NFV API Conformance Test Suite

Overview

Presented by: **Giacomo Bernini**
Elian Kraja
Nextworks

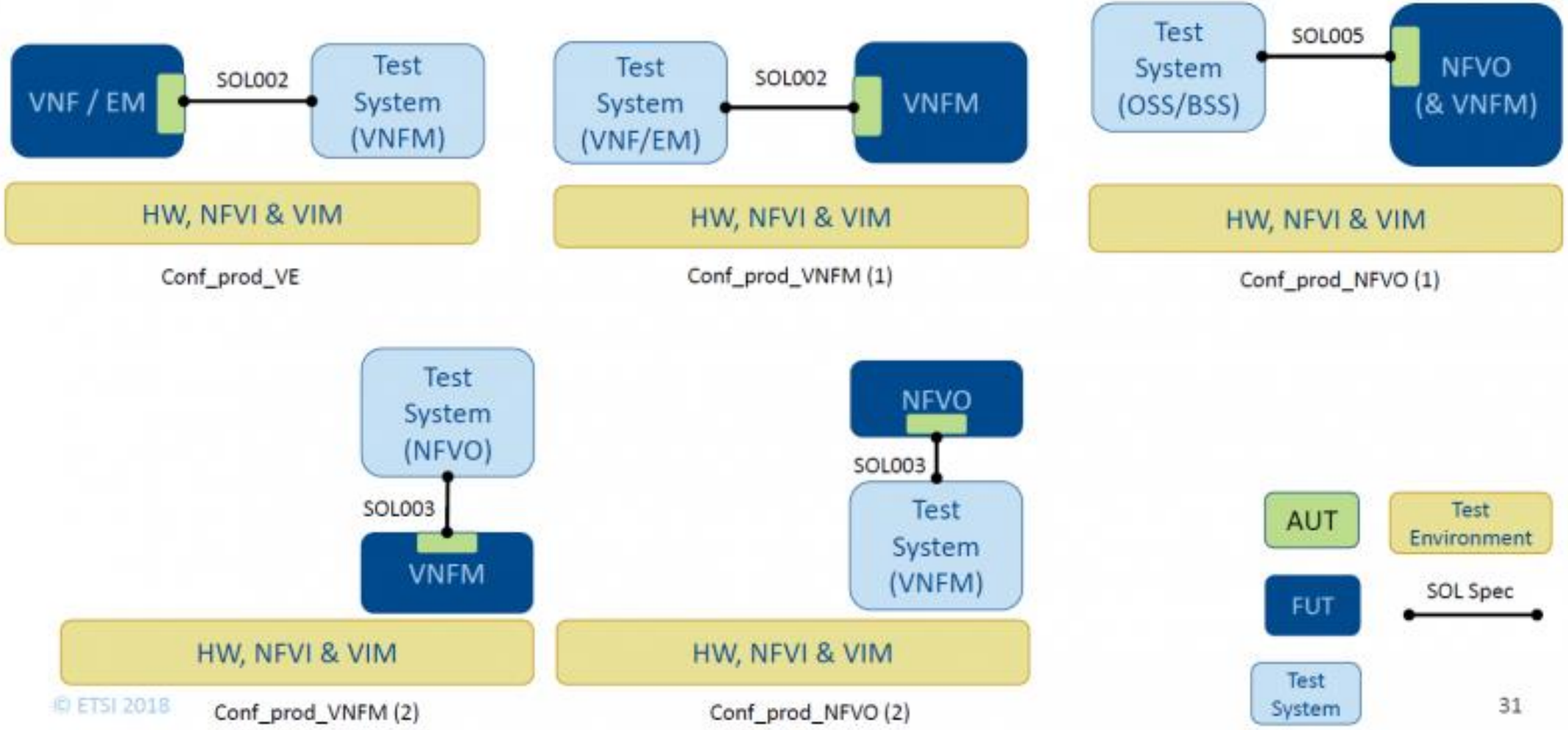
For: **4th ETSI NFV Plugtests**

05.06.2019

NFV API Conformance Test Suite: scope

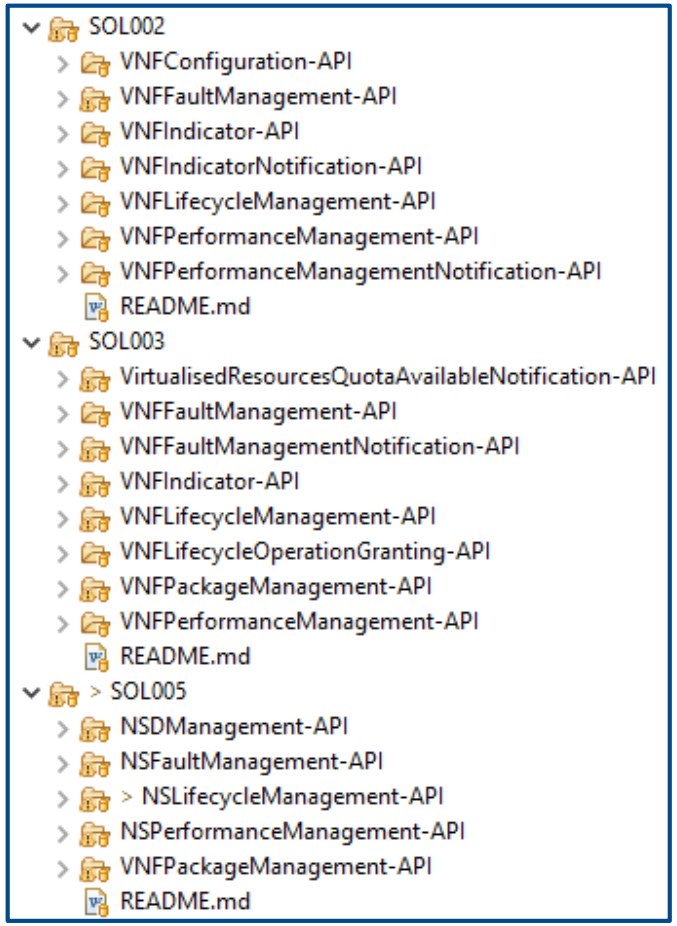
- Robot Framework API conformance test suite for ETSI NFV SOL specs
  - SOL002, SOL003, SOL005 v2.4.1
  - Part of the ETSI NFV TST010 work item
- Main verification criteria and aspects
  - API syntax check on FUT response
  - Explicit FUT status/postcondition check by defined-API
    - Query/Notification → in the same reference point
- Two main categories of Robot tests have been developed
  - Tests of individual API resource endpoints
  - Tests of NS and VNF lifecycle management workflows

