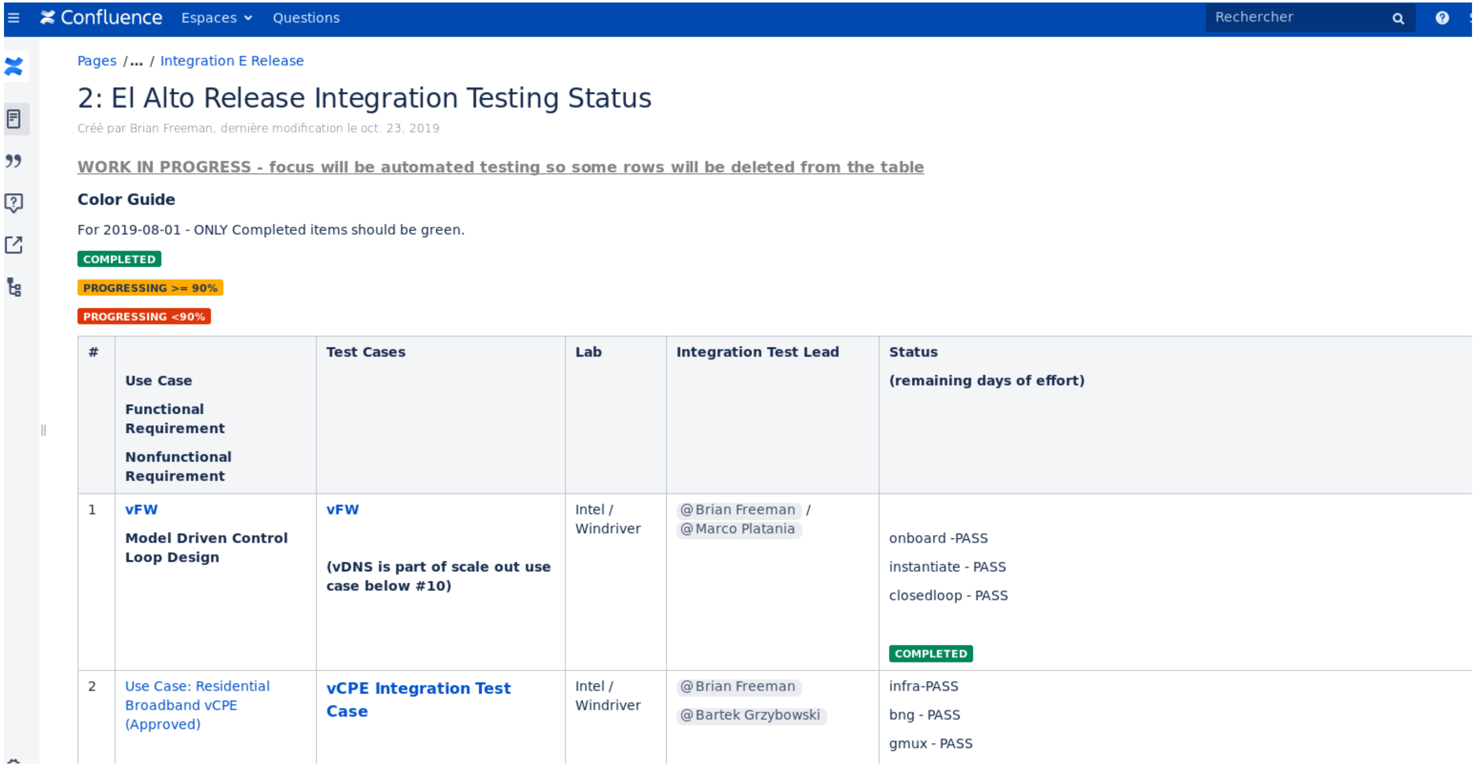
Integration meeting

Proposal to enhance The Integration Gating for Frankfurt

Today

- Main page for integration is https://wiki.onap.org/display/DW/2%3A+El+Alto+Release+Integration+Testing+Status



2: El Alto Release Integration Testing Status

Créé par Brian Freeman, dernière modification le oct. 23, 2019

WORK IN PROGRESS - focus will be automated testing so some rows will be deleted from the table

Color Guide

For 2019-08-01 - ONLY Completed items should be green.

