ARCHIVER

ARCHIVING AND PRESERVATION FOR RESEARCH ENVIRONMENTS

Cloud Test Suite Overview

Ignacio Peluaga Lozada (IT-DI-EFP)

ARCHIVER - Archiving and Preservation for Research Environments project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824516.

Outline

- Description of the cloud test & validation suite
  - as a standalone framework
- Integrated in a testing process
  - ARCHIVER project

Motivations

- Modern cloud services moving beyond the traditional categories of IaaS, PaaS and SaaS
  - Benchmarking and validation starts at infrastructure (CPU, GPU, FPGAs, Object Stores, Network….) going up the stack to the software layer
  - Commodity services? Maybe, but they need to be validated to multidisciplinary research use cases (across all the stacks)

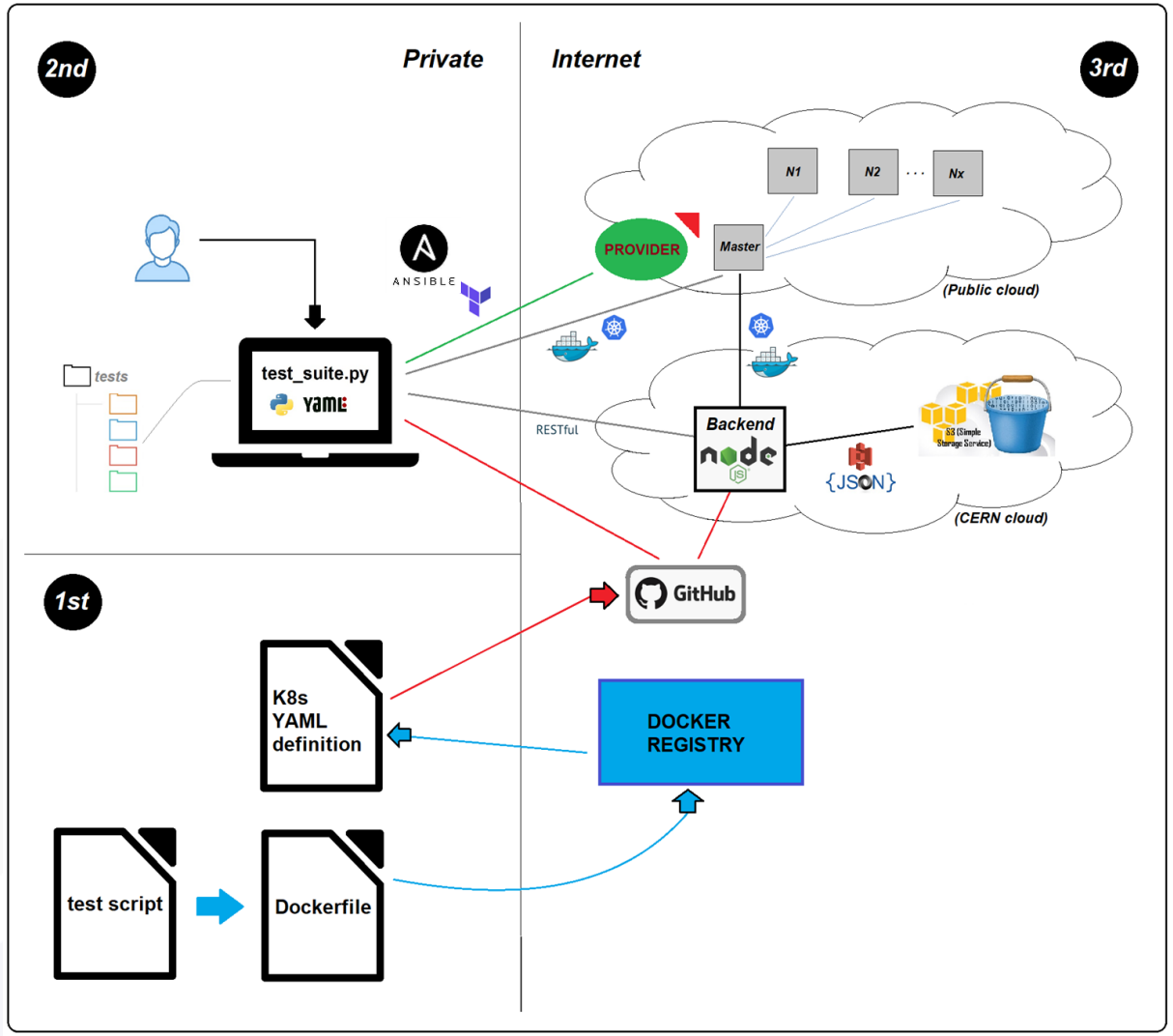
- Need a tool to automate deployment, benchmark and validate cloud services
  - Flexibility to onboard commercial clouds and satisfy diverse research workloads
  - Ease cloud services comparison
  - Use of open standards and tools; vendor independent
  - Validate exit strategies between providers
  - Foster wider adoption of cloud services by the research community

