYang dict with key associated to value :param key: string :param value: string or instance :return: -.*
  - def `clear_data` (self)
 

*Clear ListYang dict :param: - :return: -.*
  - def `reduce` (self, reference)
 

*Check if all keys of reference are going to be reduced and erase their values if yes :param reference: ListYang :return: boolean.*
  - def `merge` (self, target)
 

*Add items of target if their keys do not exist in self instance :param target: ListYang :return: -.*
  - def `__eq__` (self, other)
 

*Check if dict of other ListYang is equal :param other: ListYang :return: boolean.*
  - def `contains_operation` (self, operation)
 

*Check if any of items have operation set :param operation: string :return: boolean.*
  - def `set_operation`

*Set operation for all of items in ListYang dict' :param operation: string :return: -.*
  - def `bind`

## Variables

- string `__copyright__` = "Copyright Ericsson Hungary Ltd., 2015"
- `_data`
- `_type`

### 5.2.1 Function Documentation

#### 5.2.1.1 def gen.baseclasses.`__eq__`( `self, other` )

Check if dict of other `ListYang` is equal :param other: `ListYang` :return: boolean.

Definition at line 1374 of file `baseclasses.py`.

#### 5.2.1.2 def gen.baseclasses.`__getitem__`( `self, key` )

Returns `ListYang` value if key in dictionary :param key: string :return: instance.

Definition at line 1307 of file `baseclasses.py`.

#### 5.2.1.3 def gen.baseclasses.`__init__`( `self, tag, parent=None, type=None` )

Definition at line 1118 of file `baseclasses.py`.

#### 5.2.1.4 def gen.baseclasses.`__iter__`( `self` )

Returns iterator of `ListYang` dict :param: - :return: iterator.

Definition at line 1289 of file `baseclasses.py`.

#### 5.2.1.5 def gen.baseclasses.`__setitem__`( `self, key, value` )

Fill `ListYang` dict with key associated to value :param key: string :param value: string or instance :return: -.

Definition at line 1322 of file `baseclasses.py`.

#### 5.2.1.6 def gen.baseclasses.add( `self, item` )

add single or a list of items :param item: a single `ListedYang` or a list of `ListedYang` derivates :return: item

Definition at line 1239 of file `baseclasses.py`.

#### 5.2.1.7 def gen.baseclasses.bind( `self, relative=False` )

Definition at line 1402 of file `baseclasses.py`.

#### 5.2.1.8 def gen.baseclasses.clear\_data( `self` )

Clear `ListYang` dict :param: - :return: -.

Definition at line 1332 of file `baseclasses.py`.

**5.2.1.9 def gen.baseclasses.contains\_operation ( self, operation )**

Check if any of items have operation set :param operation: string :return: boolean.

Definition at line 1385 of file baseclasses.py.

**5.2.1.10 def gen.baseclasses.get\_type ( self )**

Returns class which references elements of \_data OrderedDict :param: - :return: Yang subclass.

Definition at line 1129 of file baseclasses.py.

**5.2.1.11 def gen.baseclasses.has\_key ( self, key )**

Returns if key is in ListYang dictionary :param key: string :return: boolean.

Definition at line 1201 of file baseclasses.py.

**5.2.1.12 def gen.baseclasses.has\_value ( self, value )**

Returns if value is in ListYang dictionary values :param value: string or instance :return: boolean.

Definition at line 1210 of file baseclasses.py.

**5.2.1.13 def gen.baseclasses.is\_initialized ( self )**

Returns if ListYang dictionary contains elements :param: - :return: boolean.

Definition at line 1228 of file baseclasses.py.

**5.2.1.14 def gen.baseclasses.items ( self )**

Returns items of ListYang dictionary :param: - :return: list.

Definition at line 1183 of file baseclasses.py.

**5.2.1.15 def gen.baseclasses.iteritems ( self )**

Returns iterator of items of ListYang dictionary :param: - :return: list.

Definition at line 1192 of file baseclasses.py.

**5.2.1.16 def gen.baseclasses.iterkeys ( self )**

Returns iterator of keys of ListYang dictionary :param: - :return: iterator.

Definition at line 1165 of file baseclasses.py.

**5.2.1.17 def gen.baseclasses.itervalues ( self )**

Returns iterator of values of ListYang dictionary :param: - :return: list.

Definition at line 1174 of file baseclasses.py.

**5.2.1.18 def gen.baseclasses.keys ( self )**

Returns indices of [ListYang](#) dictionary :param: - :return: list.

Definition at line 1147 of file baseclasses.py.

**5.2.1.19 def gen.baseclasses.length ( self )**

Returns length of [ListYang](#) dictionary :param: - :return: int.

Definition at line 1219 of file baseclasses.py.

**5.2.1.20 def gen.baseclasses.merge ( self, target )**

Add items of target if their keys do not exist in self instance :param target: [ListYang](#) :return: -.

Definition at line 1359 of file baseclasses.py.

**5.2.1.21 def gen.baseclasses.next ( self )**

Go to next element of [ListYang](#) dictionary :param: - :return: -.

Definition at line 1298 of file baseclasses.py.

**5.2.1.22 def gen.baseclasses.reduce ( self, reference )**

Check if all keys of reference are going to be reduced and erase their values if yes :param reference: [ListYang](#) :return: boolean.

Definition at line 1341 of file baseclasses.py.

**5.2.1.23 def gen.baseclasses.remove ( self, item )**

remove a single element from the list based on a key or a [ListedYang](#) :param item: key (single or composit) or a [ListedYang](#) :return: item

Definition at line 1259 of file baseclasses.py.

**5.2.1.24 def gen.baseclasses.set\_operation ( self, operation = "delete" )**

Set operation for all of items in [ListYang](#) dict' :param operation: string :return: -.

Definition at line 1397 of file baseclasses.py.

**5.2.1.25 def gen.baseclasses.set\_type ( self, type )**

Sets class which references elements of \_data OrderedDict :param: [Yang](#) subclass :return: -.

Definition at line 1138 of file baseclasses.py.

**5.2.1.26 def gen.baseclasses.values ( self )**

Returns values of [ListYang](#) dictionary :param: - :return: list.

Definition at line 1156 of file baseclasses.py.

## 5.2.2 Variable Documentation

### 5.2.2.1 string gen.baseclasses.\_\_copyright\_\_ = "Copyright Ericsson Hungary Ltd., 2015"

Definition at line 15 of file baseclasses.py.

### 5.2.2.2 gen.baseclasses.\_data

Definition at line 1120 of file baseclasses.py.

### 5.2.2.3 gen.baseclasses.\_type

Definition at line 1121 of file baseclasses.py.

## 5.3 gen.virtualizer3 Namespace Reference

### Classes

- class [Flowentry](#)
- class [FlowtableFlowtable](#)
- class [GroupingFlowentry](#)
- class [GroupingFlowtable](#)
- class [GroupingId\\_name](#)
- class [GroupingId\\_name\\_type](#)
- class [GroupingInfra\\_node](#)
- class [GroupingLink](#)
- class [GroupingLink\\_resource](#)
- class [GroupingLinks](#)
- class [GroupingNode](#)
  - Any node: infrastructure or NFs.*
- class [GroupingNodes](#)
- class [GroupingPort](#)
- class [GroupingSoftware\\_resource](#)
- class [Infra\\_node](#)
- class [Infra\\_nodeCapabilities](#)
- class [Link](#)
- class [Link\\_resource](#)
- class [LinksLinks](#)
- class [Node](#)
- class [NodePorts](#)
- class [Nodes](#)
- class [Port](#)
- class [Port\\_ref](#)
- class [Software\\_resource](#)
- class [Virtualizer](#)
  - Container for a single virtualizer.*
- class [VirtualizerNodes](#)

### Variables

- string [\\_\\_copyright\\_\\_](#) = "Copyright Ericsson Hungary Ltd., 2015"

### 5.3.1 Variable Documentation

5.3.1.1 string gen.virtualizer3.\_\_copyright\_\_ = "Copyright Ericsson Hungary Ltd., 2015"

Definition at line 15 of file virtualizer3.py.

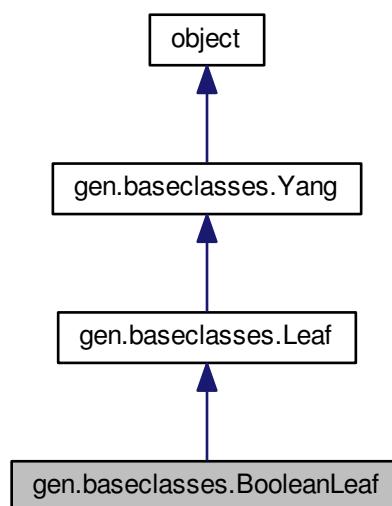
# Chapter 6

## Class Documentation

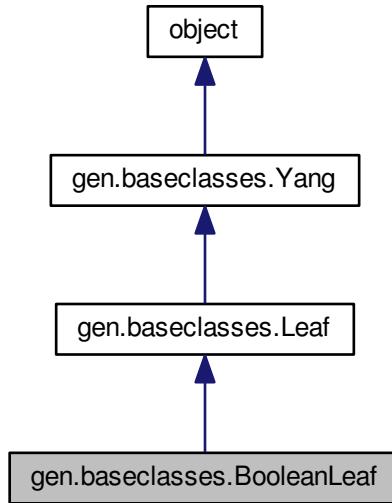
### 6.1 gen.baseclasses.BooleanLeaf Class Reference

Class defining `Leaf` with boolean extensions (e.g., `True` or `False`)

Inheritance diagram for `gen.baseclasses.BooleanLeaf`:



Collaboration diagram for gen.baseclasses.BooleanLeaf:



## Public Member Functions

- def `__init__`
- def `parse` (self, root)
 

*Abstract method to create instance class `BooleanLeaf` from XML string ;param root: ElementTree :return: -.*
- def `get_as_text` (self)
 

*Returns data value as text ;param: - :return: string.*
- def `get_value` (self)
 

*Returns data value ;param: - :return: int.*
- def `set_value` (self, value)
 

*Sets data value as decimal ;param value: int :return: -.*

## Public Attributes

- `data`
- `initialized`

### 6.1.1 Detailed Description

Class defining `Leaf` with boolean extensions (e.g., True or False)

Definition at line 851 of file baseclasses.py.

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 def `gen.baseclasses.BooleanLeaf.__init__` ( `self`, `tag`, `parent=None`, `value=None`, `units=" "`, `mandatory=False` )

Definition at line 852 of file baseclasses.py.

### 6.1.3 Member Function Documentation

#### 6.1.3.1 def gen.baseclasses.BooleanLeaf.get\_as\_text ( self )

Returns data value as text :param: - :return: string.

Definition at line 888 of file baseclasses.py.

#### 6.1.3.2 def gen.baseclasses.BooleanLeaf.get\_value ( self )

Returns data value :param: - :return: int.

Definition at line 899 of file baseclasses.py.

#### 6.1.3.3 def gen.baseclasses.BooleanLeaf.parse ( self, root )

Abstract method to create instance class BooleanLeaf from XML string :param root: ElementTree :return: -.

Definition at line 869 of file baseclasses.py.

#### 6.1.3.4 def gen.baseclasses.BooleanLeaf.set\_value ( self, value )

Sets data value as decimal :param value: int :return: -.

Definition at line 908 of file baseclasses.py.

### 6.1.4 Member Data Documentation

#### 6.1.4.1 gen.baseclasses.BooleanLeaf.data

Definition at line 854 of file baseclasses.py.

#### 6.1.4.2 gen.baseclasses.BooleanLeaf.initialized

Definition at line 880 of file baseclasses.py.

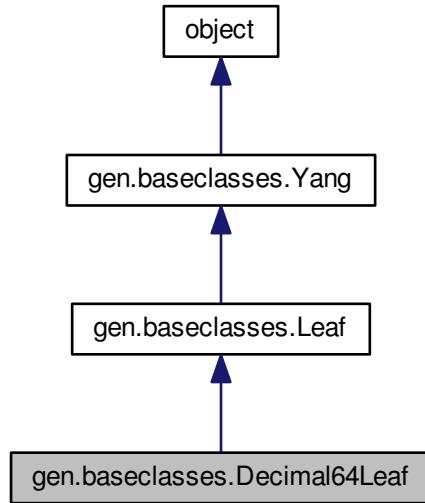
The documentation for this class was generated from the following file:

- [baseclasses.py](#)

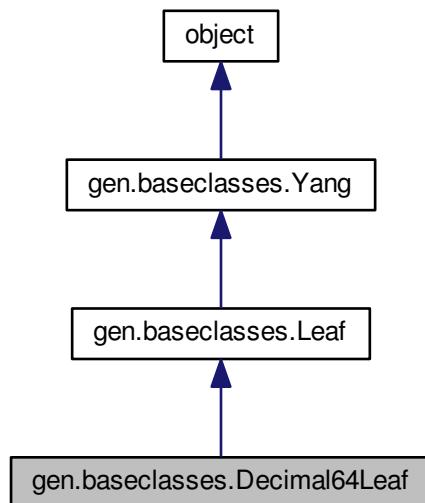
## 6.2 gen.baseclasses.Decimal64Leaf Class Reference

Class defining Leaf with decimal extensions (e.g., dec\_range)

Inheritance diagram for gen.baseclasses.Decimal64Leaf:



Collaboration diagram for gen.baseclasses.Decimal64Leaf:



## Public Member Functions

- def `__init__`
- def `parse` (`self`, `root`)

- Abstract method to instance class `Decimal64Leaf` from XML string :param root: ElementTree :return: -.*
- def `get_as_text` (`self`)
 

*Returns data value as text :param: - :return: string.*
  - def `get_value` (`self`)
 

*Returns data value :param: - :return: decimal.*
  - def `set_value` (`self, value`)
 

*Sets data value as decimal :param value: decimal :return: -.*
  - def `check_range` (`self, value`)
 

*Check if value is inside range limits :param value: decimal :return: boolean.*

## Public Attributes

- `dec_range`
- `fraction_digits`
- `data`
- `initialized`

### 6.2.1 Detailed Description

Class defining `Leaf` with decimal extensions (e.g., `dec_range`)

Definition at line 761 of file `baseclasses.py`.

### 6.2.2 Constructor & Destructor Documentation

- 6.2.2.1 def `gen.baseclasses.Decimal64Leaf.__init__` ( `self, tag, parent=None, value=None, dec_range=[], fraction_digits=1, units=" ", mandatory=False` )

Definition at line 762 of file `baseclasses.py`.

### 6.2.3 Member Function Documentation

- 6.2.3.1 def `gen.baseclasses.Decimal64Leaf.check_range` ( `self, value` )

Check if value is inside range limits :param value: decimal :return: boolean.

Definition at line 837 of file `baseclasses.py`.

- 6.2.3.2 def `gen.baseclasses.Decimal64Leaf.get_as_text` ( `self` )

Returns data value as text :param: - :return: string.

Definition at line 800 of file `baseclasses.py`.

- 6.2.3.3 def `gen.baseclasses.Decimal64Leaf.get_value` ( `self` )

Returns data value :param: - :return: decimal.

Definition at line 811 of file `baseclasses.py`.

## Class Documentation

6.2.3.4 def gen.baseclasses.Decimal64Leaf.parse ( self, root )

Abstract method to instance class Decimal64Leaf from XML string :param root: ElementTree :return: -.

Definition at line 781 of file baseclasses.py.

6.2.3.5 def gen.baseclasses.Decimal64Leaf.set\_value ( self, value )

Sets data value as decimal :param value: decimal :return: -.

Definition at line 820 of file baseclasses.py.

## 6.2.4 Member Data Documentation

6.2.4.1 gen.baseclasses.Decimal64Leaf.data

Definition at line 766 of file baseclasses.py.

6.2.4.2 gen.baseclasses.Decimal64Leaf.dec\_range

Definition at line 764 of file baseclasses.py.

6.2.4.3 gen.baseclasses.Decimal64Leaf.fraction\_digits

Definition at line 765 of file baseclasses.py.

6.2.4.4 gen.baseclasses.Decimal64Leaf.initialized

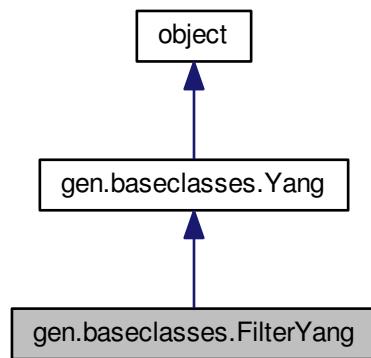
Definition at line 792 of file baseclasses.py.

The documentation for this class was generated from the following file:

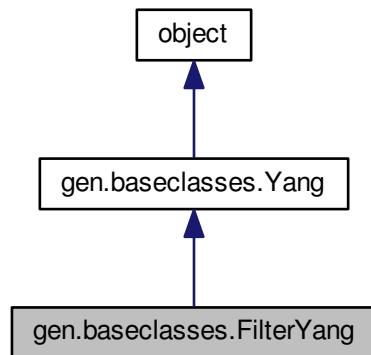
- [baseclasses.py](#)

## 6.3 gen.baseclasses.FilterYang Class Reference

Inheritance diagram for gen.baseclasses.FilterYang:



Collaboration diagram for gen.baseclasses.FilterYang:



### Public Member Functions

- def `__init__` (self, filter)
- def `run` (self, yang)

### Public Attributes

- `filter_xml`

### 6.3.1 Detailed Description

Definition at line 1407 of file baseclasses.py.

### 6.3.2 Constructor & Destructor Documentation

#### 6.3.2.1 def gen.baseclasses.FilterYang.\_\_init\_\_( self, filter )

Definition at line 1408 of file baseclasses.py.

### 6.3.3 Member Function Documentation

#### 6.3.3.1 def gen.baseclasses.FilterYang.run( self, yang )

Definition at line 1412 of file baseclasses.py.

### 6.3.4 Member Data Documentation

#### 6.3.4.1 gen.baseclasses.FilterYang.filter\_xml

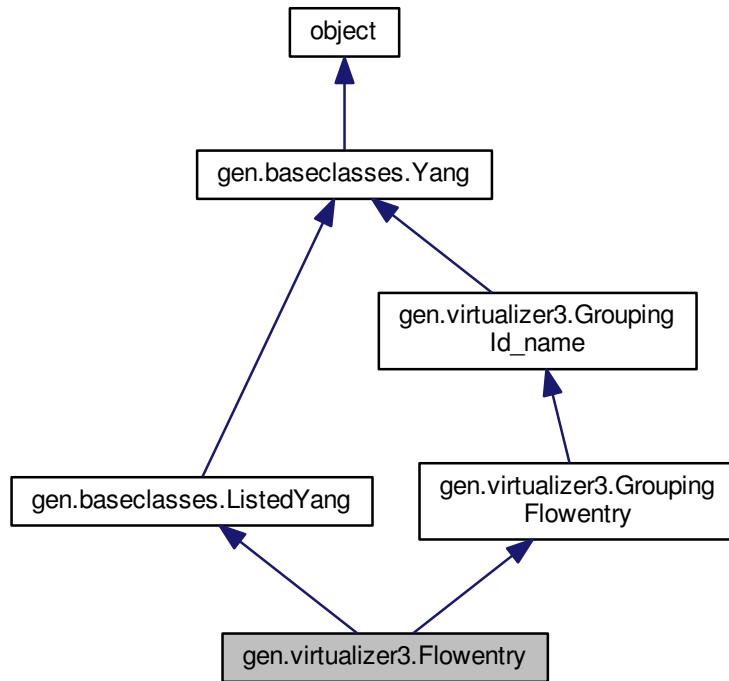
Definition at line 1410 of file baseclasses.py.