NFV API Conformance Test Suite: test configurations

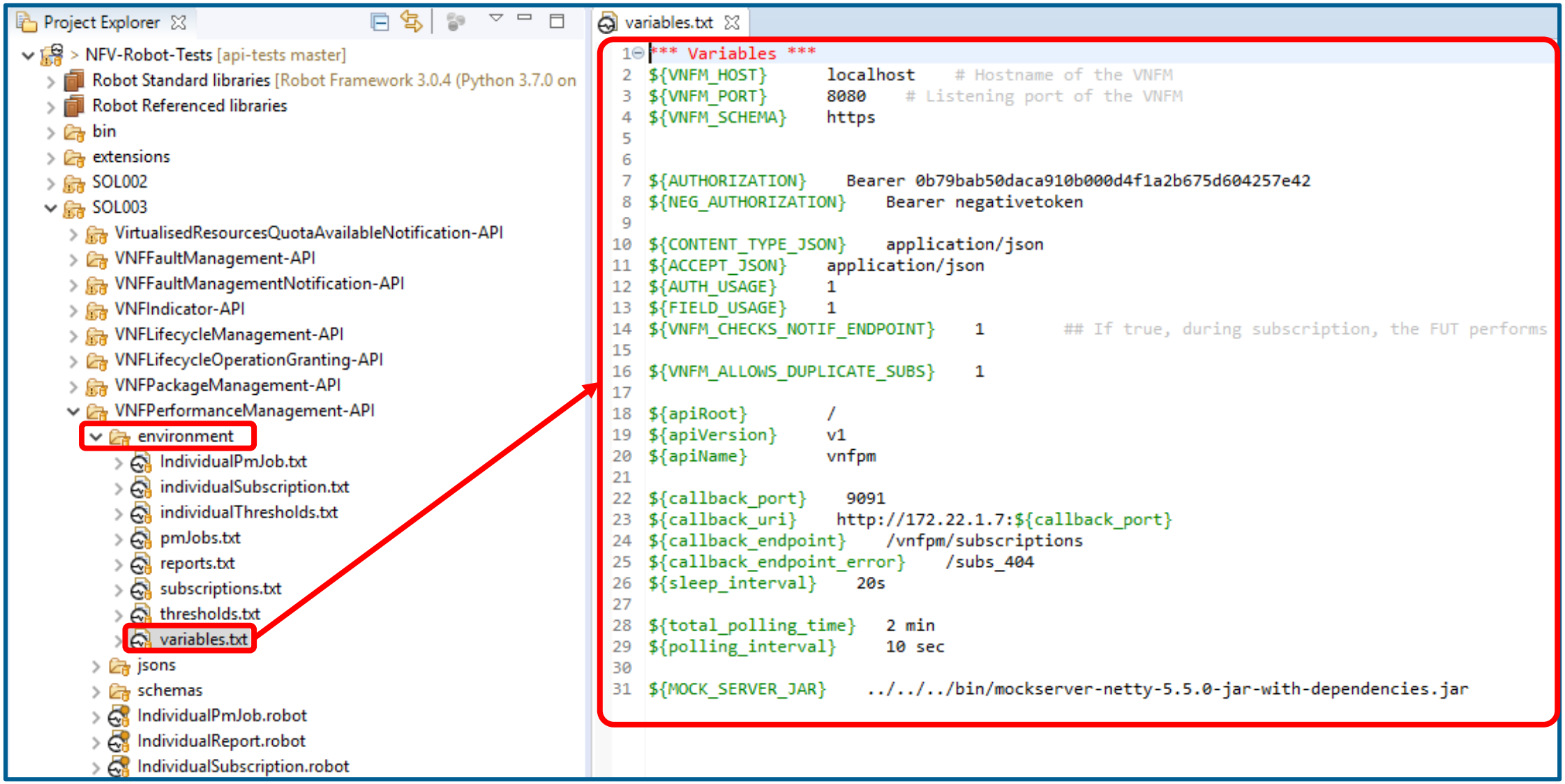


NFV API Conformance Test Suite: code repository

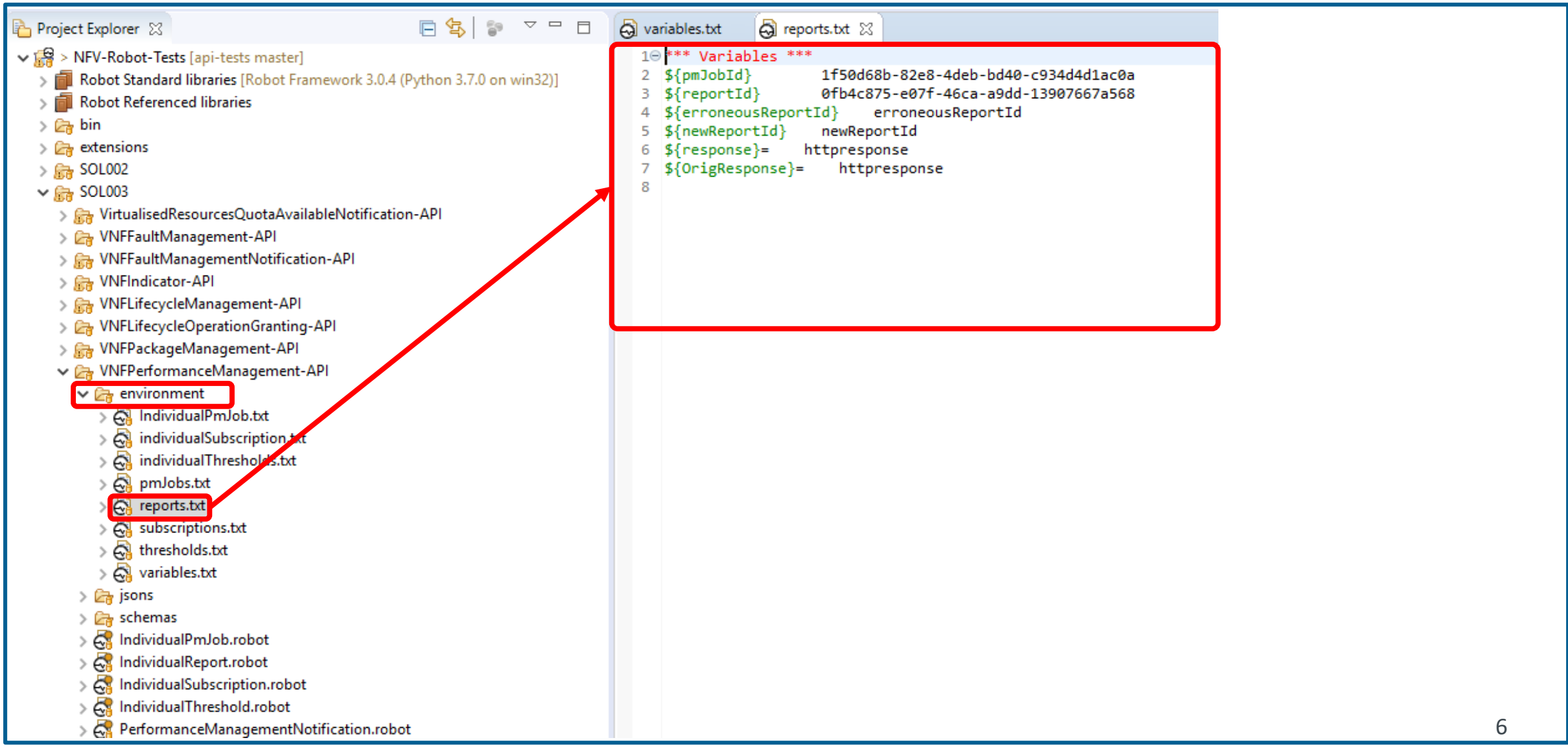
- Robot test code is available and maintained in the public Git repo hosted by ETSI Forge
  - https://forge.etsi.org/rep/nfv/api-tests
- The Robot test code is organized following the SOL specs structure
  - One folder for each SOL spec
    - One sub-folder for each interface
      - One Robot test suite file for each API resource endpoint
      - + “environment” folder
      - + “jsons” folder
      - + “schemas” folder



NFV API Conformance Test Suite: code structure



NFV API Conformance Test Suite: code structure



NFV API Conformance Test Suite: code structure

Project Explorer

- NFV-Robot-Tests [api-tests master]
  - Robot Standard libraries [Robot Framework 3.0.4 (Python 3.7.0 on win32)]
  - Robot Referenced libraries
  - bin
  - extensions
  - SOL002
    - VNFConfiguration-API
    - VNFFaultManagement-API
    - VNFIndicator-API
    - VNFIndicatorNotification-API
    - VNFLifecycleManagement-API
    - VNFPerformanceManagement-API
    - VNFPerformanceManagementNotification-API
    - README.md
  - SOL003
    - VirtualisedResourcesQuotaAvailableNotification-API
    - VNFFaultManagement-API
    - VNFFaultManagementNotification-API
    - VNFIndicator-API
    - VNFLifecycleManagement-API
    - VNFLifecycleOperationGranting-API
    - VNFPackageManagement-API
    - VNFPerformanceManagement-API
      - environment
      - jsons
        - CreatePmJobRequest.json
        - CreateThresholdRequest.json
        - PerformanceInformationAvailableNotification.json
        - subscriptions.json
        - ThresholdCrossedNotification.json
      - schemas
      - IndividualPmJob.robot
      - IndividualReport.robot
      - IndividualSubscription.robot
      - IndividualThreshold.robot

variables.txt | reports.txt | subscriptions.json

```
1 {
2     "callbackUri": "http://172.22.1.7:9091/vnfpm/subscriptions",
3     "filter": {
4         "notificationTypes": ["ThresholdCrossedNotification"]
5     }
6 }
```