Context

- 2018 - Concrete application of lessons learned and feedback gathered in HNSciCloud
- 2019 - First version of the tool used in the OCRE project
- 2020 - Serving as a base for the ARCHIVER project test suite, expanding the tests catalog
- To be adopted in the European Open Science Cloud (EOSC)

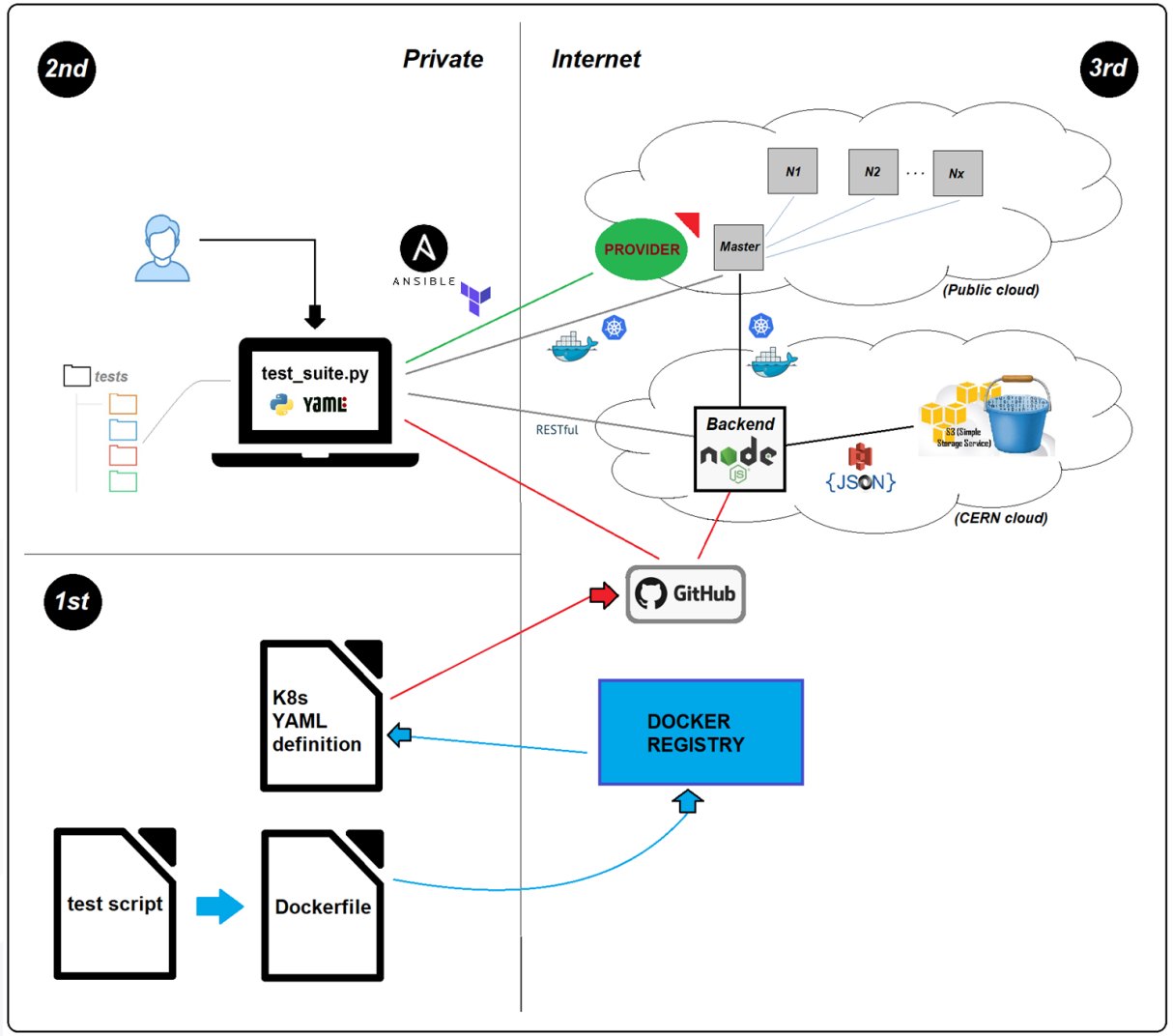
In a nutshell

- Written in **Python**
- VM provisioning with **Terraform**
- **Kubernetes** cluster bootstrapping done by **Ansible**
- Deployment of tests on **Docker** containers to the cluster
- Simple **YAML** configuration
- Results as **JSON** on **S3** bucket on CERN cloud
- **NodeJS** service for verification: results integrity (under development)



Process

1. User clones the public repository
2. Configuration by filling configs.yaml and testsCatalog.yaml
3. The test-suite, according to configs.yaml, will provision raw VMs and then bootstrap a Kubernetes cluster on them
4. Once the cluster is ready, the test-suite will deploy the tests according to testsCatalog.yaml: these run on Docker containers



Test & Benchmarking Catalog

Existing tests

- CPU benchmarking with containers (Domenico Giordano – CERN)
- DODAS: Emulate CMS jobs to verify that the node is able to run real workflows as in DODAS HTCondor environment (Daniele Spiga & Diego Ciangottini – INFN)
- Network: Performance measurements with perfSONAR (Shawn McKee – Univ. of Michigan & Marian Babik – CERN)