The documentation for this class was generated from the following file:

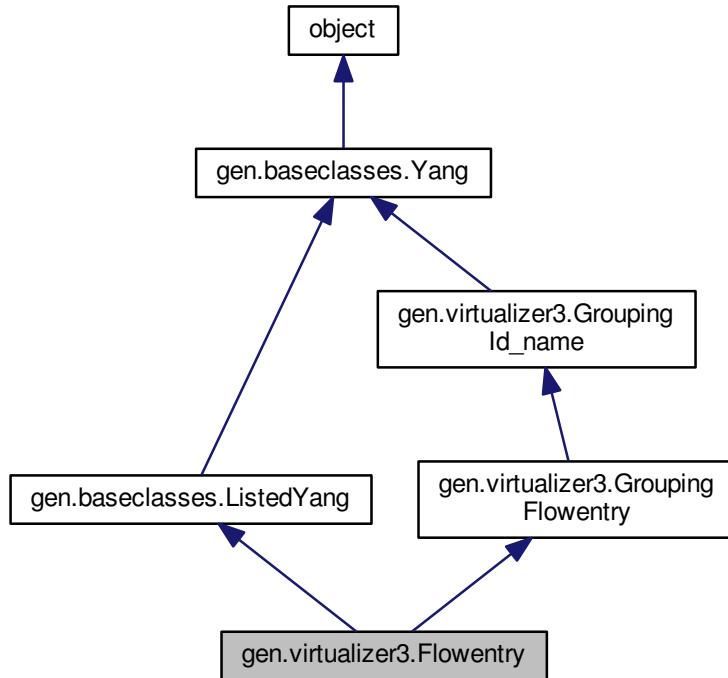
- [baseclasses.py](#)

## 6.4 gen.virtualizer3.Flowentry Class Reference

Inheritance diagram for gen.virtualizer3.Flowentry:



Collaboration diagram for gen.virtualizer3.Flowentry:



## Public Member Functions

- def `__init__`

## Additional Inherited Members

### 6.4.1 Detailed Description

Definition at line 240 of file virtualizer3.py.

### 6.4.2 Constructor & Destructor Documentation

6.4.2.1 def `gen.virtualizer3.Flowentry.__init__( self, tag = "flowentry", parent = None, id = None, name = None, priority = None, port = None, match = None, action = None, out = None, resources = None )`

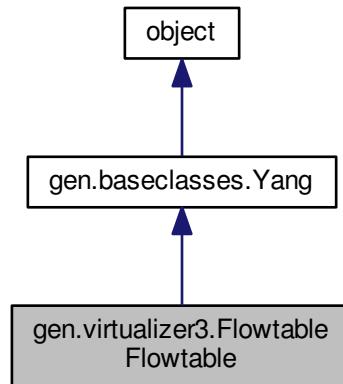
Definition at line 241 of file virtualizer3.py.

The documentation for this class was generated from the following file:

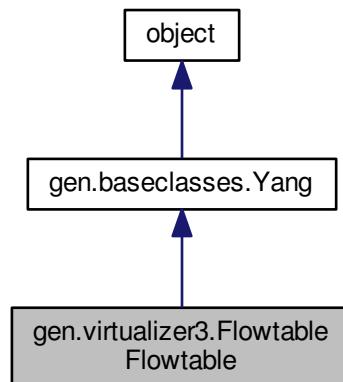
- `virtualizer3.py`

## 6.5 gen.virtualizer3.FlowtableFlowtable Class Reference

Inheritance diagram for gen.virtualizer3.FlowtableFlowtable:



Collaboration diagram for gen.virtualizer3.FlowtableFlowtable:



### Public Member Functions

- def `__init__`
- def `add` (self, item)
- def `remove` (self, item)
- def `__getitem__` (self, key)
- def `__iter__` (self)

## Public Attributes

- [flowentry](#)

### 6.5.1 Detailed Description

Definition at line 287 of file virtualizer3.py.

### 6.5.2 Constructor & Destructor Documentation

6.5.2.1 `def gen.virtualizer3.FlowtableFlowtable.__init__ ( self, tag = "flowtable", parent = None )`

Definition at line 288 of file virtualizer3.py.

### 6.5.3 Member Function Documentation

6.5.3.1 `def gen.virtualizer3.FlowtableFlowtable.__getitem__ ( self, key )`

Definition at line 301 of file virtualizer3.py.

6.5.3.2 `def gen.virtualizer3.FlowtableFlowtable.__iter__ ( self )`

Definition at line 304 of file virtualizer3.py.

6.5.3.3 `def gen.virtualizer3.FlowtableFlowtable.add ( self, item )`

Definition at line 295 of file virtualizer3.py.

6.5.3.4 `def gen.virtualizer3.FlowtableFlowtable.remove ( self, item )`

Definition at line 298 of file virtualizer3.py.

### 6.5.4 Member Data Documentation

6.5.4.1 `gen.virtualizer3.FlowtableFlowtable.flowentry`

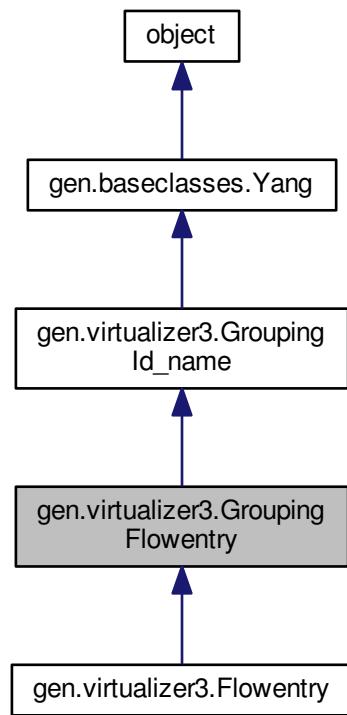
Definition at line 292 of file virtualizer3.py.

The documentation for this class was generated from the following file:

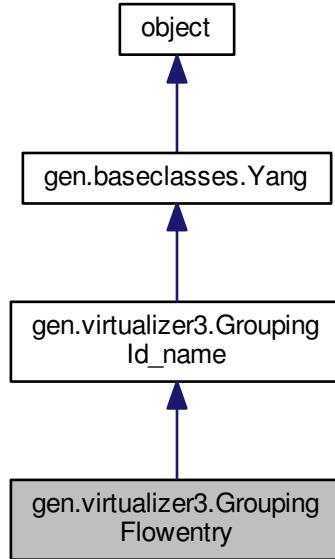
- [virtualizer3.py](#)

## 6.6 gen.virtualizer3.GroupingFlowentry Class Reference

Inheritance diagram for gen.virtualizer3.GroupingFlowentry:



Collaboration diagram for gen.virtualizer3.GroupingFlowentry:



## Public Member Functions

- def `__init__`

## Public Attributes

- `priority`
- `port`
- `match`
- `action`
- `out`
- `resources`

### 6.6.1 Detailed Description

Definition at line 79 of file virtualizer3.py.

### 6.6.2 Constructor & Destructor Documentation

**6.6.2.1 def gen.virtualizer3.GroupingFlowentry.\_\_init\_\_( self, tag, parent=None, id=None, name=None, priority=None, port=None, match=None, action=None, out=None, resources=None )**

Definition at line 80 of file virtualizer3.py.

### 6.6.3 Member Data Documentation

#### 6.6.3.1 gen.virtualizer3.GroupingFlowentry.action

Definition at line 93 of file virtualizer3.py.

#### 6.6.3.2 gen.virtualizer3.GroupingFlowentry.match

Definition at line 90 of file virtualizer3.py.

#### 6.6.3.3 gen.virtualizer3.GroupingFlowentry.out

Definition at line 96 of file virtualizer3.py.

#### 6.6.3.4 gen.virtualizer3.GroupingFlowentry.port

Definition at line 87 of file virtualizer3.py.

#### 6.6.3.5 gen.virtualizer3.GroupingFlowentry.priority

Definition at line 84 of file virtualizer3.py.

#### 6.6.3.6 gen.virtualizer3.GroupingFlowentry.resources

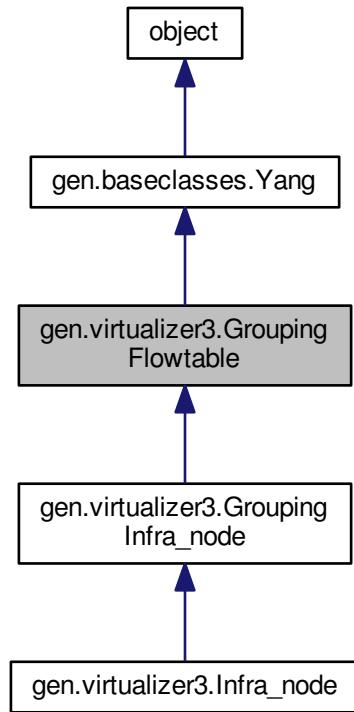
Definition at line 99 of file virtualizer3.py.

The documentation for this class was generated from the following file:

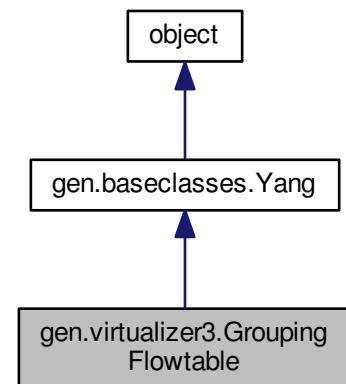
- [virtualizer3.py](#)

## 6.7 gen.virtualizer3.GroupingFlowtable Class Reference

Inheritance diagram for gen.virtualizer3.GroupingFlowtable:



Collaboration diagram for gen.virtualizer3.GroupingFlowtable:



## Public Member Functions

- def `__init__`

## Public Attributes

- `flowtable`

### 6.7.1 Detailed Description

Definition at line 108 of file `virtualizer3.py`.

### 6.7.2 Constructor & Destructor Documentation

#### 6.7.2.1 def gen.virtualizer3.GroupingFlowtable.`__init__`( `self`, `tag`, `parent`=`None`, `flowtable`=`None` )

Definition at line 109 of file `virtualizer3.py`.

### 6.7.3 Member Data Documentation

#### 6.7.3.1 gen.virtualizer3.GroupingFlowtable.`flowtable`

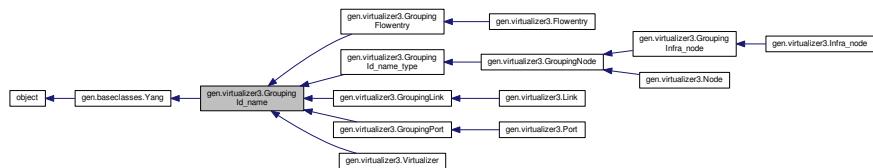
Definition at line 113 of file `virtualizer3.py`.

The documentation for this class was generated from the following file:

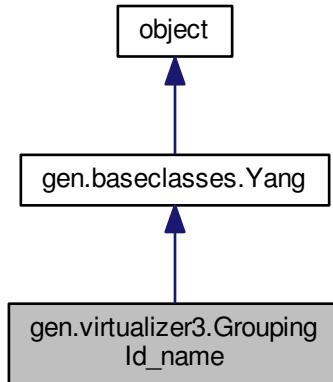
- `virtualizer3.py`

## 6.8 gen.virtualizer3.GroupingId\_name Class Reference

Inheritance diagram for `gen.virtualizer3.GroupingId_name`:



Collaboration diagram for gen.virtualizer3.GroupingId\_name:



## Public Member Functions

- def `__init__`

## Public Attributes

- `id`
- `name`

### 6.8.1 Detailed Description

Definition at line 27 of file virtualizer3.py.

### 6.8.2 Constructor & Destructor Documentation

#### 6.8.2.1 def gen.virtualizer3.GroupingId\_name.\_\_init\_\_( self, tag, parent=None, id=None, name=None )

Definition at line 28 of file virtualizer3.py.

### 6.8.3 Member Data Documentation

#### 6.8.3.1 gen.virtualizer3.GroupingId\_name.id

Definition at line 32 of file virtualizer3.py.

#### 6.8.3.2 gen.virtualizer3.GroupingId\_name.name

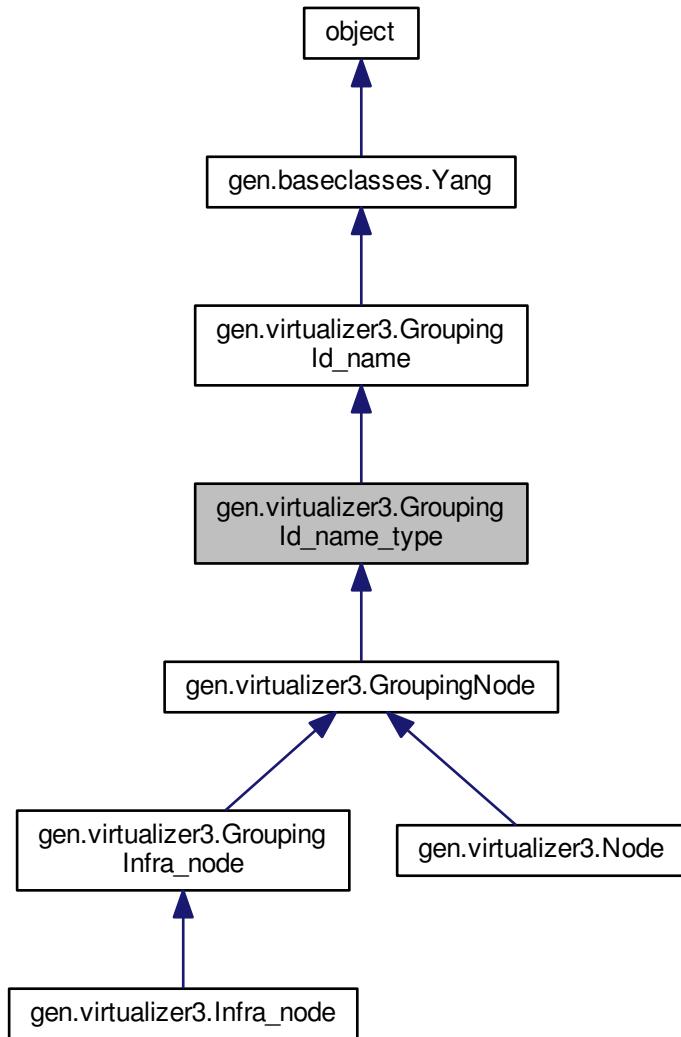
Definition at line 35 of file virtualizer3.py.

The documentation for this class was generated from the following file:

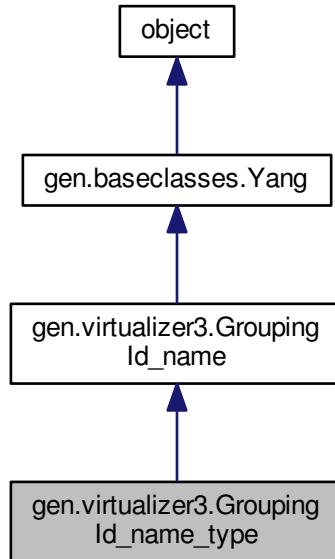
- `virtualizer3.py`

## 6.9 gen.virtualizer3.GroupingId\_name\_type Class Reference

Inheritance diagram for gen.virtualizer3.GroupingId\_name\_type:



Collaboration diagram for gen.virtualizer3.GroupingId\_name\_type:



## Public Member Functions

- `def __init__`

## Public Attributes

- `type`

### 6.9.1 Detailed Description

Definition at line 40 of file virtualizer3.py.

### 6.9.2 Constructor & Destructor Documentation

**6.9.2.1** `def gen.virtualizer3.GroupingId_name_type.__init__( self, tag, parent =None, id =None, name =None, type =None )`

Definition at line 41 of file virtualizer3.py.

### 6.9.3 Member Data Documentation

**6.9.3.1** `gen.virtualizer3.GroupingId_name_type.type`

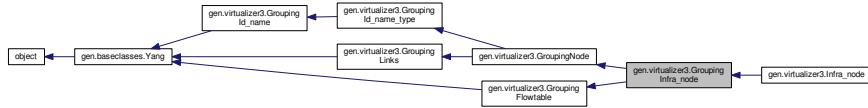
Definition at line 45 of file virtualizer3.py.

The documentation for this class was generated from the following file:

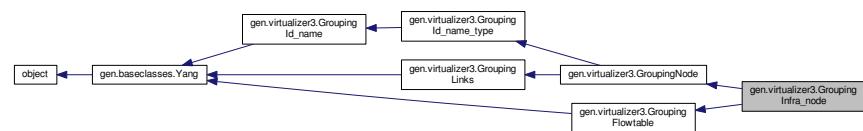
- [virtualizer3.py](#)

## 6.10 gen.virtualizer3.GroupingInfra\_node Class Reference

Inheritance diagram for gen.virtualizer3.GroupingInfra\_node:



Collaboration diagram for gen.virtualizer3.GroupingInfra\_node:



### Public Member Functions

- [def \\_\\_init\\_\\_](#)

### Public Attributes

- [NF\\_instances](#)
- [capabilities](#)

#### 6.10.1 Detailed Description

Definition at line 218 of file virtualizer3.py.

#### 6.10.2 Constructor & Destructor Documentation

**6.10.2.1 def gen.virtualizer3.GroupingInfra\_node.\_\_init\_\_( self, tag, parent =None, id =None, name =None, type =None, ports =None, links =None, resources =None, NF\_instances =None, capabilities =None, flowtable =None )**

Definition at line 219 of file virtualizer3.py.

#### 6.10.3 Member Data Documentation

##### 6.10.3.1 gen.virtualizer3.GroupingInfra\_node.capabilities

Definition at line 231 of file virtualizer3.py.

## 6.10.3.2 gen.virtualizer3.GroupingInfra\_node.NF\_instances

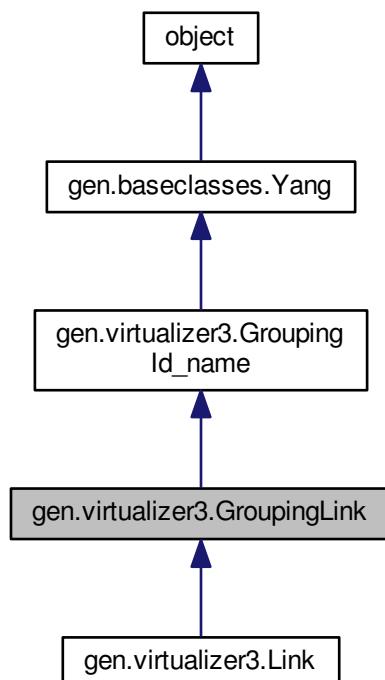
Definition at line 224 of file virtualizer3.py.