COMPLETED

PROGRESSING >= 90%

PROGRESSING <90%

#	Use Case Functional Requirement Nonfunctional Requirement	Test Cases	Lab	Integration Test Lead	Status (remaining days of effort)
1	vFW Model Driven Control Loop Design	vFW (vDNS is part of scale out use case below #10)	Intel / Windriver	@Brian Freeman / @Marco Platania	onboard -PASS instantiate - PASS closedloop - PASS COMPLETED
2	Use Case: Residential Broadband vCPE (Approved)	vCPE Integration Test Case	Intel / Windriver	@Brian Freeman @Bartek Grzybowski	infra-PASS bng - PASS gmux - PASS

What was new in El Alto

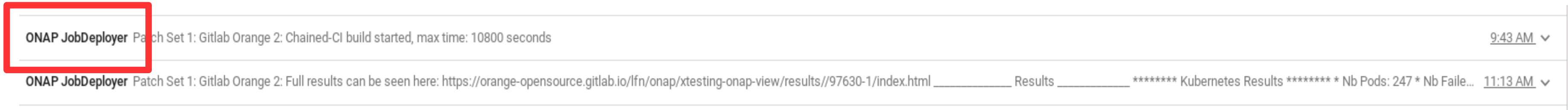
- From 1 to 3 labs
  - Windriver Lab (integration for the use cases, long duration tests)
  - Orange lab (Daily master/elalto, Gating, Openlab (last stable))
  - Ericsson lab (Daily master/dublin)
- A first draft of integration portal : http://testresults.opnfv.org/onap-integration/

ONAP Testing group reporting

Src code of the portal https://gitlab.com/Orange-OpenSource/lfn/onap/integration-view

What was new in El Alto

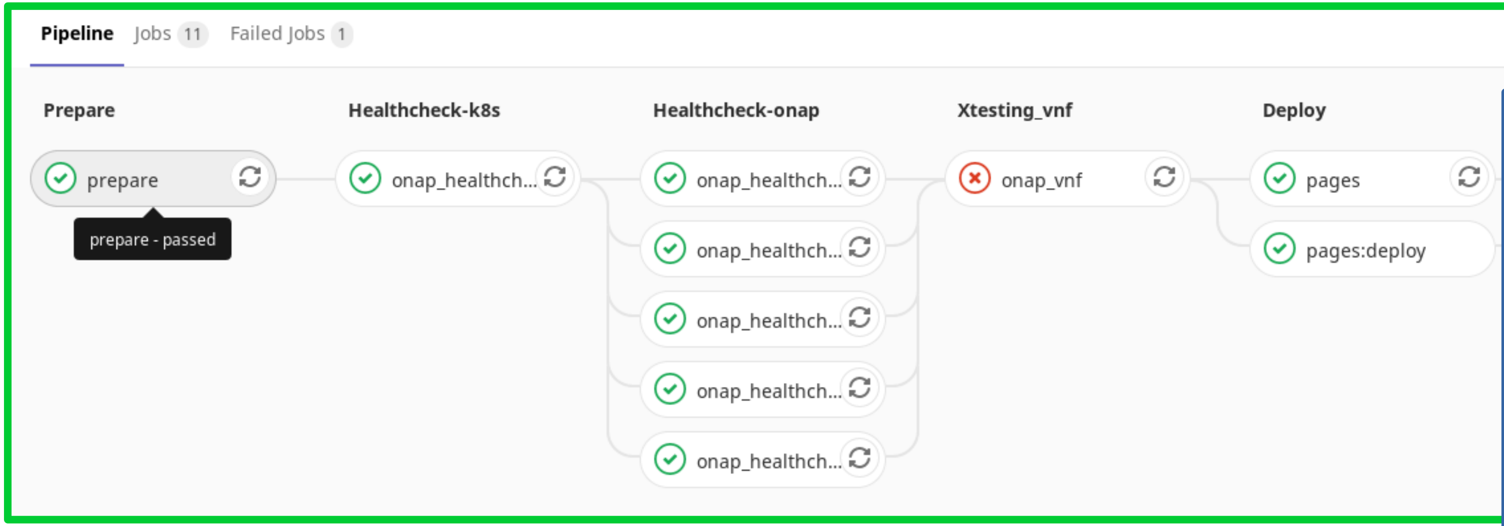
- Gating : deployement + test of a full ONAP on patch submission