- Data Repatriation: Backup data from a commercial cloud provider to Zenodo (Ignacio Peluaga – CERN)
- Storage: Basic S3 endpoint functional test (Oliver Keeble – CERN)
- GPUs: Distributed training and optimisation of Deep Learning models (Sofia Vallecorsa – CERN)

Benchmarking in collaboration with CERN openlab

- Satellite Image analysis and generation with AWS (UNOSAT)
- GPUs (UNOSAT) and FPGAs (Dune Reconstruction) in T-Systems OTC

Starting next month

- Graphcore IPUs (Microsoft Azure)
- Benchmarking TPUs (Google Cloud)

Test & Benchmarking Catalog

Additional tests (integration under development):

- Higgs Demo with Kubernetes (Ricardo Rocha – CERN)
- HPC: FDMNES - Simulation of X-ray spectroscopies (Rainer Wilcke – ESRF)
- Disk access stress, with non-streaming I/O patterns (EMBL)
- HDF5_io (DESY)
- Data isolation (SURFsara)
- SLURM (SURFsara)
- COSBench (Cloud Object Storage services benchmarking)
- Spark test (IT-DB)

In ARCHIVER project, moving up in the stack (not only IaaS):

- FAIR Evaluator (FAIRsharing team, Uni. Oxford): Evaluate the degree of “FAIRness” of repositories, looking to the ingested dataset and their quality: https://github.com/FAIRMetrics/Metrics
- AAI: INDIGO-IAM (INFN-Bologna) test to be integrated: https://github.com/indigo-iam/iam

Example: CPU benchmarking

```
. . .
cpuBenchmarking:
  run: True
perfsonarTest:
  run: True
  endpoint: psb01-gva.cern.ch
hpcTest:
  run: True
  nodes: 3
. . .
```