The documentation for this class was generated from the following file:

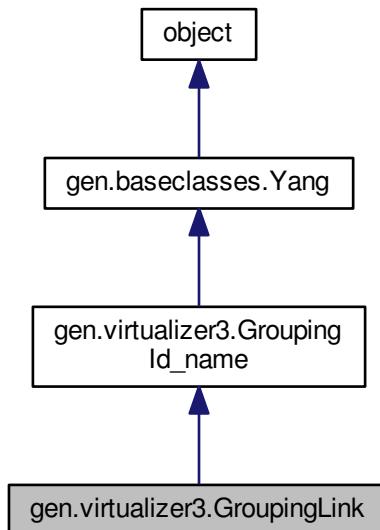
- [virtualizer3.py](#)

## 6.11 gen.virtualizer3.GroupingLink Class Reference

Inheritance diagram for gen.virtualizer3.GroupingLink:



Collaboration diagram for gen.virtualizer3.GroupingLink:



## Public Member Functions

- `def __init__`

## Public Attributes

- `src`
- `dst`
- `resources`

### 6.11.1 Detailed Description

Definition at line 122 of file virtualizer3.py.

### 6.11.2 Constructor & Destructor Documentation

6.11.2.1 `def gen.virtualizer3.GroupingLink.__init__( self, tag, parent=None, id=None, name=None, src=None, dst=None, resources=None )`

Definition at line 123 of file virtualizer3.py.

### 6.11.3 Member Data Documentation

6.11.3.1 `gen.virtualizer3.GroupingLink.dst`

Definition at line 130 of file virtualizer3.py.

## 6.11.3.2 gen.virtualizer3.GroupingLink.resources

Definition at line 133 of file virtualizer3.py.

## 6.11.3.3 gen.virtualizer3.GroupingLink.src

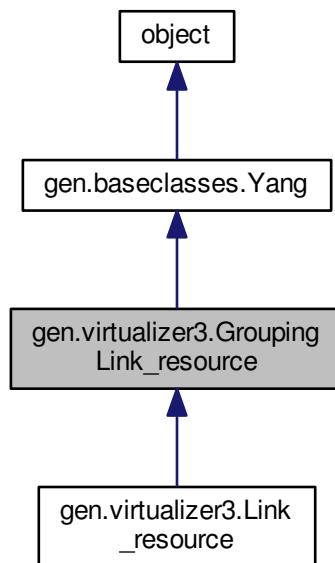
Definition at line 127 of file virtualizer3.py.

The documentation for this class was generated from the following file:

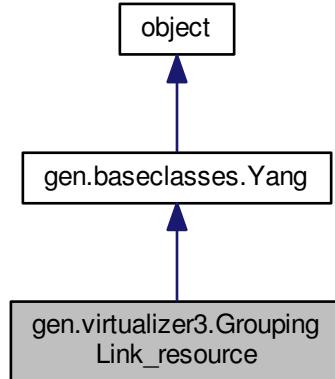
- [virtualizer3.py](#)

## 6.12 gen.virtualizer3.GroupingLink\_resource Class Reference

Inheritance diagram for gen.virtualizer3.GroupingLink\_resource:



Collaboration diagram for gen.virtualizer3.GroupingLink\_resource:



## Public Member Functions

- `def __init__`

## Public Attributes

- `delay`
- `bandwidth`

### 6.12.1 Detailed Description

Definition at line 66 of file virtualizer3.py.

### 6.12.2 Constructor & Destructor Documentation

**6.12.2.1 def gen.virtualizer3.GroupingLink\_resource.\_\_init\_\_ ( self, tag, parent = None, delay = None, bandwidth = None )**

Definition at line 67 of file virtualizer3.py.

### 6.12.3 Member Data Documentation

**6.12.3.1 gen.virtualizer3.GroupingLink\_resource.bandwidth**

Definition at line 74 of file virtualizer3.py.

**6.12.3.2 gen.virtualizer3.GroupingLink\_resource.delay**

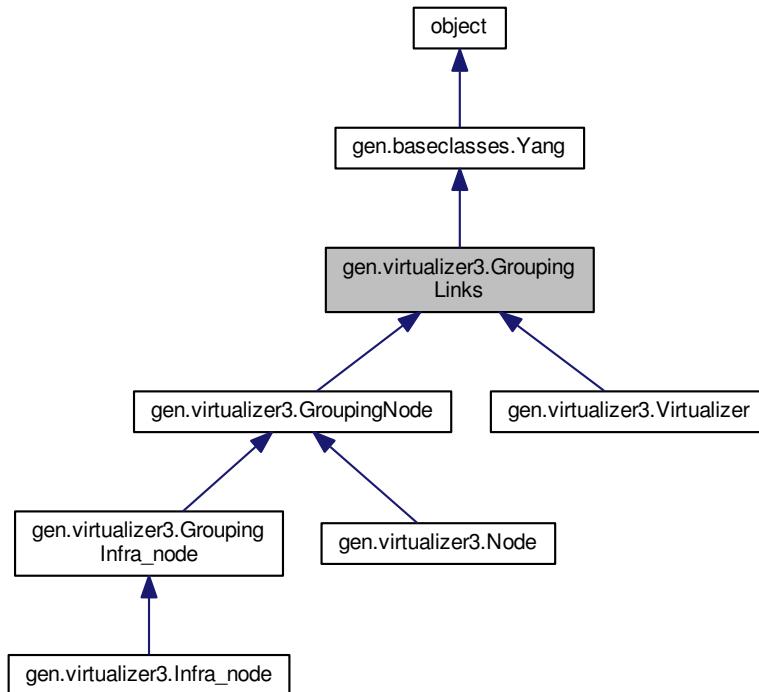
Definition at line 71 of file virtualizer3.py.

The documentation for this class was generated from the following file:

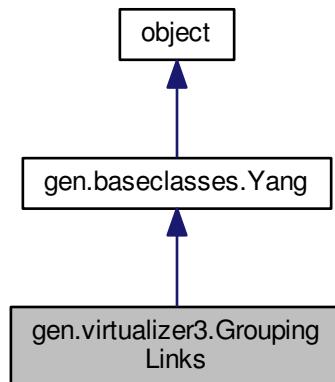
- [virtualizer3.py](#)

## 6.13 gen.virtualizer3.GroupingLinks Class Reference

Inheritance diagram for gen.virtualizer3.GroupingLinks:



Collaboration diagram for gen.virtualizer3.GroupingLinks:



## Public Member Functions

- `def __init__`

## Public Attributes

- `links`

### 6.13.1 Detailed Description

Definition at line 142 of file `virtualizer3.py`.

### 6.13.2 Constructor & Destructor Documentation

#### 6.13.2.1 `def gen.virtualizer3.GroupingLinks.__init__( self, tag, parent =None, links =None )`

Definition at line 143 of file `virtualizer3.py`.

### 6.13.3 Member Data Documentation

#### 6.13.3.1 `gen.virtualizer3.GroupingLinks.links`

Definition at line 147 of file `virtualizer3.py`.

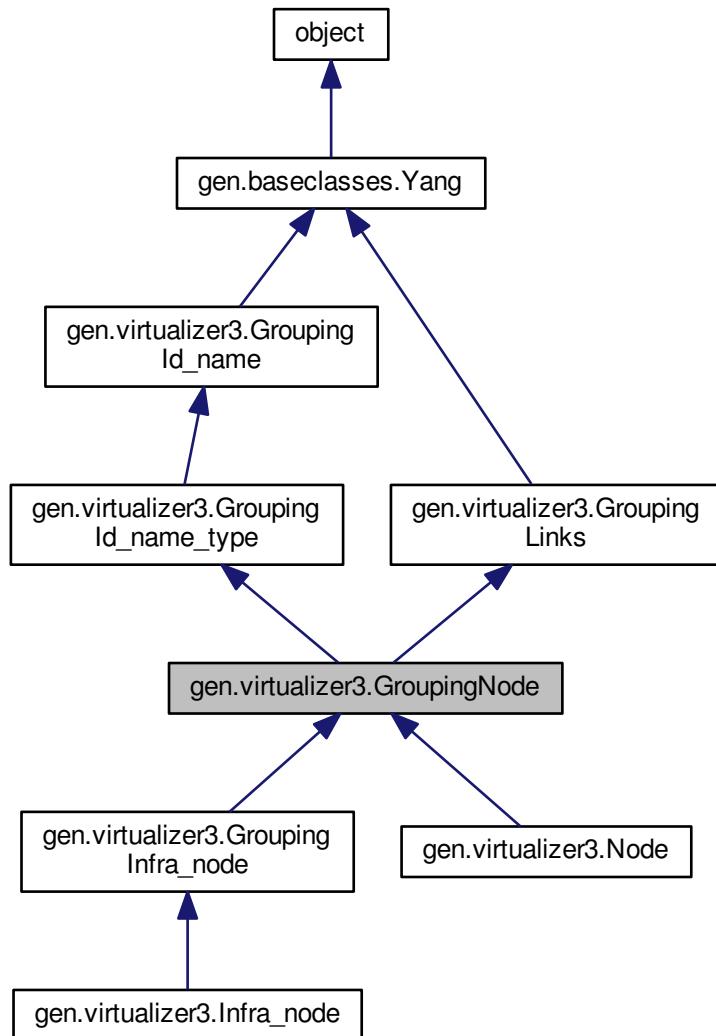
The documentation for this class was generated from the following file:

- `virtualizer3.py`

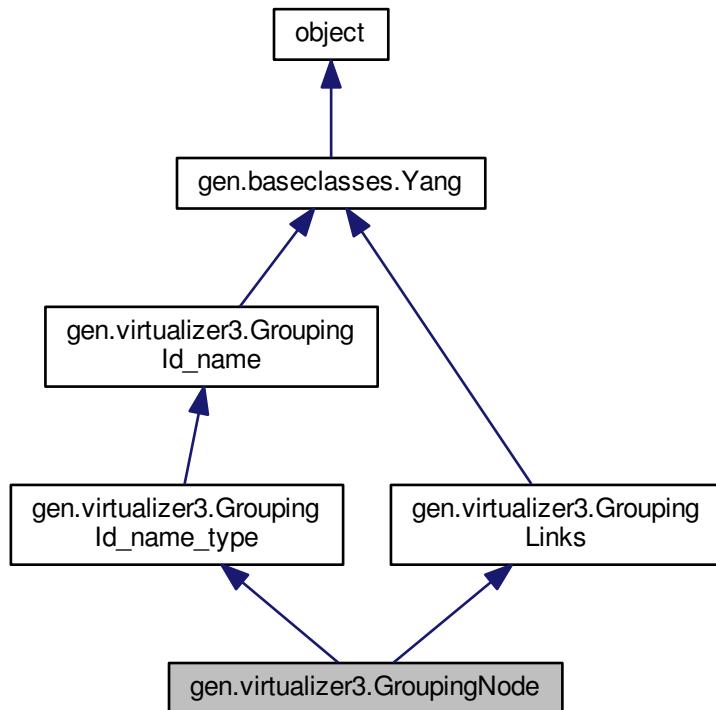
## 6.14 `gen.virtualizer3.GroupingNode` Class Reference

Any node: infrastructure or NFs.

Inheritance diagram for gen.virtualizer3.GroupingNode:



Collaboration diagram for gen.virtualizer3.GroupingNode:



## Public Member Functions

- `def __init__`

## Public Attributes

- `ports`
- `resources`

### 6.14.1 Detailed Description

Any node: infrastructure or NFs.

Definition at line 174 of file `virtualizer3.py`.

### 6.14.2 Constructor & Destructor Documentation

6.14.2.1 `def gen.virtualizer3.GroupingNode.__init__( self, tag, parent=None, id=None, name=None, type=None, ports=None, links=None, resources=None )`

Definition at line 175 of file `virtualizer3.py`.

### 6.14.3 Member Data Documentation

#### 6.14.3.1 gen.virtualizer3.GroupingNode.ports

Definition at line 180 of file virtualizer3.py.

#### 6.14.3.2 gen.virtualizer3.GroupingNode.resources

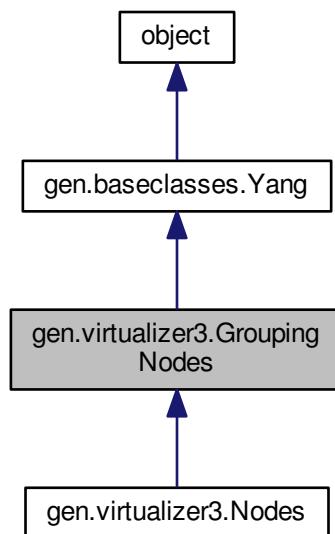
Definition at line 187 of file virtualizer3.py.

The documentation for this class was generated from the following file:

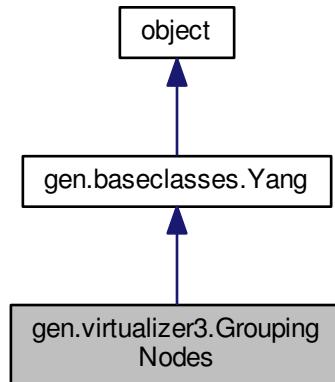
- [virtualizer3.py](#)

## 6.15 gen.virtualizer3.GroupingNodes Class Reference

Inheritance diagram for gen.virtualizer3.GroupingNodes:



Collaboration diagram for gen.virtualizer3.GroupingNodes:



## Public Member Functions

- def `__init__`
- def `add (self, item)`
- def `remove (self, item)`
- def `__getitem__ (self, key)`
- def `__iter__ (self)`

## Public Attributes

- `node`

### 6.15.1 Detailed Description

Definition at line 196 of file virtualizer3.py.

### 6.15.2 Constructor & Destructor Documentation

#### 6.15.2.1 def gen.virtualizer3.GroupingNodes.\_\_init\_\_ ( `self, tag, parent=None` )

Definition at line 197 of file virtualizer3.py.

### 6.15.3 Member Function Documentation

#### 6.15.3.1 def gen.virtualizer3.GroupingNodes.\_\_getitem\_\_ ( `self, key` )

Definition at line 210 of file virtualizer3.py.

#### 6.15.3.2 def gen.virtualizer3.GroupingNodes.\_\_iter\_\_ ( `self` )

Definition at line 213 of file virtualizer3.py.

6.15.3.3 def gen.virtualizer3.GroupingNodes.add ( self, item )

Definition at line 204 of file virtualizer3.py.

6.15.3.4 def gen.virtualizer3.GroupingNodes.remove ( self, item )

Definition at line 207 of file virtualizer3.py.

#### 6.15.4 Member Data Documentation

6.15.4.1 gen.virtualizer3.GroupingNodes.node

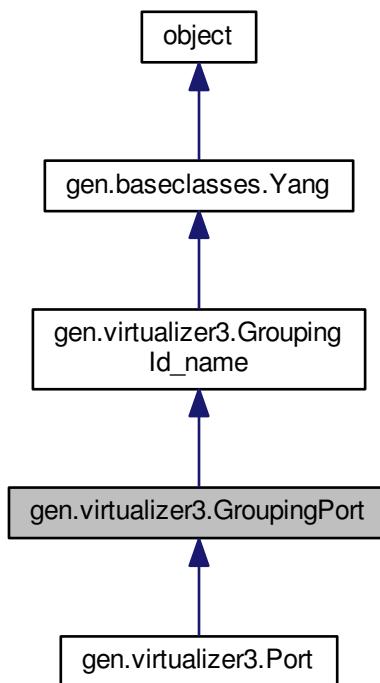
Definition at line 201 of file virtualizer3.py.

The documentation for this class was generated from the following file:

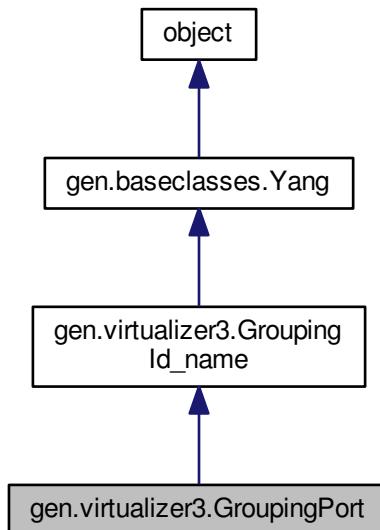
- [virtualizer3.py](#)

### 6.16 gen.virtualizer3.GroupingPort Class Reference

Inheritance diagram for gen.virtualizer3.GroupingPort:



Collaboration diagram for gen.virtualizer3.GroupingPort:



## Public Member Functions

- `def __init__`

## Public Attributes

- `port_type`
- `capability`
- `sap`

### 6.16.1 Detailed Description

Definition at line 50 of file virtualizer3.py.

### 6.16.2 Constructor & Destructor Documentation

6.16.2.1 `def gen.virtualizer3.GroupingPort.__init__ ( self, tag, parent = None, id = None, name = None, port_type = None, capability = None, sap = None )`

Definition at line 51 of file virtualizer3.py.

### 6.16.3 Member Data Documentation

#### 6.16.3.1 `gen.virtualizer3.GroupingPort.capability`

Definition at line 58 of file virtualizer3.py.

## 6.16.3.2 gen.virtualizer3.GroupingPort.port\_type

Definition at line 55 of file virtualizer3.py.

## 6.16.3.3 gen.virtualizer3.GroupingPort.sap

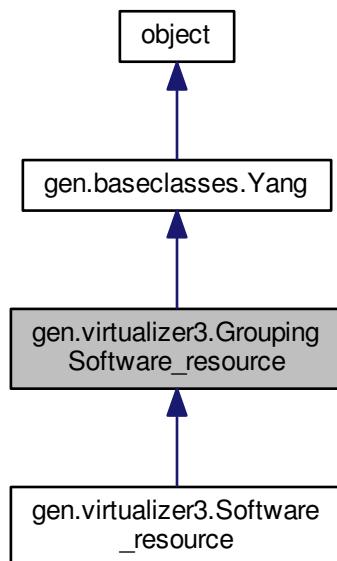
Definition at line 61 of file virtualizer3.py.

The documentation for this class was generated from the following file:

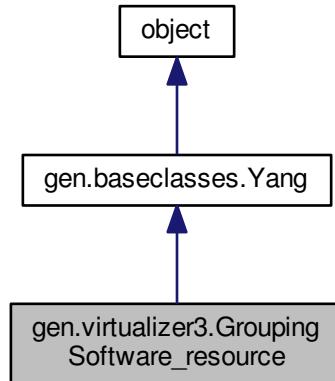
- [virtualizer3.py](#)

## 6.17 gen.virtualizer3.GroupingSoftware\_resource Class Reference

Inheritance diagram for gen.virtualizer3.GroupingSoftware\_resource:



Collaboration diagram for gen.virtualizer3.GroupingSoftware\_resource:



## Public Member Functions

- def [\\_\\_init\\_\\_](#)

## Public Attributes

- `cpu`
- `mem`
- `storage`

### 6.17.1 Detailed Description

Definition at line 156 of file virtualizer3.py.