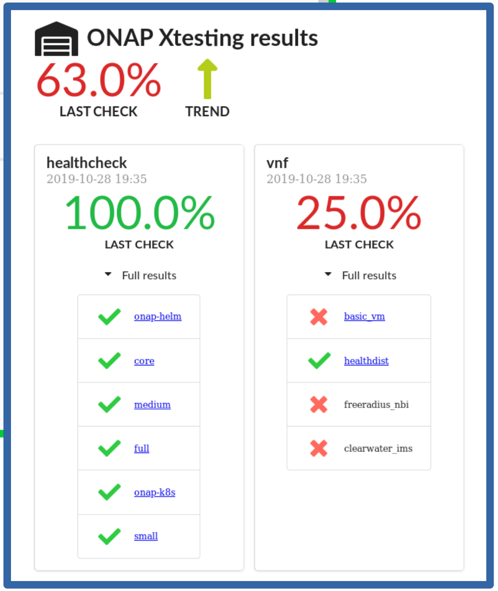


ONAP JobDeployer Patch Set 1: Gitlab Orange 2: Chained-CI build started, max time: 10800 seconds 9:43 AM

ONAP JobDeployer Patch Set 1: Gitlab Orange 2: Full results can be seen here: https://orange-opensource.gitlab.io/lfn/onap/xtesting-onap-view/results//97630-1/index.html ___________ Results ___________ ******** Kubernetes Results ******** * Nb Pods: 247 * Nb Faile... 11:13 AM

- Gating + Orange Daily Chains are based on the following « integration » gate





ONAP Xtesting results

63.0% LAST CHECK ↑ TREND

healthcheck
2019-10-28 19:35
100.0%
LAST CHECK

▾ Full results

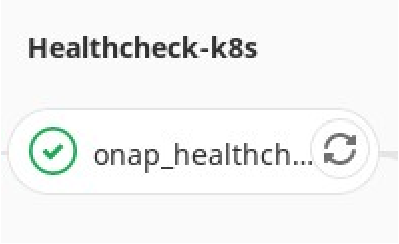
✓	onap-helm
✓	core
✓	medium
✓	full
✓	onap-k8s
✓	small

vnf
2019-10-28 19:35
25.0%
LAST CHECK

▾ Full results

✗	basic_vm
✓	healthdist
✗	freeradius_nbi
✗	clearwater_ims

healthcheck-k8s

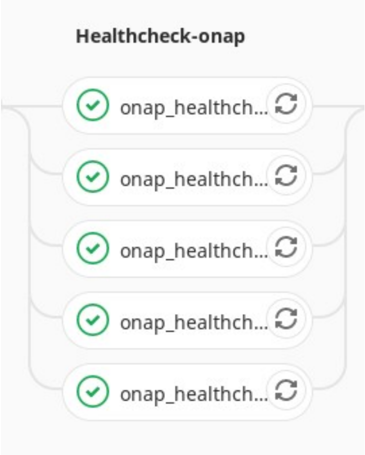


Healthcheck-k8s

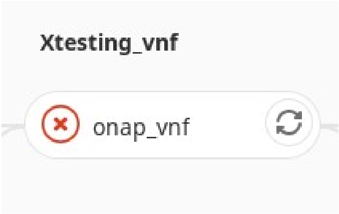
- https://github.com/Orange-OpenSource/xtesting-onap-k8s
- Tests aiming to check that ONAP is OK from a k8s perspective
- 2 tests
  - ONAP-k8s : check the pods/deployments/cm/events and print the log + the describe of the faulty pods
  - ONAP-helm : check the helm charts status

healthcheck-ONAP

- https://github.com/Orange-OpenSource/xtesting-onap-robot
- Integration of the canonical Robot healthcheck tests in Xtesting
- Used in Orange and E// CI chains
- 5 tests (run in //)
  - Core
  - Small
  - Medium
  - Full
  - Healthdist



healthcheck-ONAP



- https://gitlab.com/Orange-OpenSource/lfn/onap/onap-tests
- Onap-tests (ONAP python SDK) integration in Xtesting
- 3 tests
  - basic_vm : onboarding/distribution/instantiation/check with OpenStack Client of an Ubuntu VM
  - freeradius_nbi : ~ basic VM but instantiation is done through NBI
  - clearwater_ims : deployment of a full vIMS (reuse of existing SIP signaling testsuites to be considered)
- Some resources may be needed prior to the tests (networks, …) they are built with openstacksdk
- Resources are cleaned at the end of the tests