NFV API Conformance Test Suite: code structure

Project Explorer

- NFV-Robot-Tests [api-tests master]
  - Robot Standard libraries [Robot Framework 3.0.4 (Python 3.7.0 on win32)]
  - Robot Referenced libraries
  - bin
  - extensions
  - SOL002
    - VNFConfiguration-API
    - VNFFaultManagement-API
    - VNFIndicator-API
    - VNFIndicatorNotification-API
    - VNFLifecycleManagement-API
    - VNFPerformanceManagement-API
    - VNFPerformanceManagementNotification-API
    - README.md
  - SOL003
    - VirtualisedResourcesQuotaAvailableNotification-API
    - VNFFaultManagement-API
    - VNFFaultManagementNotification-API
    - VNFIndicator-API
    - VNFLifecycleManagement-API
    - VNFLifecycleOperationGranting-API
    - VNFPackageManagement-API
    - VNFPerformanceManagement-API
      - environment
      - jsons
      - schemas
        - criteria.schema.json
        - links.schema.json
        - PerformanceInformationAvailableNotification.schema.json
        - PerformanceReport.schema.json
        - PmJob.schema.json
        - PmJobs.schema.json
        - PmSubscription.schema.json
        - PmSubscriptions.schema.json
        - ProblemDetails.schema.json
        - reports.schema.json
        - Subscriptions.schema.json
        - Threshold.schema.json
        - ThresholdCrossedNotification.schema.json
        - Thresholds.schema.json
      - IndividualPmJob.robot
      - IndividualReport.robot

variables.txt | reports.txt | subscriptions.json | reports.schema.json

```
1  {
2      "description": "Information about available reports collected by this PM job.\n",
3      "type": "object",
4      "required": ["href", "readyTime"],
5      "properties": {
6          "href": {
7              "description": "The Uri where the report can be obtained.\n",
8              "type": "string",
9              "format": "url"
10         },
11         "readyTime": {
12             "description": "Date-time stamp.  Representation: String formatted according to IETF RFC 3339.\n",
13             "type": "string",
14             "format": "date-time"
15         },
16         "expiryTime": {
17             "description": "Date-time stamp.  Representation: String formatted according to IETF RFC 3339.\n",
18             "type": "string",
19             "format": "date-time"
20         },
21         "fileSize": {
22             "description": "The size of the report file in bytes, if known.\n",
23             "type": "integer"
24         }
25     }
26 }
```

NFV API Conformance Test Suite: code structure

Robot Tests are classified into two categories

- Individual endpoint tests
  - one Robot file (TEST SUITE) for each resource endpoint
    - Each include a set of Robot TEST CASES related to that resource endpoint
  - The name of the Robot file <u>IS</u> the name of the resource endpoint
    - e.g. *PmJobs.robot*
- LCM Workflow tests
  - One Robot file for each LCM workflow
  - The name of the Robot file <u>IS</u> a self-explanative name of the workflow
    - e.g. *InstantiateVNFTaskWorkflow.robot*

NFV API Conformance Test Suite: code highlight

SOL002 – VNF Configuration TD

```
*** Test Cases ***
Set new VNF Configuration
    [Documentation]    Test ID: 6.3.1.1.1
    ...    Test title: Set a new VNF Configuration
    ...    Test objective: The objective is to test the creation of a new VNF configuration and
    ...                    validation of the returned configuration data structure
    ...    Pre-conditions: A VNF instance is instantiated
    ...    Reference:  section 9.4.2.3.4 - SOL002 v2.4.1
    ...    Config ID: Config_prod_VE
    ...    Applicability: The VNF supports the generation of HTTP Etag opaque identifiers
    ...    Post-Conditions: The configuration is successfully set in the VNF and it matches
    Send VNF configuration
    Check HTTP Response Status Code Is    200
    Check HTTP Response Header Contains    ETag
    Check HTTP Response Body Json Schema Is   vnfConfigModifications
    Check Postcondition VNF Is Configured
```

```
Check HTTP Response Status Code Is
    [Arguments]    ${expected_status}
    ${status}=    Convert To Integer    ${expected_status}
    Should Be Equal    ${response['status']}    ${status}
    Log    Status code validated
```

```
Check HTTP Response Header Contains
    [Arguments]    ${CONTENT_TYPE}
    Log    ${response['headers']}
    Should Contain    ${response['headers']}    ${CONTENT_TYPE}
    Log    Header is present
```