### 6.17.2 Constructor & Destructor Documentation

6.17.2.1 def `gen.virtualizer3.GroupingSoftware_resource.__init__( self, tag, parent = None, cpu = None, mem = None, storage = None )`

Definition at line 157 of file virtualizer3.py.

### 6.17.3 Member Data Documentation

6.17.3.1 `gen.virtualizer3.GroupingSoftware_resource.cpu`

Definition at line 161 of file virtualizer3.py.

6.17.3.2 `gen.virtualizer3.GroupingSoftware_resource.mem`

Definition at line 164 of file virtualizer3.py.

### 6.17.3.3 gen.virtualizer3.GroupingSoftware\_resource.storage

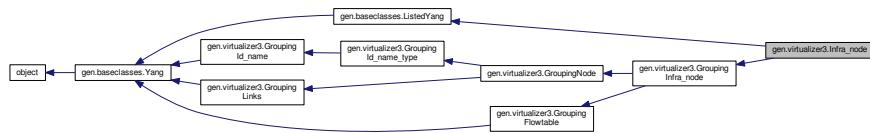
Definition at line 167 of file virtualizer3.py.

The documentation for this class was generated from the following file:

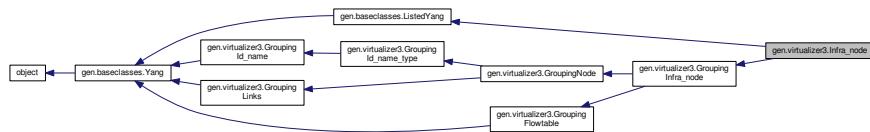
- [virtualizer3.py](#)

## 6.18 gen.virtualizer3.Infra\_node Class Reference

Inheritance diagram for gen.virtualizer3.Infra\_node:



Collaboration diagram for gen.virtualizer3.Infra\_node:



## Public Member Functions

- [def \\_\\_init\\_\\_](#)

## Additional Inherited Members

### 6.18.1 Detailed Description

Definition at line 272 of file virtualizer3.py.

### 6.18.2 Constructor & Destructor Documentation

**6.18.2.1 def gen.virtualizer3.Infra\_node.\_\_init\_\_( self, tag = "node", parent = None, id = None, name = None, type = None, ports = None, links = None, resources = None, NF\_instances = None, capabilities = None, flowtable = None )**

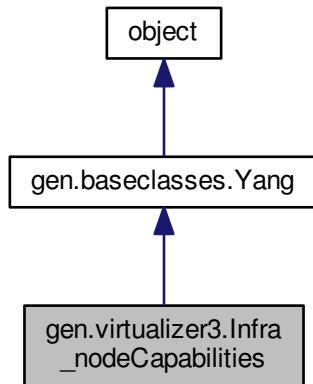
Definition at line 273 of file virtualizer3.py.

The documentation for this class was generated from the following file:

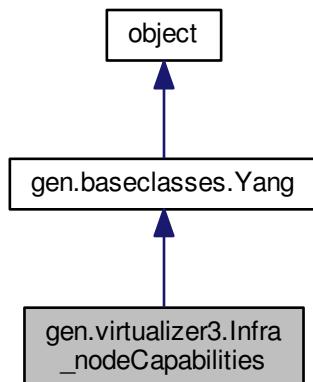
- [virtualizer3.py](#)

## 6.19 gen.virtualizer3.Infra\_nodeCapabilities Class Reference

Inheritance diagram for gen.virtualizer3.Infra\_nodeCapabilities:



Collaboration diagram for gen.virtualizer3.Infra\_nodeCapabilities:



### Public Member Functions

- `def __init__`

### Public Attributes

- `supported_NFs`

### 6.19.1 Detailed Description

Definition at line 367 of file virtualizer3.py.

### 6.19.2 Constructor & Destructor Documentation

```
6.19.2.1 def gen.virtualizer3.Infra_nodeCapabilities.__init__ ( self, tag = "capabilities", parent = None, supported_NFs = None )
```

Definition at line 368 of file virtualizer3.py.

### 6.19.3 Member Data Documentation

```
6.19.3.1 gen.virtualizer3.Infra_nodeCapabilities.supported_NFs
```

Definition at line 372 of file virtualizer3.py.

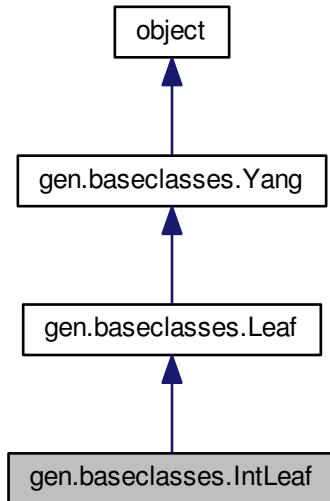
The documentation for this class was generated from the following file:

- [virtualizer3.py](#)

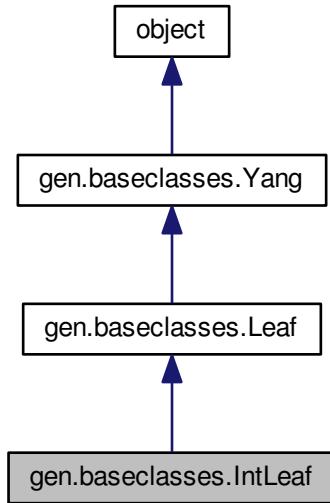
## 6.20 gen.baseclasses.IntLeaf Class Reference

Class defining `Leaf` with integer extensions (e.g., range)

Inheritance diagram for `gen.baseclasses.IntLeaf`:



Collaboration diagram for gen.baseclasses.IntLeaf:



## Public Member Functions

- def `__init__`
- def `parse (self, root)`

*Creates instance `IntLeaf` setting its value from XML string :param root: ElementTree :return: -.*
- def `get_as_text (self)`

*Returns data value as text :param: - :return: string.*
- def `get_value (self)`

*Returns data value :param: - :return: int.*
- def `set_value (self, value)`

*Sets data value as int :param value: int :return: -.*
- def `check_range (self, value)`

*Check if value is inside range limits :param value: int :return: boolean.*

## Public Attributes

- `int_range`
- `data`
- `initialized`

### 6.20.1 Detailed Description

Class defining `Leaf` with integer extensions (e.g., range)

Definition at line 657 of file `baseclasses.py`.

## Class Documentation

### 6.20.2 Constructor & Destructor Documentation

6.20.2.1 `def gen.baseclasses.IntLeaf.__init__( self, tag, parent=None, value=None, int_range=[], units="", mandatory=False )`

Definition at line 658 of file baseclasses.py.

### 6.20.3 Member Function Documentation

6.20.3.1 `def gen.baseclasses.IntLeaf.check_range( self, value )`

Check if value is inside range limits :param value: int :return: boolean.

Definition at line 746 of file baseclasses.py.

6.20.3.2 `def gen.baseclasses.IntLeaf.get_as_text( self )`

Returns data value as text :param: - :return: string.

Definition at line 709 of file baseclasses.py.

6.20.3.3 `def gen.baseclasses.IntLeaf.get_value( self )`

Returns data value :param: - :return: int.

Definition at line 720 of file baseclasses.py.

6.20.3.4 `def gen.baseclasses.IntLeaf.parse( self, root )`

Creates instance `IntLeaf` setting its value from XML string :param root: ElementTree :return: -.

Definition at line 676 of file baseclasses.py.

6.20.3.5 `def gen.baseclasses.IntLeaf.set_value( self, value )`

Sets data value as int :param value: int :return: -.

Definition at line 729 of file baseclasses.py.

### 6.20.4 Member Data Documentation

6.20.4.1 `gen.baseclasses.IntLeaf.data`

Definition at line 661 of file baseclasses.py.

6.20.4.2 `gen.baseclasses.IntLeaf.initialized`

Definition at line 701 of file baseclasses.py.

6.20.4.3 `gen.baseclasses.IntLeaf.int_range`

Definition at line 660 of file baseclasses.py.

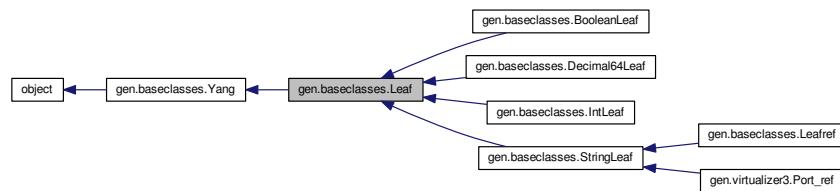
The documentation for this class was generated from the following file:

- [baseclasses.py](#)

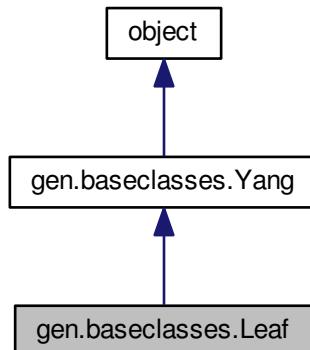
## 6.21 gen.baseclasses.Leaf Class Reference

Class defining [Leaf](#) basis with attributes and methods.

Inheritance diagram for gen.baseclasses.Leaf:



Collaboration diagram for gen.baseclasses.Leaf:



### Public Member Functions

- def `__init__`
- def `get_as_text` (self)
 

*Abstract method to get data as text.*
- def `get_value` (self)
 

*Abstract method to get data value.*
- def `set_value` (self, value)
 

*Abstract method to set data value.*
- def `get_units` (self)
 

*Return self.units :return: string.*
- def `set_units` (self, units)
 

*Set self.units :param units: :return: -.*
- def `get_mandatory` (self)

*Return self.mandatory :return: string.*

- def `set_mandatory` (self, mandatory)

*Set self.mandatory :param mandatory: :return: -.*

- def `is_initialized` (self)

*Overrides Yang method to check if data contains value ;param: - :return: boolean.*

- def `clear_data` (self)

*Erases data defining it as None ;param: - :return: -.*

- def `delete` (self)

*Erases data defining it as None ;param: - :return: -.*

- def `reduce` (self, reference)

*Overrides Yang method to reduce other, return True if its data if different from self.data or \_operation attributes mismatch ;param reference: instance of Yang :return: boolean.*

- def `__eq__` (self, other)

*Check if other leaf has the same attributes and values, returns True if yes ;param other: instance :return: boolean.*

## Public Attributes

- `data`
- `mandatory`
- `units`

### 6.21.1 Detailed Description

Class defining `Leaf` basis with attributes and methods.

Definition at line 453 of file `baseclasses.py`.

### 6.21.2 Constructor & Destructor Documentation

#### 6.21.2.1 def gen.baseclasses.Leaf.\_\_init\_\_ ( self, tag, parent=None )

Definition at line 454 of file `baseclasses.py`.

### 6.21.3 Member Function Documentation

#### 6.21.3.1 def gen.baseclasses.Leaf.\_\_eq\_\_ ( self, other )

Check if other leaf has the same attributes and values, returns True if yes ;param other: instance :return: boolean.

Definition at line 583 of file `baseclasses.py`.

#### 6.21.3.2 def gen.baseclasses.Leaf.clear\_data ( self )

Erases data defining it as None ;param: - :return: -.

Definition at line 550 of file `baseclasses.py`.

#### 6.21.3.3 def gen.baseclasses.Leaf.delete ( self )

Erases data defining it as None ;param: - :return: -.

Definition at line 559 of file `baseclasses.py`.

**6.21.3.4 def gen.baseclasses.Leaf.get\_as\_text( self )**

Abstract method to get data as text.

Definition at line 467 of file baseclasses.py.

**6.21.3.5 def gen.baseclasses.Leaf.get\_mandatory( self )**

Return self.mandatory :return: string.

Definition at line 506 of file baseclasses.py.

**6.21.3.6 def gen.baseclasses.Leaf.get\_units( self )**

Return self.units :return: string.

Definition at line 489 of file baseclasses.py.

**6.21.3.7 def gen.baseclasses.Leaf.get\_value( self )**

Abstract method to get data value.

Definition at line 474 of file baseclasses.py.

**6.21.3.8 def gen.baseclasses.Leaf.is\_initialized( self )**

Overides [Yang](#) method to check if data contains value :param: - :return: boolean.

Definition at line 524 of file baseclasses.py.

**6.21.3.9 def gen.baseclasses.Leaf.reduce( self, reference )**

Overides [Yang](#) method to reduce other, return True if its data if different from self.data or \_operation attributes mismatch :param reference: instance of [Yang](#) :return: boolean.

Definition at line 568 of file baseclasses.py.

**6.21.3.10 def gen.baseclasses.Leaf.set\_mandatory( self, mandatory )**

Set self.mandatory :param mandatory: :return: -.

Definition at line 515 of file baseclasses.py.

**6.21.3.11 def gen.baseclasses.Leaf.set\_units( self, units )**

Set self.units :param units: :return: -.

Definition at line 498 of file baseclasses.py.

**6.21.3.12 def gen.baseclasses.Leaf.set\_value( self, value )**

Abstract method to set data value.

Definition at line 481 of file baseclasses.py.

## 6.21.4 Member Data Documentation

### 6.21.4.1 gen.baseclasses.Leaf.data

Definition at line 456 of file baseclasses.py.

### 6.21.4.2 gen.baseclasses.Leaf.mandatory

Definition at line 458 of file baseclasses.py.

### 6.21.4.3 gen.baseclasses.Leaf.units

Definition at line 460 of file baseclasses.py.

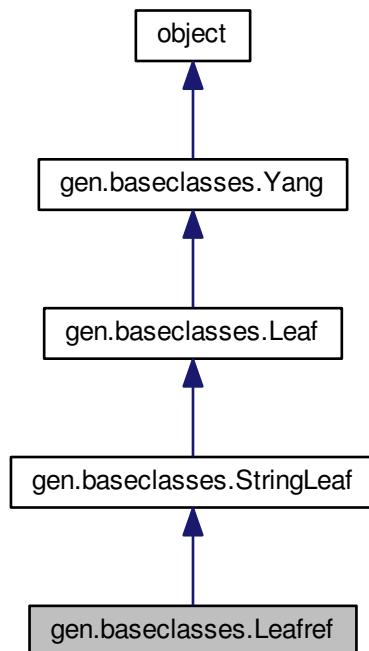
The documentation for this class was generated from the following file:

- [baseclasses.py](#)

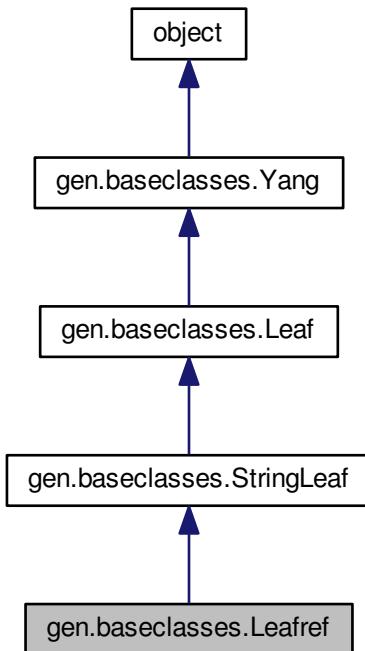
## 6.22 gen.baseclasses.Leafref Class Reference

Class defining `Leaf` extensions for stringleaf when its data references other instances.

Inheritance diagram for `gen.baseclasses.Leafref`:



Collaboration diagram for gen.baseclasses.Leafref:



## Public Member Functions

- def `__init__`
- def `set_value` (self, value)
 

*Sets data value as either a path or a Yang object* :param value: path string or Yang object :return: -.
- def `is_initialized` (self)
 

*Overrides Leaf method to check if data contains data and target is set* :param: - :return: boolean.
- def `get_as_text` (self)
 

*If data return its value as text, otherwise get relative path to target* :param: - :return: string.
- def `get_target` (self)
 

*Returns get path to target if data is initialized* :param: - :return: string.
- def `bind`

*Binds the target and add the referee to the referende list in the target.*
- def `unbind` (self)

## Public Attributes

- `target`
- `data`

### 6.22.1 Detailed Description

Class defining Leaf extensions for stringleaf when its data references other instances.

Definition at line 921 of file baseclasses.py.

## 6.22.2 Constructor & Destructor Documentation

6.22.2.1 `def gen.baseclasses.Leafref.__init__( self, tag, parent = None, value = None, units = "", mandatory = False )`

Definition at line 922 of file baseclasses.py.

## 6.22.3 Member Function Documentation

6.22.3.1 `def gen.baseclasses.Leafref.bind( self, relative = False )`

Binds the target and add the referee to the referende list in the target.

The path is updated to relative or absolut based on the parameter :param: relative: Boolean :return: -

Definition at line 1002 of file baseclasses.py.

6.22.3.2 `def gen.baseclasses.Leafref.get_as_text( self )`

If data return its value as text, otherwise get relative path to target :param: - :return: string.

Definition at line 974 of file baseclasses.py.

6.22.3.3 `def gen.baseclasses.Leafref.get_target( self )`

Returns get path to target if data is initialized :param: - :return: string.

Definition at line 989 of file baseclasses.py.

6.22.3.4 `def gen.baseclasses.Leafref.is_initialized( self )`

Overides `Leaf` method to check if data contains data and target is set :param: - :return: boolean.

Definition at line 962 of file baseclasses.py.

6.22.3.5 `def gen.baseclasses.Leafref.set_value( self, value )`

Sets data value as either a path or a `Yang` object :param value: path string or `Yang` object :return: -.

Definition at line 934 of file baseclasses.py.

6.22.3.6 `def gen.baseclasses.Leafref.unbind( self )`

Definition at line 1013 of file baseclasses.py.

## 6.22.4 Member Data Documentation

6.22.4.1 `gen.baseclasses.Leafref.data`

Definition at line 937 of file baseclasses.py.

6.22.4.2 `gen.baseclasses.Leafref.target`

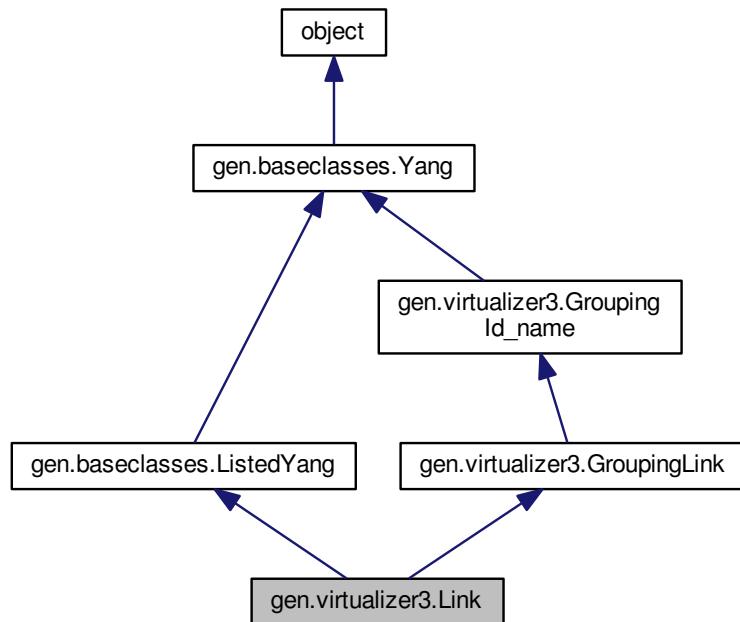
Definition at line 923 of file baseclasses.py.