Status

- Regarding the number of use cases declared in the Integration Testing Status page, almost no « use cases » is really fully automated today
- For each new release we need to recheck on the lab to discover possible regressions, it is costly in term of time and resources
- the early feedback to the developpers ensure more stability (cf gating), it should be generalize through the feedback from the use cases
- Today the daily chains are running tests that are not referenced in the use case dashboards, not the ones from the wiki tables.
- **The status is only declarative in a wiki page, there is no way to « verify » it programatically.**

Proposition for Frankfurt

- The ambitious is to **bring as many use cases as possible to an « integration » Gate run in the CI daily chains of different labs**
- This « **integration gate** » shall be **runnable** towards **any ONAP system** (not only the labs involved in integration) – it excludes specific/proprietary use cases
- This « integration gate » shall be **independant from the CI system** (jenkins/gitlab ci/...)
- This « integration gate » shall harmonize the way to launch the tests leveraging **xtesting** framework
- The « integration gate » shall be run idnependantly from the milestones (master rolling release)
- Ideally any Master branch with a full integration gate could be released as a version
- All the **results** from **integration labs** shall be systematically **pushed into a central DB** for further processing
- For Frankfurt it sounds hard to convert all the use cases but we can start...and migrate from progressively from the wiki table to a consolidated dashboard based on the results of « integration gates »

Some words on xtesting

http://testresults.opnfv.org/functest/stockholm/

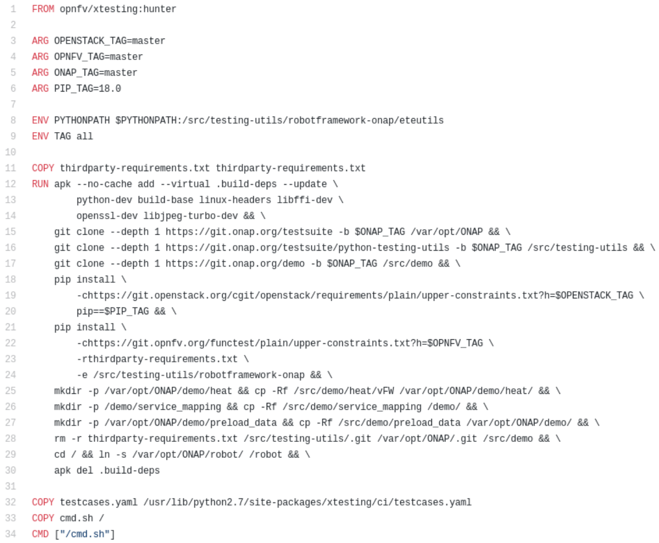
https://xtesting.readthedocs.io/en/latest/