low-level Robot code

```
Send VNF configuration
    log    Trying to perform a PATCH. This method modifies the configuration
    Set Headers  {"Accept":"${ACCEPT}"}
    Set Headers  {"Content-Type": "${CONTENT_TYPE}"}
    Run Keyword If    ${AUTH_USAGE} == 1    Set Headers    {"Authorization":"${AUTHORIZATION}"}
    ${body}=    Get File    jsons/vnfConfigModifications.json
    Patch    ${apiRoot}/${apiName}/${apiVersion}/configuration    ${body}
    Set Suite Variable    &{etag}    ${response['headers']['ETag']}
    ${output}=    Output    response
    Set Suite Variable    ${response}    ${output}
```

```
Check HTTP Response Body Json Schema Is
    [Arguments]    ${input}
    Should Contain    ${response['headers']['Content-Type']}
    ${schema} =    Catenate    ${input}    .schema.json
    Validate Json    ${schema}    ${response['body']}
    Log    Json Schema Validation OK
```

```
Check Postcondition VNF Is Configured
    Log    Check Postcondition for VNF Configuration
    Get VNF configuration
    ${input_file}=    Get File    jsons/vnfConfigModifications.json
    ${input}=    evaluate    json.loads('''${input_file}''')    json
    Should Be Equal  ${response['body']}    ${input}
```

How to run the tests: pre-requisites

- Software pre-requisites
  - Python >=3.0 + Pip
  - Robot Framework version >=3.0
  - Python libraries for Robot Framework
    - https://forge.etsi.org/gitlab/nfv/api-tests/wikis/NFV-API-Conformance-Test-Specification#dependencies-and-preconditions
- Optionally, for easy browsing of the code the installation of Eclipse and its Robot plugin (RED) is strongly suggested
  - https://forge.etsi.org/gitlab/nfv/api-tests/wikis/NFV-API-Conformance-Test-Specification#robot-framework-ide

How to run the tests: preparation

1. Identify the set of Robot test suites and test cases applicable for the given FUT
   - The code structure should help in selecting which tests can be run
2. Configure the Robot Test System according to the local test environment
   a) Set global attributes/parameters in ‘variables.txt’
   b) Set per-resource-endpoint attributes/parameters in the ‘environment’ folder
   c) Where required, set the request bodies in the ‘jsons’ folder

How to run the tests: execution

- For each applicable Robot test suite (i.e. resource endpoint or LCM workflow Robot file), run:

```
$ robot -T -d <path_to_output_dir> <path_to_robot_file>
```

  - Additional *- -task <test_case_name>* option can be used to run an individual test case within a Robot test suite file
  - Additional *- - include <tag-a, tag-b,..>* or *- - exclude <tag-a, tag-b,..>* option can be used to include/exclude test cases based on tags
- Check results and logs in the '*<path_to_output_dir>*' folder

Robot CLI user guide

- *http://robotframework.org/robotframework/latest/RobotFrameworkUserGuide.html#executing-test-cases*

How to run the tests: results analysis

- Robot generates three outputs as result of each test execution
  - **report.html** : summary execution info & statistics
  - **log.html** : detailed per test case log trace
    - main source for debugging and analyzing any test failure
  - **output.xml** : detailed per test case output for post-processing

How to report issues on the Robot Test Suite

- An issue tracker is available on the ETSI Forge git repo to report bugs/inconsistencies found in the Robot code:
  - https://forge.etsi.org/gitlab/nfv/api-tests/issues

Thank you

Giacomo Bernini
Elian Kraja

Nextworks