The documentation for this class was generated from the following file:

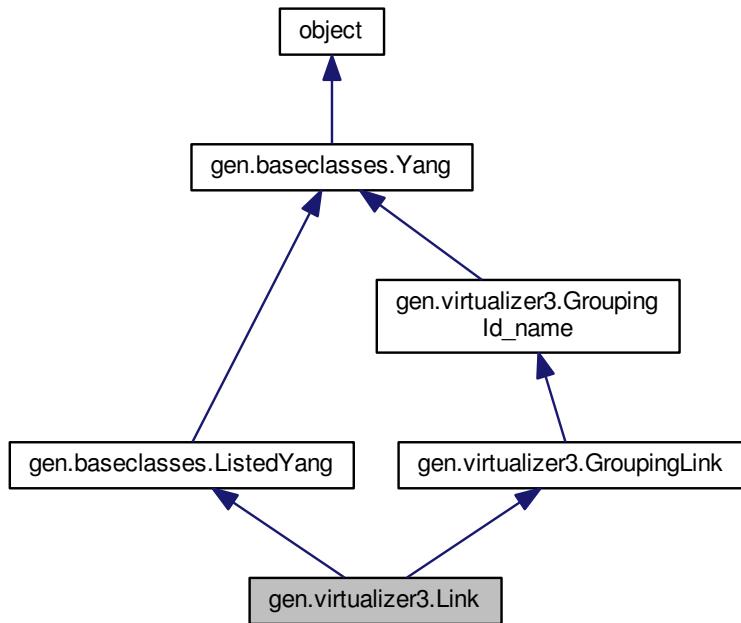
- [baseclasses.py](#)

## 6.23 gen.virtualizer3.Link Class Reference

Inheritance diagram for gen.virtualizer3.Link:



Collaboration diagram for gen.virtualizer3.Link:



## Public Member Functions

- `def __init__`

## Additional Inherited Members

### 6.23.1 Detailed Description

Definition at line 248 of file `virtualizer3.py`.

### 6.23.2 Constructor & Destructor Documentation

6.23.2.1 `def gen.virtualizer3.Link.__init__( self, tag = "link", parent = None, id = None, name = None, src = None, dst = None, resources = None )`

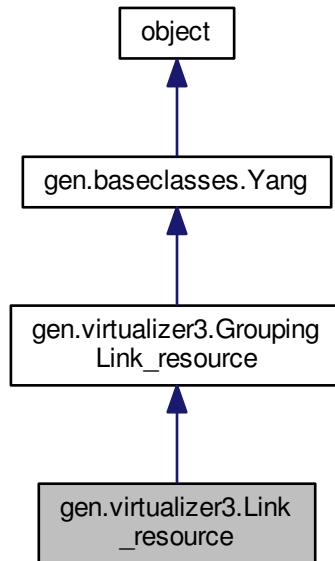
Definition at line 249 of file `virtualizer3.py`.

The documentation for this class was generated from the following file:

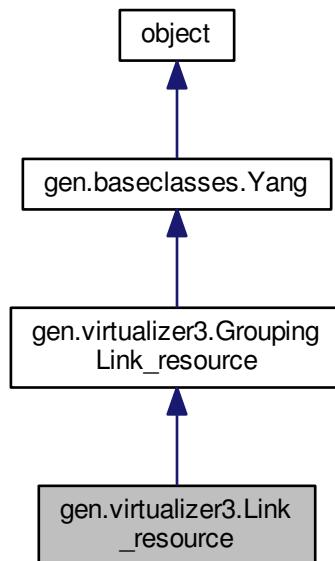
- `virtualizer3.py`

## 6.24 gen.virtualizer3.Link\_resource Class Reference

Inheritance diagram for gen.virtualizer3.Link\_resource:



Collaboration diagram for gen.virtualizer3.Link\_resource:



## Public Member Functions

- def `__init__`

## Additional Inherited Members

### 6.24.1 Detailed Description

Definition at line 280 of file `virtualizer3.py`.

### 6.24.2 Constructor & Destructor Documentation

6.24.2.1 def `gen.virtualizer3.Link_resource.__init__( self, tag = "resources", parent = None, delay = None, bandwidth = None )`

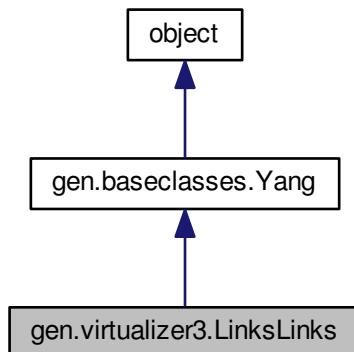
Definition at line 281 of file `virtualizer3.py`.

The documentation for this class was generated from the following file:

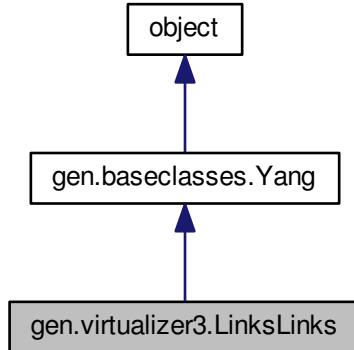
- `virtualizer3.py`

## 6.25 gen.virtualizer3.LinksLinks Class Reference

Inheritance diagram for `gen.virtualizer3.LinksLinks`:



Collaboration diagram for gen.virtualizer3.LinksLinks:



## Public Member Functions

- def `__init__`
- def `add` (self, item)
- def `remove` (self, item)
- def `__getitem__` (self, key)
- def `__iter__` (self)

## Public Attributes

- `link`

### 6.25.1 Detailed Description

Definition at line 309 of file virtualizer3.py.

### 6.25.2 Constructor & Destructor Documentation

#### 6.25.2.1 def gen.virtualizer3.LinksLinks.`__init__` ( `self`, `tag = "links"`, `parent = None` )

Definition at line 310 of file virtualizer3.py.

### 6.25.3 Member Function Documentation

#### 6.25.3.1 def gen.virtualizer3.LinksLinks.`__getitem__` ( `self`, `key` )

Definition at line 323 of file virtualizer3.py.

#### 6.25.3.2 def gen.virtualizer3.LinksLinks.`__iter__` ( `self` )

Definition at line 326 of file virtualizer3.py.

6.25.3.3 def gen.virtualizer3.LinksLinks.add ( self, item )

Definition at line 317 of file virtualizer3.py.

6.25.3.4 def gen.virtualizer3.LinksLinks.remove ( self, item )

Definition at line 320 of file virtualizer3.py.

#### 6.25.4 Member Data Documentation

6.25.4.1 gen.virtualizer3.LinksLinks.link

Definition at line 314 of file virtualizer3.py.

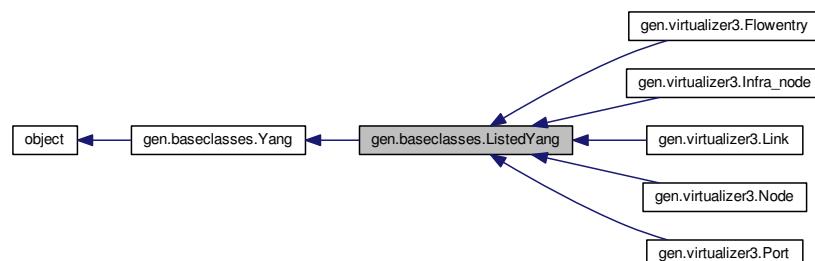
The documentation for this class was generated from the following file:

- [virtualizer3.py](#)

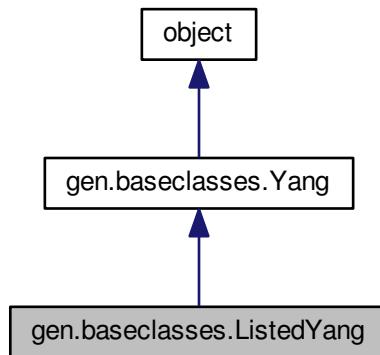
### 6.26 gen.baseclasses.ListedYang Class Reference

Class defined for Virtualizer classes inherit when modeled as list.

Inheritance diagram for gen.baseclasses.ListedYang:



Collaboration diagram for gen.baseclasses.ListedYang:



## Public Member Functions

- def `__init__`
- def `get_parent (self)`

*Returns parent's parent of ListedYang :param: - :return: instance of Yang.*
- def `keys (self)`

*Abstract method to get identifiers of class that inherit ListedYang.*
- def `get_key_tags (self)`

*Abstract method to get tags of class that inherit ListedYang.*
- def `get_path (self)`

*Returns path of ListedYang based on tags and values of its components :param: - :return: string.*
- def `empty_copy (self)`

*Performs copy of instance defining its components with deep copy :param: - :return: instance.*
- def `reduce (self, reference)`

*Delete instances which equivalently exist in the reference tree.*

## Additional Inherited Members

### 6.26.1 Detailed Description

Class defined for Virtualizer classes inherit when modeled as list.

Definition at line 1022 of file baseclasses.py.

### 6.26.2 Constructor & Destructor Documentation

#### 6.26.2.1 def gen.baseclasses.ListedYang.\_\_init\_\_( self, tag, keys, parent = None )

Definition at line 1023 of file baseclasses.py.

### 6.26.3 Member Function Documentation

#### 6.26.3.1 def gen.baseclasses.ListedYang.empty\_copy ( *self* )

Performs copy of instance defining its components with deep copy :param: - :return: instance.

Definition at line 1086 of file baseclasses.py.

#### 6.26.3.2 def gen.baseclasses.ListedYang.get\_key\_tags ( *self* )

Abstract method to get tags of class that inherit ListedYang.

Definition at line 1052 of file baseclasses.py.

#### 6.26.3.3 def gen.baseclasses.ListedYang.get\_parent ( *self* )

Returns parent's parent of ListedYang :param: - :return: instance of Yang.

Definition at line 1033 of file baseclasses.py.

#### 6.26.3.4 def gen.baseclasses.ListedYang.get\_path ( *self* )

Returns path of ListedYang based on tags and values of its components :param: - :return: string.

Definition at line 1066 of file baseclasses.py.

#### 6.26.3.5 def gen.baseclasses.ListedYang.keys ( *self* )

Abstract method to get identifiers of class that inherit ListedYang.

Definition at line 1040 of file baseclasses.py.

#### 6.26.3.6 def gen.baseclasses.ListedYang.reduce ( *self, reference* )

Delete instances which equivalently exist in the reference tree.

The call is recursive, a node is removed if and only if all of its children are removed. :param reference: Yang :return:

Definition at line 1099 of file baseclasses.py.

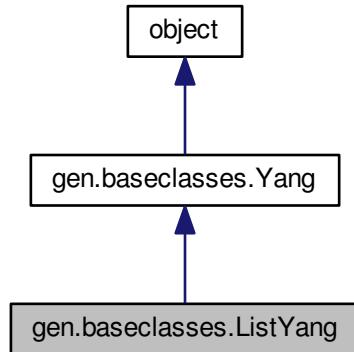
The documentation for this class was generated from the following file:

- [baseclasses.py](#)

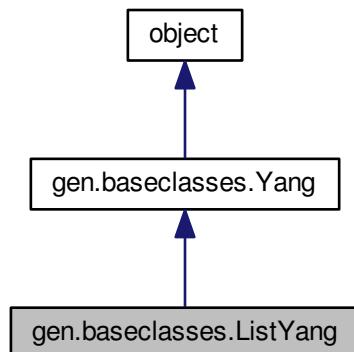
## 6.27 gen.baseclasses.ListYang Class Reference

Class to express list as dictionary.

Inheritance diagram for gen.baseclasses.ListYang:



Collaboration diagram for gen.baseclasses.ListYang:



## Additional Inherited Members

### 6.27.1 Detailed Description

Class to express list as dictionary.

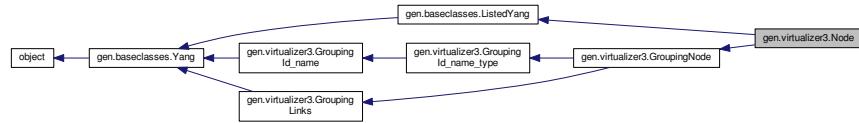
Definition at line 1117 of file baseclasses.py.

The documentation for this class was generated from the following file:

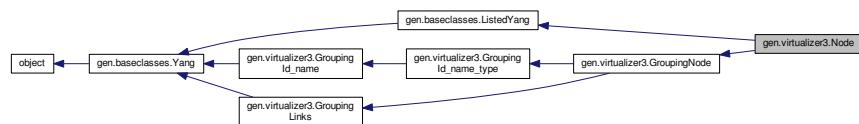
- [baseclasses.py](#)

## 6.28 gen.virtualizer3.Node Class Reference

Inheritance diagram for gen.virtualizer3.Node:



Collaboration diagram for gen.virtualizer3.Node:



### Public Member Functions

- `def __init__`

### Additional Inherited Members

#### 6.28.1 Detailed Description

Definition at line 264 of file virtualizer3.py.

#### 6.28.2 Constructor & Destructor Documentation

**6.28.2.1 def gen.virtualizer3.Node.\_\_init\_\_( self, tag = "node", parent = None, id = None, name = None, type = None, ports = None, links = None, resources = None )**

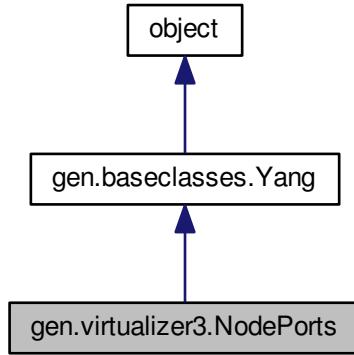
Definition at line 265 of file virtualizer3.py.

The documentation for this class was generated from the following file:

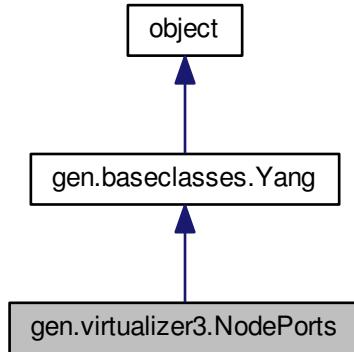
- `virtualizer3.py`

## 6.29 gen.virtualizer3.NodePorts Class Reference

Inheritance diagram for gen.virtualizer3.NodePorts:



Collaboration diagram for gen.virtualizer3.NodePorts:



### Public Member Functions

- def `__init__`
- def `add` (self, item)
- def `remove` (self, item)
- def `__getitem__` (self, key)
- def `__iter__` (self)

### Public Attributes

- `port`

### 6.29.1 Detailed Description

Definition at line 331 of file virtualizer3.py.

### 6.29.2 Constructor & Destructor Documentation

6.29.2.1 `def gen.virtualizer3.NodePorts.__init__ ( self, tag = "ports", parent = None )`

Definition at line 332 of file virtualizer3.py.

### 6.29.3 Member Function Documentation

6.29.3.1 `def gen.virtualizer3.NodePorts.__getitem__ ( self, key )`

Definition at line 345 of file virtualizer3.py.

6.29.3.2 `def gen.virtualizer3.NodePorts.__iter__ ( self )`

Definition at line 348 of file virtualizer3.py.

6.29.3.3 `def gen.virtualizer3.NodePorts.add ( self, item )`

Definition at line 339 of file virtualizer3.py.

6.29.3.4 `def gen.virtualizer3.NodePorts.remove ( self, item )`

Definition at line 342 of file virtualizer3.py.

### 6.29.4 Member Data Documentation

6.29.4.1 `gen.virtualizer3.NodePorts.port`

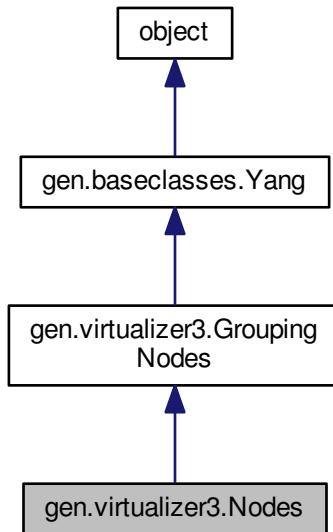
Definition at line 336 of file virtualizer3.py.

The documentation for this class was generated from the following file:

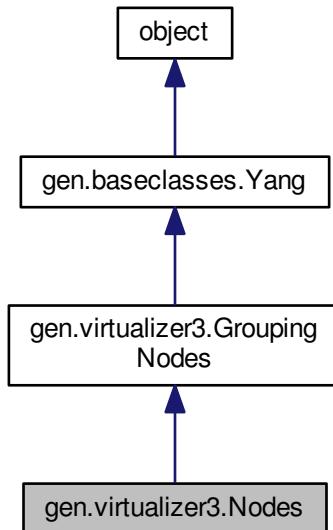
- [virtualizer3.py](#)

## 6.30 gen.virtualizer3.Nodes Class Reference

Inheritance diagram for gen.virtualizer3.Nodes:



Collaboration diagram for gen.virtualizer3.Nodes:



## Public Member Functions

- def init

## Additional Inherited Members

### **6.30.1 Detailed Description**

Definition at line 360 of file virtualizer3.py.

### 6.30.2 Constructor & Destructor Documentation

**6.30.2.1 def gen.virtualizer3.Nodes.\_init\_( self, tag = "NF\_instances", parent = None )**

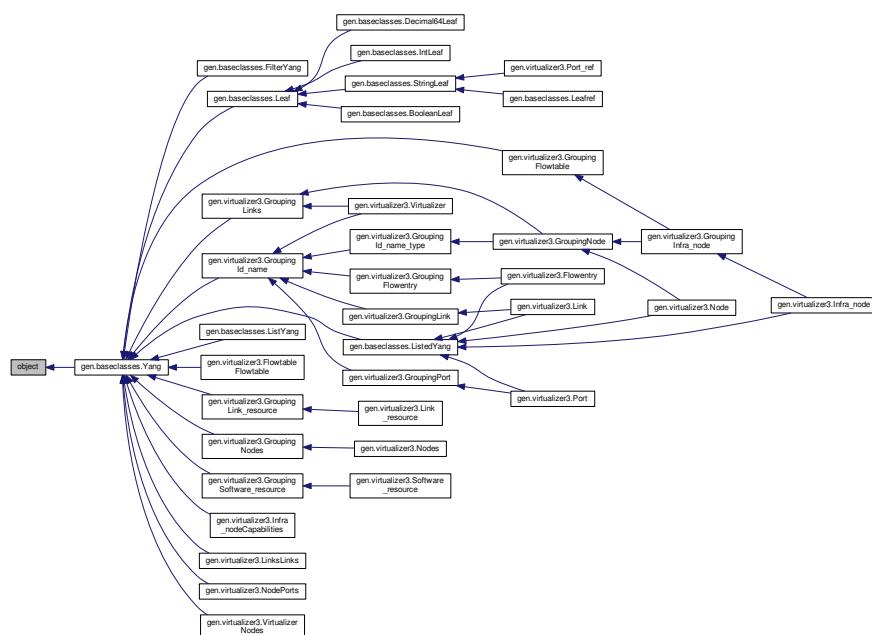
Definition at line 361 of file virtualizer3.py.