http://testresults.opnfv.org/functest/ons2019/

Xtesting in motion

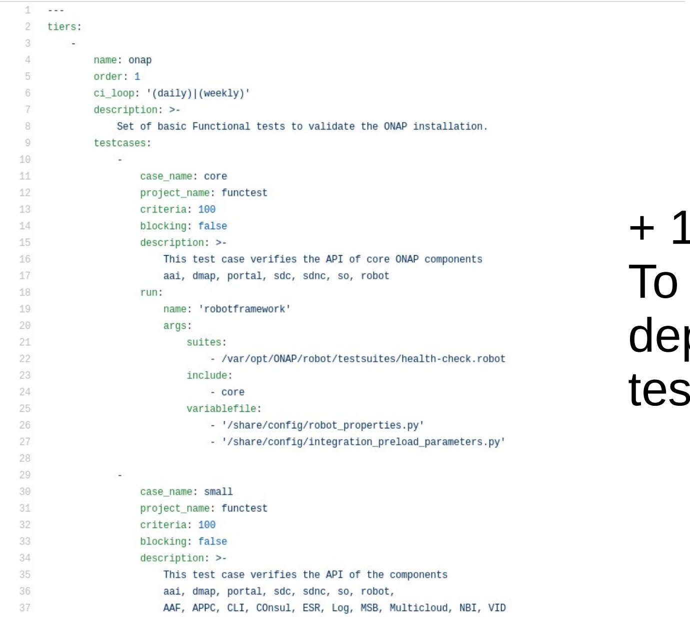
The canonical robot healthcheck tests

1 Docker file to build the exec env for the robot based tests



```
FROM opnfv/xtesting:hunter

ARG OPENSTACK_TAG=master
ARG OPNFV_TAG=master
ARG ONAP_TAG=master
ARG PIP_TAG=18.0

ENV PYTHONPATH $PYTHONPATH:/src/testing-utils/robotframework-onap/eteutils
ENV TAG all

COPY thirdparty-requirements.txt thirdparty-requirements.txt
RUN apk --no-cache add --virtual .build-deps --update \
        python-dev build-base linux-headers libffi-dev \
        openssl-dev libjpeg-turbo-dev && \
    git clone --depth 1 https://git.onap.org/testsuite -b $ONAP_TAG /var/opt/ONAP && \
    git clone --depth 1 https://git.onap.org/testsuite/python-testing-utils -b $ONAP_TAG /src/testing-utils && \
    git clone --depth 1 https://git.onap.org/demo -b $ONAP_TAG /src/demo && \
    pip install \
        -chttps://git.openstack.org/cgit/openstack/requirements/plain/upper-constraints.txt?h=$OPENSTACK_TAG \
        pip==$PIP_TAG && \
    pip install \
        -chttps://git.opnfv.org/functest/plain/upper-constraints.txt?h=$OPNFV_TAG \
        -rthirdparty-requirements.txt \
        -e /src/testing-utils/robotframework-onap && \
    mkdir -p /var/opt/ONAP/demo/heat && cp -Rf /src/demo/heat/vFW /var/opt/ONAP/demo/heat/ && \
    mkdir -p /demo/service_mapping && cp -Rf /src/demo/service_mapping /demo/ && \
    mkdir -p /var/opt/ONAP/demo/preload_data && cp -Rf /src/demo/preload_data /var/opt/ONAP/demo/ && \
    rm -r thirdparty-requirements.txt /src/testing-utils/.git /var/opt/ONAP/.git /src/demo && \
    cd / && ln -s /var/opt/ONAP/robot/ /robot && \
    apk del .build-deps

COPY testcases.yaml /usr/lib/python2.7/site-packages/xtesting/ci/testcases.yaml
COPY cmd.sh /
CMD ["/cmd.sh"]
```

+ 1 testcases.yaml to reference the test cases



```
---
tiers:
    -
        name: onap
        order: 1
        ci_loop: '(daily)|(weekly)'
        description: >-
            Set of basic Functional tests to validate the ONAP installation.
        testcases:
            -
                case_name: core
                project_name: functest
                criteria: 100
                blocking: false
                description: >-
                    This test case verifies the API of core ONAP components
                    aai, dmap, portal, sdc, sdnc, so, robot
                run:
                    name: 'robotframework'
                    args:
                        suites:
                            - /var/opt/ONAP/robot/testsuites/health-check.robot
                        include:
                            - core
                        variablefile:
                            - '/share/config/robot_properties.py'
                            - '/share/config/integration_preload_parameters.py'

            -
                case_name: small
                project_name: functest
                criteria: 100
                blocking: false
                description: >-
                    This test case verifies the API of the components
                    aai, dmap, portal, sdc, sdnc, so, robot,
                    AAF, APPC, CLI, COnsul, ESR, Log, MSB, Multicloud, NBI, VID
```