The documentation for this class was generated from the following file:

- virtualizer3.py

## 6.31 object Class Reference

### Inheritance diagram for object:

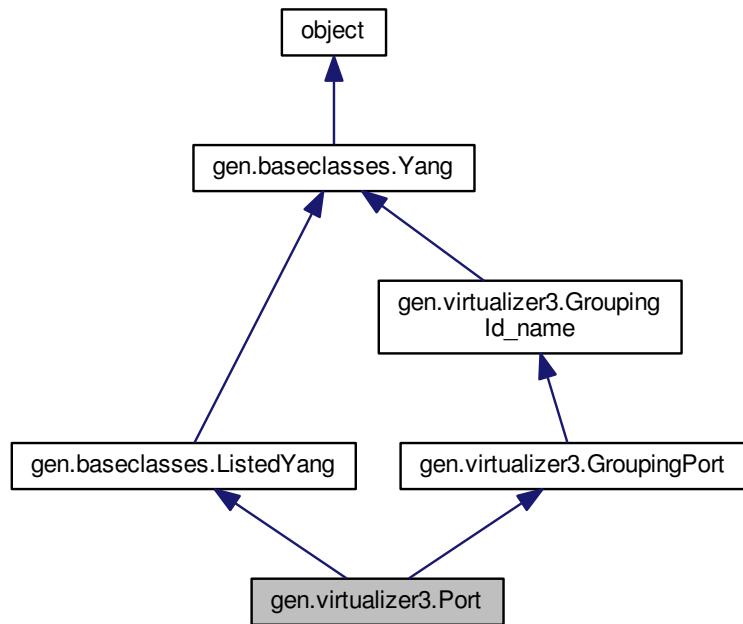


The documentation for this class was generated from the following file:

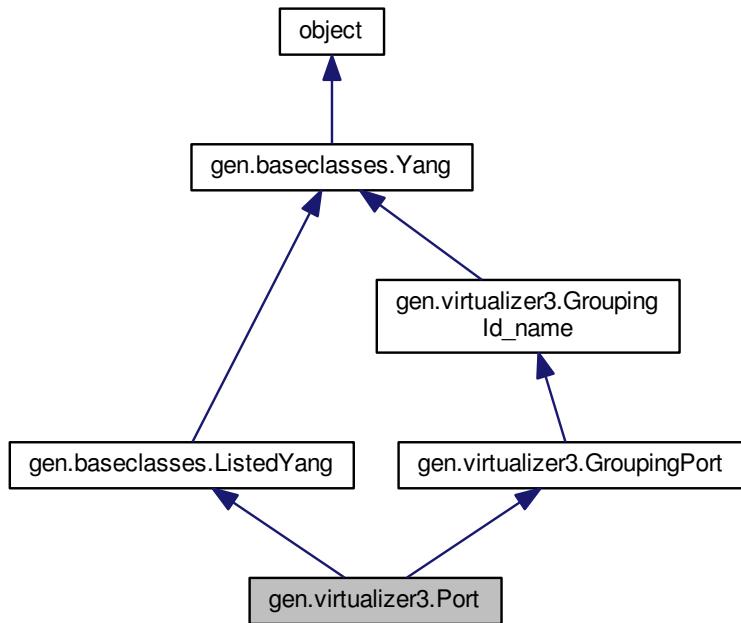
- `baseclasses.py`

## 6.32 gen.virtualizer3.Port Class Reference

Inheritance diagram for gen.virtualizer3.Port:



Collaboration diagram for gen.virtualizer3.Port:



## Public Member Functions

- `def __init__`

## Additional Inherited Members

### 6.32.1 Detailed Description

Definition at line 256 of file `virtualizer3.py`.

### 6.32.2 Constructor & Destructor Documentation

**6.32.2.1 `def gen.virtualizer3.Port.__init__( self, tag = "port", parent = None, id = None, name = None, port_type = None, capability = None, sap = None )`**

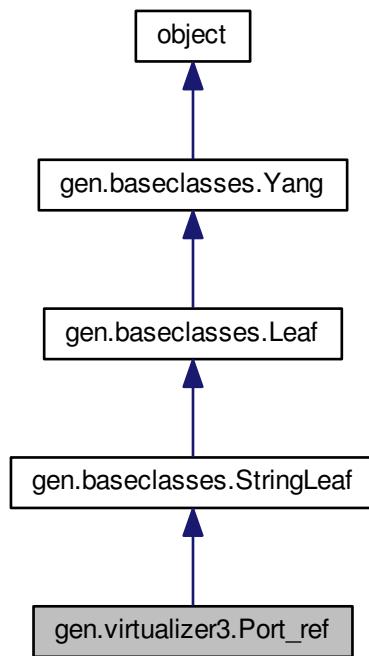
Definition at line 257 of file `virtualizer3.py`.

The documentation for this class was generated from the following file:

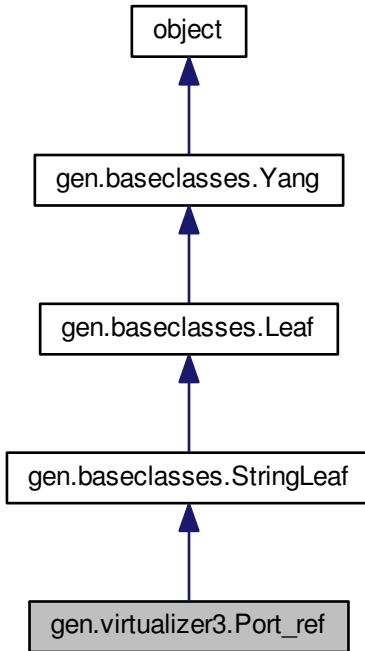
- `virtualizer3.py`

## 6.33 gen.virtualizer3.Port\_ref Class Reference

Inheritance diagram for gen.virtualizer3.Port\_ref:



Collaboration diagram for gen.virtualizer3.Port\_ref:



## Additional Inherited Members

### 6.33.1 Detailed Description

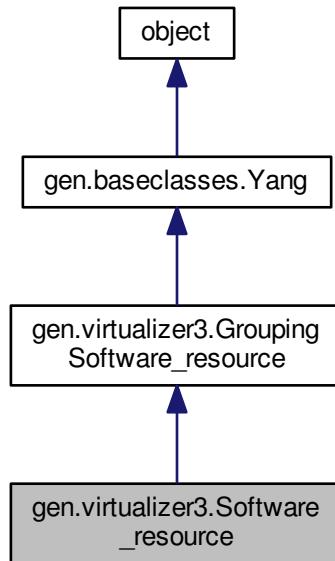
Definition at line 22 of file virtualizer3.py.

The documentation for this class was generated from the following file:

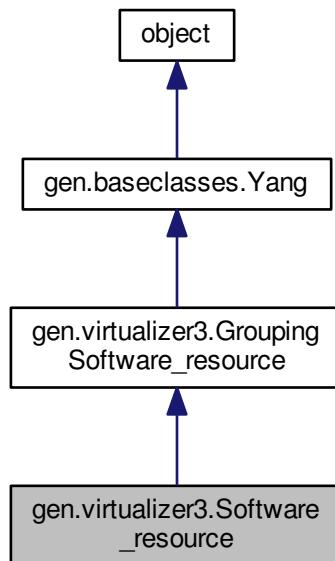
- [virtualizer3.py](#)

## 6.34 gen.virtualizer3.Software\_resource Class Reference

Inheritance diagram for gen.virtualizer3.Software\_resource:



Collaboration diagram for gen.virtualizer3.Software\_resource:



## Public Member Functions

- def `__init__`

## Additional Inherited Members

### 6.34.1 Detailed Description

Definition at line 353 of file `virtualizer3.py`.

### 6.34.2 Constructor & Destructor Documentation

6.34.2.1 def `gen.virtualizer3.Software_resource.__init__`( `self`, `tag = "resources"`, `parent = None`, `cpu = None`, `mem = None`, `storage = None` )

Definition at line 354 of file `virtualizer3.py`.

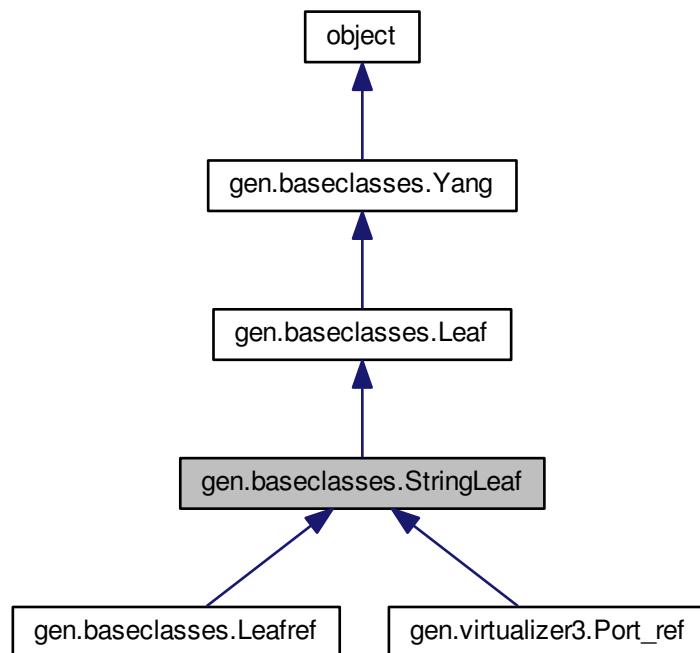
The documentation for this class was generated from the following file:

- `virtualizer3.py`

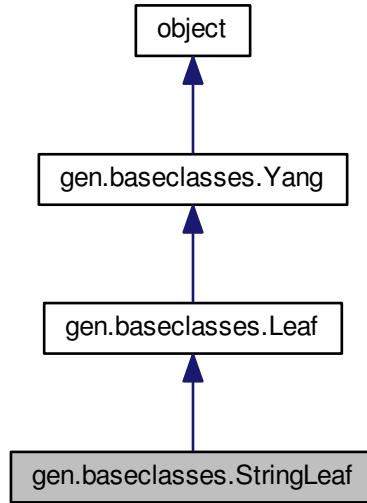
## 6.35 gen.baseclasses.StringLeaf Class Reference

Class defining `Leaf` with string extensions.

Inheritance diagram for `gen.baseclasses.StringLeaf`:



Collaboration diagram for gen.baseclasses.StringLeaf:



## Public Member Functions

- def [\\_\\_init\\_\\_](#)
- def [parse](#) (self, root)
 

*Abstract method to create instance class StringLeaf from XML string :param root: ElementTree :return: -.*
- def [get\\_as\\_text](#) (self)
 

*Returns data value as text :param: - :return: string.*
- def [get\\_value](#) (self)
 

*Returns data value :param: - :return: string.*
- def [set\\_value](#) (self, value)
 

*Sets data value :param value: string :return: -.*

## Public Attributes

- [data](#)

### 6.35.1 Detailed Description

Class defining [Leaf](#) with string extensions.

Definition at line 595 of file baseclasses.py.

### 6.35.2 Constructor & Destructor Documentation

#### 6.35.2.1 def gen.baseclasses.StringLeaf.\_\_init\_\_( self, tag, parent = None, value = None, units = " ", mandatory = False )

Definition at line 596 of file baseclasses.py.

### 6.35.3 Member Function Documentation

#### 6.35.3.1 def gen.baseclasses.StringLeaf.get\_as\_text( self )

Returns data value as text :param: - :return: string.

Definition at line 629 of file baseclasses.py.

#### 6.35.3.2 def gen.baseclasses.StringLeaf.get\_value( self )

Returns data value :param: - :return: string.

Definition at line 640 of file baseclasses.py.

#### 6.35.3.3 def gen.baseclasses.StringLeaf.parse( self, root )

Abstract method to create instance class `StringLeaf` from XML string :param root: ElementTree :return: -.

Definition at line 611 of file baseclasses.py.

#### 6.35.3.4 def gen.baseclasses.StringLeaf.set\_value( self, value )

Sets data value :param value: string :return: -.

Definition at line 649 of file baseclasses.py.

### 6.35.4 Member Data Documentation

#### 6.35.4.1 gen.baseclasses.StringLeaf.data

Definition at line 618 of file baseclasses.py.

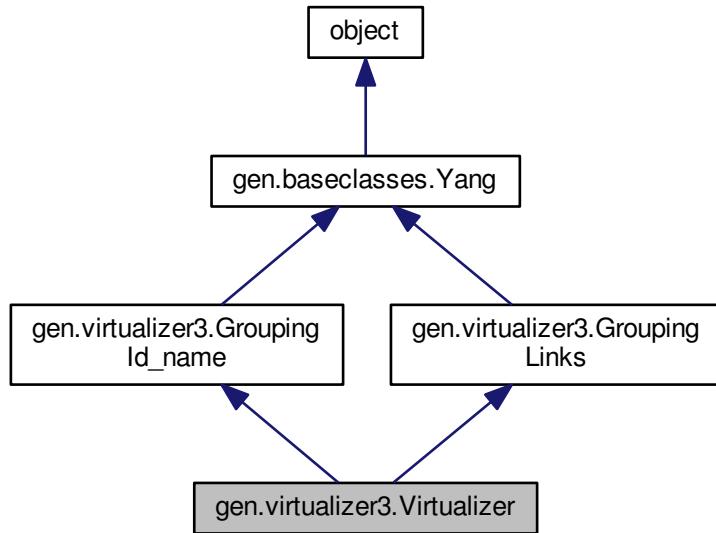
The documentation for this class was generated from the following file:

- [baseclasses.py](#)

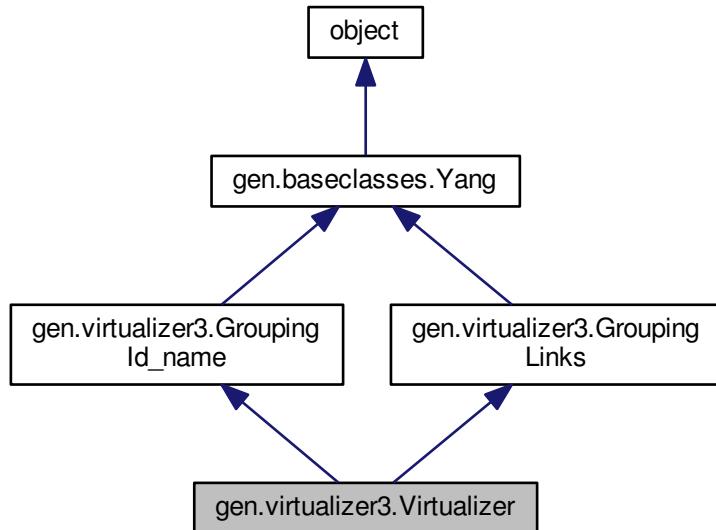
## 6.36 gen.virtualizer3.Virtualizer Class Reference

Container for a single virtualizer.

Inheritance diagram for gen.virtualizer3.Virtualizer:



Collaboration diagram for gen.virtualizer3.Virtualizer:



## Public Member Functions

- def `__init__`

## Public Attributes

- [nodes](#)

### 6.36.1 Detailed Description

Container for a single virtualizer.

Definition at line 383 of file `virtualizer3.py`.

### 6.36.2 Constructor & Destructor Documentation

```
6.36.2.1 def gen.virtualizer3.Virtualizer.__init__( self, tag = "virtualizer", parent = None, id = None, name = None, nodes = None, links = None )
```

Definition at line 384 of file `virtualizer3.py`.

### 6.36.3 Member Data Documentation

```
6.36.3.1 gen.virtualizer3.Virtualizer.nodes
```

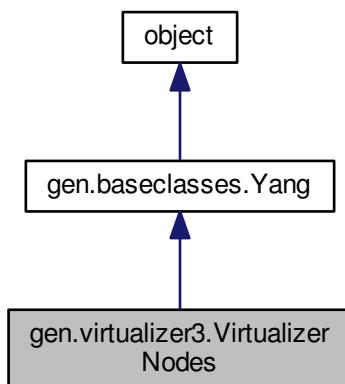
Definition at line 389 of file `virtualizer3.py`.

The documentation for this class was generated from the following file:

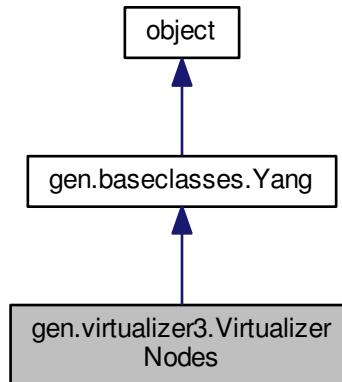
- [virtualizer3.py](#)

## 6.37 gen.virtualizer3.VirtualizerNodes Class Reference

Inheritance diagram for `gen.virtualizer3.VirtualizerNodes`:



Collaboration diagram for gen.virtualizer3.VirtualizerNodes:



## Public Member Functions

- def `__init__`
- def `add` (`self`, `item`)
- def `remove` (`self`, `item`)
- def `__getitem__` (`self`, `key`)
- def `__iter__` (`self`)

## Public Attributes

- `node`

### 6.37.1 Detailed Description

Definition at line 398 of file virtualizer3.py.

### 6.37.2 Constructor & Destructor Documentation

#### 6.37.2.1 def gen.virtualizer3.VirtualizerNodes.\_\_init\_\_ ( `self`, `tag = "nodes"`, `parent = None` )

Definition at line 399 of file virtualizer3.py.

### 6.37.3 Member Function Documentation

#### 6.37.3.1 def gen.virtualizer3.VirtualizerNodes.\_\_getitem\_\_ ( `self`, `key` )

Definition at line 412 of file virtualizer3.py.

#### 6.37.3.2 def gen.virtualizer3.VirtualizerNodes.\_\_iter\_\_ ( `self` )

Definition at line 415 of file virtualizer3.py.

## 6.37.3.3 def gen.virtualizer3.VirtualizerNodes.add ( self, item )

Definition at line 406 of file virtualizer3.py.

## 6.37.3.4 def gen.virtualizer3.VirtualizerNodes.remove ( self, item )

Definition at line 409 of file virtualizer3.py.