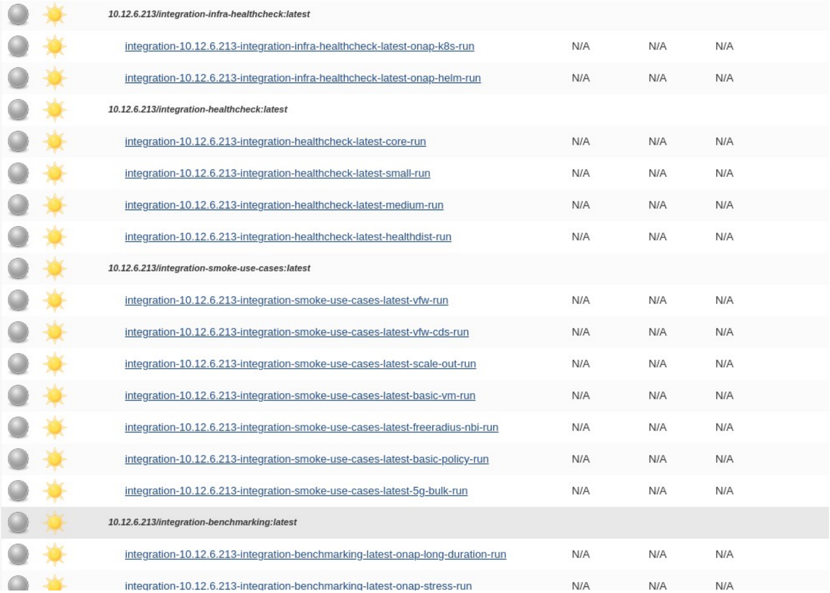
+ 1 requirements.txt
To manage the dependencies related to the test suites

Xtesting in motion

- Xtesting is an integration framework – it does not deal with the tests and will not create tests for you.
- Xtesting artifacts are dockers runnable towards any SUT on any CI systems. So it is an overhead but small compared to the benefits. It is a way for the integration team to validate the automation of a use case (xtestingization not to be done by the test developers)
- For labs that have no CI system, xtesting offer an ansible role published on Galaxy to build a jenkins gate for you

Integration gate protoype

http://10.12.6.213:8080/job/integration-latest-daily/



●	☀	*10.12.6.213/integration-infra-healthcheck:latest*			
●	☀	integration-10.12.6.213-integration-infra-healthcheck-latest-onap-k8s-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-infra-healthcheck-latest-onap-helm-run	N/A	N/A	N/A
●	☀	*10.12.6.213/integration-healthcheck:latest*			
●	☀	integration-10.12.6.213-integration-healthcheck-latest-core-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-healthcheck-latest-small-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-healthcheck-latest-medium-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-healthcheck-latest-healthdist-run	N/A	N/A	N/A
●	☀	*10.12.6.213/integration-smoke-use-cases:latest*			
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-vfw-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-vfw-cds-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-scale-out-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-basic-vm-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-freeradius-nbi-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-basic-policy-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-smoke-use-cases-latest-5g-bulk-run	N/A	N/A	N/A
●	☀	*10.12.6.213/integration-benchmarking:latest*			
●	☀	integration-10.12.6.213-integration-benchmarking-latest-onap-long-duration-run	N/A	N/A	N/A
●	☀	integration-10.12.6.213-integration-benchmarking-latest-onap-stress-run	N/A	N/A	N/A

Integration gate protoype what we already have

- An integration framework => xtesting
- A common test result DB/API => http://testresults.opnfv.org/onap/api/v1/pods
- Some dockers (even if some refactoring is needed)
  - Integration-infra-k8s : ready to use but need to be re-insourced in ONAP (currently on github)
  - xtesting-onap-robot to be renamed and moved to ONAP (on github)
  - some part for the VNFs to be modified and moved to ONAP

Integration gate protoype TODO list

- Define the test categories, create an integration project in the test DB and define the test cases http://testresults.opnfv.org/onap/api/v1/projects/integration/cases
  - For the prototype, I considered the following categories
    - Infra-healthcheck
    - Healthcheck
    - Smoke-use-cases (use cases need for the release validation)
    - Candidate-use-cases (for new use cases)
    - Security
    - Benchmarking
- Create a new repo integration/xtesting to manage to dockers needed for the integration gate
- Integrate the test cases...
- Deploy the Gates
  - On Orange labs we will complete the current gitlab-ci pipeline
  - The jenkins generated gate could be reused in E/// lab and/or in windriver lab

Questions