## 6.37.4 Member Data Documentation

## 6.37.4.1 gen.virtualizer3.VirtualizerNodes.node

Definition at line 403 of file virtualizer3.py.

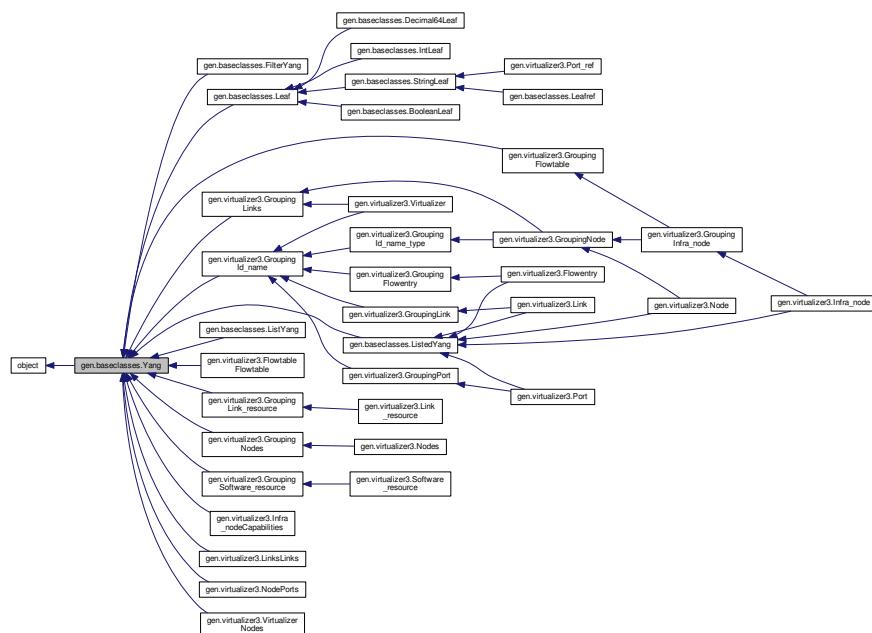
The documentation for this class was generated from the following file:

- [virtualizer3.py](#)

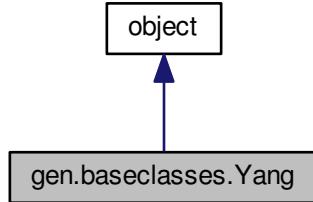
## 6.38 gen.baseclasses.Yang Class Reference

Class defining the root attributes and methods for all Virtualizer classes.

Inheritance diagram for gen.baseclasses.Yang:



Collaboration diagram for gen.baseclasses.Yang:



## Public Member Functions

- def `__init__`
- def `get_parent (self)`

*Returns the parent in the class subtree.*
- def `set_parent (self, parent)`

*Set the parent to point to the next node up in the Yang class instance tree :param parent: Yang :return: -.*
- def `get_tag (self)`

*Returns the YANG tag for the class.*
- def `set_tag (self, tag)`

*Set the YANG tag for the class :param tag: string :return: -.*
- def `xml`

*Dump the class subtree as XML string :return: string.*
- def `get_as_text`

*Dump the class subtree as TEXT string :return: string.*
- def `reduce`

*Delete instances which equivalently exist in the reference tree.*
- def `get_path (self)`

*Returns the complete path (since the root) of the instance of Yang :param: - :return: string.*
- def `walk_path (self, path)`

*Follows the specified path to return the instance it represents :param path: string :return: attribute instance of Yang.*
- def `get_rel_path (self, target)`

*Returns the relative path from self to the target :param target: instance of Yang :return: string.*
- def `parse`

*Class method to create virtualizer from XML string :param parent: Yang :param root: ElementTree :return: class instance of Yang.*
- def `__str__ (self)`

*Override str methor to dump the class subtree as XML string :return: string.*
- def `contains_operation`

*Verifies if the instance contains operation set for any of its attributes :param operation: string :return: boolean.*
- def `get_operation (self)`

*Returns the \_operation attribute :param: - :return: string.*
- def `set_operation`

*Defines operation for instance :param operation: string :return: -.*
- def `is_initialized (self)`

*Check if any of the attributes of instance are initialized, returns True if yes :param: - :return: boolean.*

- def `__eq__` (self, other)
 

*Check if all the attributes and class attributes are the same in instance and other, returns True if yes :param other: instance of Yang :return: boolean.*
- def `merge` (self, target)
 

*Merge instance with target recursively, keeping in instance only attributes initialized :param target: instance of Yang :return: -.*
- def `empty_copy` (self)
 

*Performs copy of instance of Yang :param: - :return: instance copy (of Yang)*
- def `full_copy` (self)
 

*Performs deepcopy of instance of Yang :param: - :return: instance copy (of Yang)*
- def `delete` (self)
 

*Remove element when ListYang and set to None when Leaf :param: - :return: -.*
- def `set_referred` (self, leaf\_ref)
 

*Append in referred names of leafs referred (children of) by instance of Yang :param leaf\_ref: LeafRef :return: -.*
- def `unset_referred` (self, leaf\_ref)
 

*Append in referred names of leafs referred (children of) by instance of Yang :param leaf\_ref: LeafRef :return: -.*
- def `bind`

*Binds all elements of self attributes :param: relative: Boolean :return: -.*

## Public Attributes

- `operation`

### 6.38.1 Detailed Description

Class defining the root attributes and methods for all Virtualizer classes.

Definition at line 28 of file baseclasses.py.

### 6.38.2 Constructor & Destructor Documentation

#### 6.38.2.1 def gen.baseclasses.Yang.\_\_init\_\_ ( self, tag, parent = None )

Definition at line 29 of file baseclasses.py.

### 6.38.3 Member Function Documentation

#### 6.38.3.1 def gen.baseclasses.Yang.\_\_eq\_\_ ( self, other )

Check if all the attributes and class attributes are the same in instance and other, returns True if yes :param other: instance of Yang :return: boolean.

Definition at line 311 of file baseclasses.py.

#### 6.38.3.2 def gen.baseclasses.Yang.\_\_str\_\_ ( self )

Overide str methor to dump the class subtree as XML string :return: string.

Definition at line 253 of file baseclasses.py.

## Class Documentation

6.38.3.3 def gen.baseclasses.Yang.bind( *self*, *relative* = False )

Binds all elements of self attributes :param: relative: Boolean :return: -.

Definition at line 411 of file baseclasses.py.

6.38.3.4 def gen.baseclasses.Yang.contains\_operation( *self*, *operation* = "delete" )

Verifies if the instance contains operation set for any of its attributes :param operation: string :return: boolean.

Definition at line 262 of file baseclasses.py.

6.38.3.5 def gen.baseclasses.Yang.delete( *self* )

Remove element when [ListYang](#) and set to None when [Leaf](#) :param: - :return: -.

Definition at line 378 of file baseclasses.py.

6.38.3.6 def gen.baseclasses.Yang.empty\_copy( *self* )

Performs copy of instance of [Yang](#) :param: - :return: instance copy (of [Yang](#))

Definition at line 360 of file baseclasses.py.

6.38.3.7 def gen.baseclasses.Yang.full\_copy( *self* )

Performs deepcopy of instance of [Yang](#) :param: - :return: instance copy (of [Yang](#))

Definition at line 369 of file baseclasses.py.

6.38.3.8 def gen.baseclasses.Yang.get\_as\_text( *self*, *ordered* = True )

Dump the class subtree as TEXT string :return: string.

Definition at line 86 of file baseclasses.py.

6.38.3.9 def gen.baseclasses.Yang.get\_operation( *self* )

Returns the \_operation attribute :param: - :return: string.

Definition at line 277 of file baseclasses.py.

6.38.3.10 def gen.baseclasses.Yang.get\_parent( *self* )

Returns the parent in the class subtree.

:return: [Yang](#)

Definition at line 41 of file baseclasses.py.

6.38.3.11 def gen.baseclasses.Yang.get\_path( *self* )

Returns the complete path (since the root) of the instance of [Yang](#) :param: - :return: string.

Definition at line 140 of file baseclasses.py.

**6.38.3.12 def gen.baseclasses.Yang.get\_rel\_path ( self, target )**

Returns the relative path from self to the target :param target: instance of Yang :return: string.

Definition at line 189 of file baseclasses.py.

**6.38.3.13 def gen.baseclasses.Yang.get\_tag ( self )**

Returns the YANG tag for the class.

:return: string

Definition at line 58 of file baseclasses.py.

**6.38.3.14 def gen.baseclasses.Yang.is\_initialized ( self )**

Check if any of the attributes of instance are initialized, returns True if yes :param: - :return: boolean.

Definition at line 298 of file baseclasses.py.

**6.38.3.15 def gen.baseclasses.Yang.merge ( self, target )**

Merge instance with target recursively, keeping in instance only attributes initialized :param target: instance of Yang :return: -.

Definition at line 340 of file baseclasses.py.

**6.38.3.16 def gen.baseclasses.Yang.parse ( cls, parent = None, root = None )**

Class method to create virtualizer from XML string :param parent: Yang :param root: ElementTree :return: class instance of Yang.

Definition at line 214 of file baseclasses.py.

**6.38.3.17 def gen.baseclasses.Yang.reduce ( self, reference, ignores = None )**

Delete instances which equivalently exist in the reference tree.

The call is recursive, a node is removed if and only if all of its children are removed. :param reference: Yang :return: True if object to be removed otherwise False

Definition at line 108 of file baseclasses.py.

**6.38.3.18 def gen.baseclasses.Yang.set\_operation ( self, operation = "delete" )**

Defines operation for instance :param operation: string :return: -.

Definition at line 286 of file baseclasses.py.

**6.38.3.19 def gen.baseclasses.Yang.set\_parent ( self, parent )**

Set the parent to point to the next node up in the Yang class instance tree :param parent: Yang :return: -.

Definition at line 50 of file baseclasses.py.

**6.38.3.20 def gen.baseclasses.Yang.set\_referred ( self, leaf\_ref )**

Append in referred names of leafs referred (children of) by instance of Yang :param leaf\_ref: LeafRef :return: -.

Definition at line 391 of file baseclasses.py.

**6.38.3.21 def gen.baseclasses.Yang.set\_tag ( self, tag )**

Set the YANG tag for the class :param tag: string :return: -.

Definition at line 67 of file baseclasses.py.

**6.38.3.22 def gen.baseclasses.Yang.unset\_referred ( self, leaf\_ref )**

Append in referred names of leafs referred (children of) by instance of Yang :param leaf\_ref: LeafRef :return: -.

Definition at line 401 of file baseclasses.py.

**6.38.3.23 def gen.baseclasses.Yang.walk\_path ( self, path )**

Follows the specified path to return the instance it represents :param path: string :return: attribute instance of Yang.

Definition at line 152 of file baseclasses.py.

**6.38.3.24 def gen.baseclasses.Yang.xml ( self, ordered = True )**

Dump the class subtree as XML string :return: string.

Definition at line 75 of file baseclasses.py.

## 6.38.4 Member Data Documentation

**6.38.4.1 gen.baseclasses.Yang.operation**

Definition at line 447 of file baseclasses.py.

The documentation for this class was generated from the following file:

- [baseclasses.py](#)

# Chapter 7

## File Documentation

### 7.1 baseclasses.py File Reference

#### Classes

- class `gen.baseclasses.Yang`  
*Class defining the root attributes and methods for all Virtualizer classes.*
- class `gen.baseclasses.Leaf`  
*Class defining Leaf basis with attributes and methods.*
- class `gen.baseclasses.StringLeaf`  
*Class defining Leaf with string extensions.*
- class `gen.baseclasses.IntLeaf`  
*Class defining Leaf with integer extensions (e.g., range)*
- class `gen.baseclasses.Decimal64Leaf`  
*Class defining Leaf with decimal extensions (e.g., dec\_range)*
- class `gen.baseclasses.BooleanLeaf`  
*Class defining Leaf with boolean extensions (e.g., True or False)*
- class `gen.baseclasses.Leafref`  
*Class defining Leaf extensions for stringleaf when its data references other instances.*
- class `gen.baseclasses.ListedYang`  
*Class defined for Virtualizer classes inherit when modeled as list.*
- class `gen.baseclasses.ListYang`  
*Class to express list as dictionary.*
- class `gen.baseclasses.FilterYang`

#### Namespaces

- `gen.baseclasses`

#### Functions

- def `gen.baseclasses.__init__`
- def `gen.baseclasses.get_type (self)`  
*Returns class which references elements of \_data OrderedDict :param: - :return: Yang subclass.*
- def `gen.baseclasses.set_type (self, type)`  
*Sets class which references elements of \_data OrderedDict :param: Yang subclass :return: -.*
- def `gen.baseclasses.keys (self)`

- def `gen.baseclasses.values` (self)
 

*Returns indices of ListYang dictionary :param: - :return: list.*
- def `gen.baseclasses.iterkeys` (self)
 

*Returns iterator of keys of ListYang dictionary :param: - :return: iterator.*
- def `gen.baseclasses.itervalues` (self)
 

*Returns iterator of values of ListYang dictionary :param: - :return: list.*
- def `gen.baseclasses.items` (self)
 

*Returns items of ListYang dictionary :param: - :return: list.*
- def `gen.baseclasses.iteritems` (self)
 

*Returns iterator of items of ListYang dictionary :param: - :return: list.*
- def `gen.baseclasses.has_key` (self, key)
 

*Returns if key is in ListYang dictionary :param key: string :return: boolean.*
- def `gen.baseclasses.has_value` (self, value)
 

*Returns if value is in ListYang dictionary values :param value: string or instance :return: boolean.*
- def `gen.baseclasses.length` (self)
 

*Returns length of ListYang dictionary :param: - :return: int.*
- def `gen.baseclasses.is_initialized` (self)
 

*Returns if ListYang dictionary contains elements :param: - :return: boolean.*
- def `gen.baseclasses.add` (self, item)
 

*add single or a list of items :param item: a single ListedYang or a list of ListedYang derivates :return: item*
- def `gen.baseclasses.remove` (self, item)
 

*remove a single element from the list based on a key or a ListedYang :param item: key (single or composit) or a ListedYang :return: item*
- def `gen.baseclasses.__iter__` (self)
 

*Returns iterator of ListYang dict :param: - :return: iterator.*
- def `gen.baseclasses.next` (self)
 

*Go to next element of ListYang dictionary :param: - :return: -.*
- def `gen.baseclasses.__getitem__` (self, key)
 

*Returns ListYang value if key in dictionary :param key: string :return: instance.*
- def `gen.baseclasses.__setitem__` (self, key, value)
 

*Fill ListYang dict with key associated to value :param key: string :param value: string or instance :return: -.*
- def `gen.baseclasses.clear_data` (self)
 

*Clear ListYang dict :param: - :return: -.*
- def `gen.baseclasses.reduce` (self, reference)
 

*Check if all keys of reference are going to be reduced and erase their values if yes :param reference: ListYang :return: boolean.*
- def `gen.baseclasses.merge` (self, target)
 

*Add items of target if their keys do not exist in self instance :param target: ListYang :return: -.*
- def `gen.baseclasses.__eq__` (self, other)
 

*Check if dict of other ListYang is equal :param other: ListYang :return: boolean.*
- def `gen.baseclasses.contains_operation` (self, operation)
 

*Check if any of items have operation set :param operation: string :return: boolean.*
- def `gen.baseclasses.set_operation`

*Set operation for all of items in ListYang dict' :param operation: string :return: -.*
- def `gen.baseclasses.bind`

## Variables

- string `gen.baseclasses.__copyright__` = "Copyright Ericsson Hungary Ltd., 2015"
- `gen.baseclasses._data`
- `gen.baseclasses._type`

## 7.2 virtualizer3.py File Reference

### Classes

- class `gen.virtualizer3.Port_ref`
- class `gen.virtualizer3.GroupingId_name`
- class `gen.virtualizer3.GroupingId_name_type`
- class `gen.virtualizer3.GroupingPort`
- class `gen.virtualizer3.GroupingLink_resource`
- class `gen.virtualizer3.GroupingFlowentry`
- class `gen.virtualizer3.GroupingFlowtable`
- class `gen.virtualizer3.GroupingLink`
- class `gen.virtualizer3.GroupingLinks`
- class `gen.virtualizer3.GroupingSoftware_resource`
- class `gen.virtualizer3.GroupingNode`
  - Any node: infrastructure or NFs.*
- class `gen.virtualizer3.GroupingNodes`
- class `gen.virtualizer3.GroupingInfra_node`
- class `gen.virtualizer3.Flowentry`
- class `gen.virtualizer3.Link`
- class `gen.virtualizer3.Port`
- class `gen.virtualizer3.Node`
- class `gen.virtualizer3.Infra_node`
- class `gen.virtualizer3.Link_resource`
- class `gen.virtualizer3.FlowtableFlowtable`
- class `gen.virtualizer3.LinksLinks`
- class `gen.virtualizer3.NodePorts`
- class `gen.virtualizer3.Software_resource`
- class `gen.virtualizer3.Nodes`
- class `gen.virtualizer3.Infra_nodeCapabilities`
- class `gen.virtualizer3.Virtualizer`
  - Container for a single virtualizer.*
- class `gen.virtualizer3.VirtualizerNodes`

### Namespaces

- `gen.virtualizer3`

### Variables

- string `gen.virtualizer3.__copyright__` = "Copyright Ericsson Hungary Ltd., 2015"

# Index

\_\_copyright\_\_  
gen::baseclasses, 240  
gen::virtualizer3, 241

\_\_eq\_\_  
gen::baseclasses, 237  
gen::baseclasses::Leaf, 283  
gen::baseclasses::Yang, 315

\_\_getitem\_\_  
gen::baseclasses, 237  
gen::virtualizer3::FlowtableFlowtable, 253  
gen::virtualizer3::GroupingNodes, 272  
gen::virtualizer3::LinksLinks, 292  
gen::virtualizer3::NodePorts, 299  
gen::virtualizer3::VirtualizerNodes, 312

\_\_init\_\_  
gen::baseclasses, 237  
gen::baseclasses::BooleanLeaf, 243  
gen::baseclasses::Decimal64Leaf, 246  
gen::baseclasses::FilterYang, 249  
gen::baseclasses::IntLeaf, 281  
gen::baseclasses::Leaf, 283  
gen::baseclasses::Leafref, 287  
gen::baseclasses::ListedYang, 294  
gen::baseclasses::StringLeaf, 308  
gen::baseclasses::Yang, 315  
gen::virtualizer3::Flowentry, 251  
gen::virtualizer3::FlowtableFlowtable, 253  
gen::virtualizer3::GroupingFlowentry, 255  
gen::virtualizer3::GroupingFlowtable, 258  
gen::virtualizer3::GroupingId\_name, 259  
gen::virtualizer3::GroupingId\_name\_type, 261  
gen::virtualizer3::GroupingInfra\_node, 262  
gen::virtualizer3::GroupingLink, 264  
gen::virtualizer3::GroupingLink\_resource, 266  
gen::virtualizer3::GroupingLinks, 268  
gen::virtualizer3::GroupingNode, 270  
gen::virtualizer3::GroupingNodes, 272  
gen::virtualizer3::GroupingPort, 274  
gen::virtualizer3::GroupingSoftware\_resource, 276  
gen::virtualizer3::Infra\_node, 277  
gen::virtualizer3::Infra\_nodeCapabilities, 279  
gen::virtualizer3::Link, 289  
gen::virtualizer3::Link\_resource, 291  
gen::virtualizer3::LinksLinks, 292  
gen::virtualizer3::Node, 297  
gen::virtualizer3::NodePorts, 299  
gen::virtualizer3::Nodes, 301  
gen::virtualizer3::Port, 303  
gen::virtualizer3::Software\_resource, 307

gen::virtualizer3::Virtualizer, 311  
gen::virtualizer3::VirtualizerNodes, 312

\_\_iter\_\_  
gen::baseclasses, 237  
gen::virtualizer3::FlowtableFlowtable, 253  
gen::virtualizer3::GroupingNodes, 272  
gen::virtualizer3::LinksLinks, 292  
gen::virtualizer3::NodePorts, 299  
gen::virtualizer3::VirtualizerNodes, 312

\_\_setitem\_\_  
gen::baseclasses, 237

\_\_str\_\_  
gen::baseclasses::Yang, 315

\_data  
gen::baseclasses, 240

\_type  
gen::baseclasses, 240

action  
gen::virtualizer3::GroupingFlowentry, 256

add  
gen::baseclasses, 237  
gen::virtualizer3::FlowtableFlowtable, 253  
gen::virtualizer3::GroupingNodes, 272  
gen::virtualizer3::LinksLinks, 292  
gen::virtualizer3::NodePorts, 299  
gen::virtualizer3::VirtualizerNodes, 312

bandwidth  
gen::virtualizer3::GroupingLink\_resource, 266

baseclasses.py, 319

bind  
gen::baseclasses, 237  
gen::baseclasses::Leafref, 287  
gen::baseclasses::Yang, 315

capabilities  
gen::virtualizer3::GroupingInfra\_node, 262

capability  
gen::virtualizer3::GroupingPort, 274

check\_range  
gen::baseclasses::Decimal64Leaf, 246  
gen::baseclasses::IntLeaf, 281

clear\_data  
gen::baseclasses, 237  
gen::baseclasses::Leaf, 283

contains\_operation  
gen::baseclasses, 237  
gen::baseclasses::Yang, 316

cpu

**Index**

gen::virtualizer3::GroupingSoftware\_resource, 276  
**data**  
 gen::baseclasses::BooleanLeaf, 244  
 gen::baseclasses::Decimal64Leaf, 247  
 gen::baseclasses::IntLeaf, 281  
 gen::baseclasses::Leaf, 285  
 gen::baseclasses::Leafref, 287  
 gen::baseclasses::StringLeaf, 309  
**dec\_range**  
 gen::baseclasses::Decimal64Leaf, 247  
**delay**  
 gen::virtualizer3::GroupingLink\_resource, 266  
**delete**  
 gen::baseclasses::Leaf, 283  
 gen::baseclasses::Yang, 316  
**dst**  
 gen::virtualizer3::GroupingLink, 264  
**empty\_copy**  
 gen::baseclasses::ListedYang, 295  
 gen::baseclasses::Yang, 316  
**filter\_xml**  
 gen::baseclasses::FilterYang, 249  
**flowentry**  
 gen::virtualizer3::FlowtableFlowtable, 253  
**flowtable**  
 gen::virtualizer3::GroupingFlowtable, 258  
**fraction\_digits**  
 gen::baseclasses::Decimal64Leaf, 247  
**full\_copy**  
 gen::baseclasses::Yang, 316  
**gen**, 235  
 gen.baseclasses, 235  
 gen.baseclasses.BooleanLeaf, 242  
 gen.baseclasses.Decimal64Leaf, 244  
 gen.baseclasses.FilterYang, 248  
 gen.baseclasses.IntLeaf, 279  
 gen.baseclasses.Leaf, 282  
 gen.baseclasses.Leafref, 285  
 gen.baseclasses.ListYang, 295  
 gen.baseclasses.ListedYang, 293  
 gen.baseclasses.StringLeaf, 307  
 gen.baseclasses.Yang, 313  
 gen.virtualizer3, 240  
 gen.virtualizer3.Flowentry, 250  
 gen.virtualizer3.FlowtableFlowtable, 252  
 gen.virtualizer3.GroupingFlowentry, 254  
 gen.virtualizer3.GroupingFlowtable, 257  
 gen.virtualizer3.GroupingId\_name, 258  
 gen.virtualizer3.GroupingId\_name\_type, 260  
 gen.virtualizer3.GroupingInfra\_node, 262  
 gen.virtualizer3.GroupingLink, 263  
 gen.virtualizer3.GroupingLink\_resource, 265  
 gen.virtualizer3.GroupingLinks, 267  
 gen.virtualizer3.GroupingNode, 268  
 gen.virtualizer3.GroupingNodes, 271  
 gen.virtualizer3.GroupingPort, 273  
 gen.virtualizer3.GroupingSoftware\_resource, 275  
 gen.virtualizer3.Infra\_node, 277  
 gen.virtualizer3.Infra\_nodeCapabilities, 278  
 gen.virtualizer3.Link, 288  
 gen.virtualizer3.Link\_resource, 290  
 gen.virtualizer3.LinksLinks, 291  
 gen.virtualizer3.Node, 297  
 gen.virtualizer3.NodePorts, 298  
 gen.virtualizer3.Nodes, 300  
 gen.virtualizer3.Port, 302  
 gen.virtualizer3.Port\_ref, 304  
 gen.virtualizer3.Software\_resource, 306  
 gen.virtualizer3.Virtualizer, 309  
 gen.virtualizer3.VirtualizerNodes, 311  
**gen::baseclasses**  
 \_\_copyright\_\_, 240  
 \_\_eq\_\_, 237  
 \_\_getitem\_\_, 237  
 \_\_init\_\_, 237  
 \_\_iter\_\_, 237  
 \_\_setitem\_\_, 237  
 \_data, 240  
 \_type, 240  
 add, 237  
 bind, 237  
 clear\_data, 237  
 contains\_operation, 237  
 get\_type, 238  
 has\_key, 238  
 has\_value, 238  
 is\_initialized, 238  
 items, 238  
 iteritems, 238  
 iterkeys, 238  
 itervalues, 238  
 keys, 238  
 length, 239  
 merge, 239  
 next, 239  
 reduce, 239  
 remove, 239  
 set\_operation, 239  
 set\_type, 239  
 values, 239  
**gen::baseclasses::BooleanLeaf**  
 \_\_init\_\_, 243  
 data, 244  
 get\_as\_text, 244  
 get\_value, 244  
 initialized, 244  
 parse, 244  
 set\_value, 244  
**gen::baseclasses::Decimal64Leaf**  
 \_\_init\_\_, 246  
 check\_range, 246  
 data, 247  
 dec\_range, 247

**Index**

fraction\_digits, 247  
 get\_as\_text, 246  
 get\_value, 246  
 initialized, 247  
 parse, 246  
 set\_value, 247  
 gen::baseclasses::FilterYang  
     \_\_init\_\_, 249  
     filter\_xml, 249  
     run, 249  
 gen::baseclasses::IntLeaf  
     \_\_init\_\_, 281  
     check\_range, 281  
     data, 281  
     get\_as\_text, 281  
     get\_value, 281  
     initialized, 281  
     int\_range, 281  
     parse, 281  
     set\_value, 281  
 gen::baseclasses::Leaf  
     \_\_eq\_\_, 283  
     \_\_init\_\_, 283  
     clear\_data, 283  
     data, 285  
     delete, 283  
     get\_as\_text, 283  
     get\_mandatory, 284  
     get\_units, 284  
     get\_value, 284  
     is\_initialized, 284  
     mandatory, 285  
     reduce, 284  
     set\_mandatory, 284  
     set\_units, 284  
     set\_value, 284  
     units, 285  
 gen::baseclasses::Leafref  
     \_\_init\_\_, 287  
     bind, 287  
     data, 287  
     get\_as\_text, 287  
     get\_target, 287  
     is\_initialized, 287  
     set\_value, 287  
     target, 287  
     unbind, 287  
 gen::baseclasses::ListedYang  
     \_\_init\_\_, 294  
     empty\_copy, 295  
     get\_key\_tags, 295  
     get\_parent, 295  
     get\_path, 295  
     keys, 295  
     reduce, 295  
 gen::baseclasses::StringLeaf  
     \_\_init\_\_, 308  
     data, 309  
 get\_as\_text, 309  
 get\_value, 309  
 parse, 309  
 set\_value, 309  
 gen::baseclasses::Yang  
     \_\_eq\_\_, 315  
     \_\_init\_\_, 315  
     \_\_str\_\_, 315  
     bind, 315  
     contains\_operation, 316  
     delete, 316  
     empty\_copy, 316  
     full\_copy, 316  
     get\_as\_text, 316  
     get\_operation, 316  
     get\_parent, 316  
     get\_path, 316  
     get\_rel\_path, 316  
     get\_tag, 317  
     is\_initialized, 317  
     merge, 317  
     operation, 318  
     parse, 317  
     reduce, 317  
     set\_operation, 317  
     set\_parent, 317  
     set\_referred, 317  
     set\_tag, 318  
     unset\_referred, 318  
     walk\_path, 318  
     xml, 318  
 gen::virtualizer3  
     \_\_copyright\_\_, 241  
 gen::virtualizer3::Flowentry  
     \_\_init\_\_, 251  
 gen::virtualizer3::FlowtableFlowtable  
     \_\_getitem\_\_, 253  
     \_\_init\_\_, 253  
     \_\_iter\_\_, 253  
     add, 253  
     flowentry, 253  
     remove, 253  
 gen::virtualizer3::GroupingFlowentry  
     \_\_init\_\_, 255  
     action, 256  
     match, 256  
     out, 256  
     port, 256  
     priority, 256  
     resources, 256  
 gen::virtualizer3::GroupingFlowtable  
     \_\_init\_\_, 258  
     flowtable, 258  
 gen::virtualizer3::GroupingId\_name  
     \_\_init\_\_, 259  
     id, 259  
     name, 259  
 gen::virtualizer3::GroupingId\_name\_type

**Index**

\_\_init\_\_, 261  
 type, 261  
 gen::virtualizer3::GroupingInfra\_node  
     \_\_init\_\_, 262  
     capabilities, 262  
     NF\_instances, 262  
 gen::virtualizer3::GroupingLink  
     \_\_init\_\_, 264  
     dst, 264  
     resources, 264  
     src, 265  
 gen::virtualizer3::GroupingLink\_resource  
     \_\_init\_\_, 266  
     bandwidth, 266  
     delay, 266  
 gen::virtualizer3::GroupingLinks  
     \_\_init\_\_, 268  
     links, 268  
 gen::virtualizer3::GroupingNode  
     \_\_init\_\_, 270  
     ports, 271  
     resources, 271  
 gen::virtualizer3::GroupingNodes  
     \_\_getitem\_\_, 272  
     \_\_init\_\_, 272  
     \_\_iter\_\_, 272  
     add, 272  
     node, 273  
     remove, 273  
 gen::virtualizer3::GroupingPort  
     \_\_init\_\_, 274  
     capability, 274  
     port\_type, 274  
     sap, 275  
 gen::virtualizer3::GroupingSoftware\_resource  
     \_\_init\_\_, 276  
     cpu, 276  
     mem, 276  
     storage, 276  
 gen::virtualizer3::Infra\_node  
     \_\_init\_\_, 277  
 gen::virtualizer3::Infra\_nodeCapabilities  
     \_\_init\_\_, 279  
     supported\_NFs, 279  
 gen::virtualizer3::Link  
     \_\_init\_\_, 289  
 gen::virtualizer3::Link\_resource  
     \_\_init\_\_, 291  
 gen::virtualizer3::LinksLinks  
     \_\_getitem\_\_, 292  
     \_\_init\_\_, 292  
     \_\_iter\_\_, 292  
     add, 292  
     link, 293  
     remove, 293  
 gen::virtualizer3::Node  
     \_\_init\_\_, 297  
 gen::virtualizer3::NodePorts  
     \_\_getitem\_\_, 299  
     \_\_init\_\_, 299  
     \_\_iter\_\_, 299  
     add, 299  
     port, 299  
     remove, 299  
 gen::virtualizer3::Nodes  
     \_\_init\_\_, 301  
 gen::virtualizer3::Port  
     \_\_init\_\_, 303  
 gen::virtualizer3::Software\_resource  
     \_\_init\_\_, 307  
 gen::virtualizer3::Virtualizer  
     \_\_init\_\_, 311  
     nodes, 311  
 gen::virtualizer3::VirtualizerNodes  
     \_\_getitem\_\_, 312  
     \_\_init\_\_, 312  
     \_\_iter\_\_, 312  
     add, 312  
     node, 313  
     remove, 313  
 get\_as\_text  
     gen::baseclasses::BooleanLeaf, 244  
     gen::baseclasses::Decimal64Leaf, 246  
     gen::baseclasses::IntLeaf, 281  
     gen::baseclasses::Leaf, 283  
     gen::baseclasses::Leafref, 287  
     gen::baseclasses::StringLeaf, 309  
     gen::baseclasses::Yang, 316  
 get\_key\_tags  
     gen::baseclasses::ListedYang, 295  
 get\_mandatory  
     gen::baseclasses::Leaf, 284  
 get\_operation  
     gen::baseclasses::Yang, 316  
 get\_parent  
     gen::baseclasses::ListedYang, 295  
     gen::baseclasses::Yang, 316  
 get\_path  
     gen::baseclasses::ListedYang, 295  
     gen::baseclasses::Yang, 316  
 get\_rel\_path  
     gen::baseclasses::Yang, 316  
 get\_tag  
     gen::baseclasses::Yang, 317  
 get\_target  
     gen::baseclasses::Leafref, 287  
 get\_type  
     gen::baseclasses, 238  
 get\_units  
     gen::baseclasses::Leaf, 284  
 get\_value  
     gen::baseclasses::BooleanLeaf, 244  
     gen::baseclasses::Decimal64Leaf, 246  
     gen::baseclasses::IntLeaf, 281  
     gen::baseclasses::Leaf, 284  
     gen::baseclasses::StringLeaf, 309

**Index**

has\_key  
     gen::baseclasses, 238

has\_value  
     gen::baseclasses, 238

id  
     gen::virtualizer3::GroupingId\_name, 259

initialized  
     gen::baseclasses::BooleanLeaf, 244  
     gen::baseclasses::Decimal64Leaf, 247  
     gen::baseclasses::IntLeaf, 281

int\_range  
     gen::baseclasses::IntLeaf, 281

is\_initialized  
     gen::baseclasses, 238  
     gen::baseclasses::Leaf, 284  
     gen::baseclasses::Leafref, 287  
     gen::baseclasses::Yang, 317

items  
     gen::baseclasses, 238

iteritems  
     gen::baseclasses, 238

iterkeys  
     gen::baseclasses, 238

itervalues  
     gen::baseclasses, 238

keys  
     gen::baseclasses, 238  
     gen::baseclasses::ListedYang, 295

length  
     gen::baseclasses, 239

link  
     gen::virtualizer3::LinksLinks, 293

links  
     gen::virtualizer3::GroupingLinks, 268

mandatory  
     gen::baseclasses::Leaf, 285

match  
     gen::virtualizer3::GroupingFlowentry, 256

mem  
     gen::virtualizer3::GroupingSoftware\_resource, 276

merge  
     gen::baseclasses, 239  
     gen::baseclasses::Yang, 317

NF\_instances  
     gen::virtualizer3::GroupingInfra\_node, 262

name  
     gen::virtualizer3::GroupingId\_name, 259

next  
     gen::baseclasses, 239

node  
     gen::virtualizer3::GroupingNodes, 273  
     gen::virtualizer3::VirtualizerNodes, 313

nodes  
     gen::virtualizer3::Virtualizer, 311

object, 301

operation  
     gen::baseclasses::Yang, 318

out  
     gen::virtualizer3::GroupingFlowentry, 256

parse  
     gen::baseclasses::BooleanLeaf, 244  
     gen::baseclasses::Decimal64Leaf, 246  
     gen::baseclasses::IntLeaf, 281  
     gen::baseclasses::StringLeaf, 309  
     gen::baseclasses::Yang, 317

port  
     gen::virtualizer3::GroupingFlowentry, 256  
     gen::virtualizer3::NodePorts, 299

port\_type  
     gen::virtualizer3::GroupingPort, 274

ports  
     gen::virtualizer3::GroupingNode, 271

priority  
     gen::virtualizer3::GroupingFlowentry, 256

reduce  
     gen::baseclasses, 239  
     gen::baseclasses::Leaf, 284  
     gen::baseclasses::ListedYang, 295  
     gen::baseclasses::Yang, 317

remove  
     gen::baseclasses, 239  
     gen::virtualizer3::FlowtableFlowtable, 253  
     gen::virtualizer3::GroupingNodes, 273  
     gen::virtualizer3::LinksLinks, 293  
     gen::virtualizer3::NodePorts, 299  
     gen::virtualizer3::VirtualizerNodes, 313

resources  
     gen::virtualizer3::GroupingFlowentry, 256  
     gen::virtualizer3::GroupingLink, 264  
     gen::virtualizer3::GroupingNode, 271

run  
     gen::baseclasses::FilterYang, 249

sap  
     gen::virtualizer3::GroupingPort, 275

set\_mandatory  
     gen::baseclasses::Leaf, 284

set\_operation  
     gen::baseclasses, 239  
     gen::baseclasses::Yang, 317

set\_parent  
     gen::baseclasses::Yang, 317

set\_referred  
     gen::baseclasses::Yang, 317

set\_tag  
     gen::baseclasses::Yang, 318

set\_type  
     gen::baseclasses, 239

set\_units  
     gen::baseclasses::Leaf, 284

set\_value

gen::baseclasses::BooleanLeaf, 244  
gen::baseclasses::Decimal64Leaf, 247  
gen::baseclasses::IntLeaf, 281  
gen::baseclasses::Leaf, 284  
gen::baseclasses::Leafref, 287  
gen::baseclasses::StringLeaf, 309  
src  
    gen::virtualizer3::GroupingLink, 265  
storage  
    gen::virtualizer3::GroupingSoftware\_resource, 276  
supported\_NFs  
    gen::virtualizer3::Infra\_nodeCapabilities, 279  
  
target  
    gen::baseclasses::Leafref, 287  
type  
    gen::virtualizer3::GroupingId\_name\_type, 261  
  
unbind  
    gen::baseclasses::Leafref, 287  
units  
    gen::baseclasses::Leaf, 285  
unset\_referred  
    gen::baseclasses::Yang, 318  
  
values  
    gen::baseclasses, 239  
virtualizer3.py, 321  
  
walk\_path  
    gen::baseclasses::Yang, 318  
  
xml  
    gen::baseclasses::Yang, 318