testsCatalog.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: cpu-bmk-pod
spec:
  hostNetwork: true
  containers:
  - name: cpu-benchmarking-cont
    image: gitlab-registry.cern.ch/cloud-infrastructure/cloud-benchmark-suite/cloud-benchmark-suite-standalone:1.8
    imagePullPolicy: Always
    securityContext:
      privileged: true
    command: ["/bin/sh","-c"]
    args:
      - cern-benchmark --benchmarks="kv;whetstone;DB12;hyper-benchmark" --freetext="Cloud Validation Suite" -o
&& tail -f /dev/null
```

Example: CPU benchmarking deployment

1. Test-Suite completes YAML file according to configuration
2. Deploys the pod using kubernetes API and the YAML file
3. Container on the pod uses the image cloud-benchmark-suite-standalone
4. Once the pod is ready the command defined on the Pod yaml file runs four benchmarks (DIRAC, ATLAS Kit Validation, Whetstone, Hyper-benchmark)

More information on HEP CPU benchmarking:
https://www.epj-conferences.org/articles/epjconf/pdf/2019/19/epjconf_chep2018_08011.pdf

Example: perfSONAR network test deployment

```
. . .
cpuBenchmarking:
  run: True
perfsonarTest:
  run: True
  endpoint: psb01-gva.cern.ch
hpcTest:
  run: True
  nodes: 3
. . .
```

testsCatalog.yaml

```
apiVersion: v1
kind: Pod
metadata:
  name: ps-pod
spec:
  hostNetwork: true
  containers:
  - name: ps-cont
    image: perfsonar/testpoint:latest
    securityContext:
      privileged: true
    imagePullPolicy: Always
```

Example: perfSONAR network test deployment

1. Test-Suite completes YAML file according to configuration
2. Deploys the pod using Kubernetes API and the YAML file
3. Container on the pod uses perfSONAR testpoint image, runs pScheduler
4. Once pod is ready the Test-Suite runs pScheduler tasks remotely using the API: copy python script containing the tests and run specifying the destination host, taken from testsCatalog.yaml, as a CLI parameter.

Example: perfSONAR network test deployment

- Uses pScheduler API on the container instead of CLI
- Runs tests against provided endpoint: throughput, rtt, trace and latency
- Results are written to a JSON file

```
pscheduler task --format=json throughput --dest=$ENDPOINT > throughput.json

pscheduler task --format=json rtt --dest=$ENDPOINT > rtt.json

pscheduler task --format=json trace --dest=$ENDPOINT > trace.json

pscheduler task --format=json latency --dest=$ENDPOINT > latency.json
```

Harvesting results and completion

1. Once the tests complete, the Test-Suite harvests the resulting JSON files from the pods using kubectl
2. Once all results are harvested, pods are killed and resources (VMs) optionally destroyed
3. Verification System: launch the TS –skipping provisioning and bootstrapping phase- from a server running on the CERN cloud. Harvested results are then pushed to an S3 bucket (optional). This feature is still under development.

Next steps

- Use Deployment and/or Job instead of Pod
- Skip bastion method, allow reaching a cluster behind NAT.
- New tests and Updates of current ones
  - S3, perfSONAR, Higgs demo, Deep Learning, etc.
- Increase setting and configuration
  - Different regions, broader OS, stack versions, etc.
- Build website to see results in a human-readable way instead of raw JSON.
- Quota limits management option (currently on the dev. branch, new release next week).

Licensing

- Test Suite framework
  - Developed by CERN-IT-EFP, FOSS under AGPL

- Licensing on tests are responsibility of the authors
  - Terms and repository provided by the contact person

Presence in ISC (Frankfurt)

Selected poster (ISC 2019) and digital poster + lighting talk (ISC 2020)

ISC 2020 more info and registration: https://www.isc-hpc.com/

Testing the Suite Deployment

Get involved

Feedback is welcome!

The test contribution process is established as follows:

1) Discussion with development team to present the test/use case/idea
2) Assessment of the work to be done and set up requirements
3) Info about documentation and contact of the person providing the test plus its license

REPOSITORY: https://github.com/cern-it-efp/EOSC-testsuite

DOCUMENTATION: https://eosc-testsuite.readthedocs.io

CONTACT: developers@archiver-project.eu

ARCHIVER

ARCHIVING AND PRESERVATION FOR RESEARCH ENVIRONMENTS

QA & Testing in ARCHIVER

Jakub Urban (IT-DI-EFP)

ARCHIVER - Archiving and Preservation for Research Environments project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 824516.

Project Objective

Focus: R&D in Archiving and Data Preservation Services using commercial clouds

Procurement R&D budget: 3.4M euro

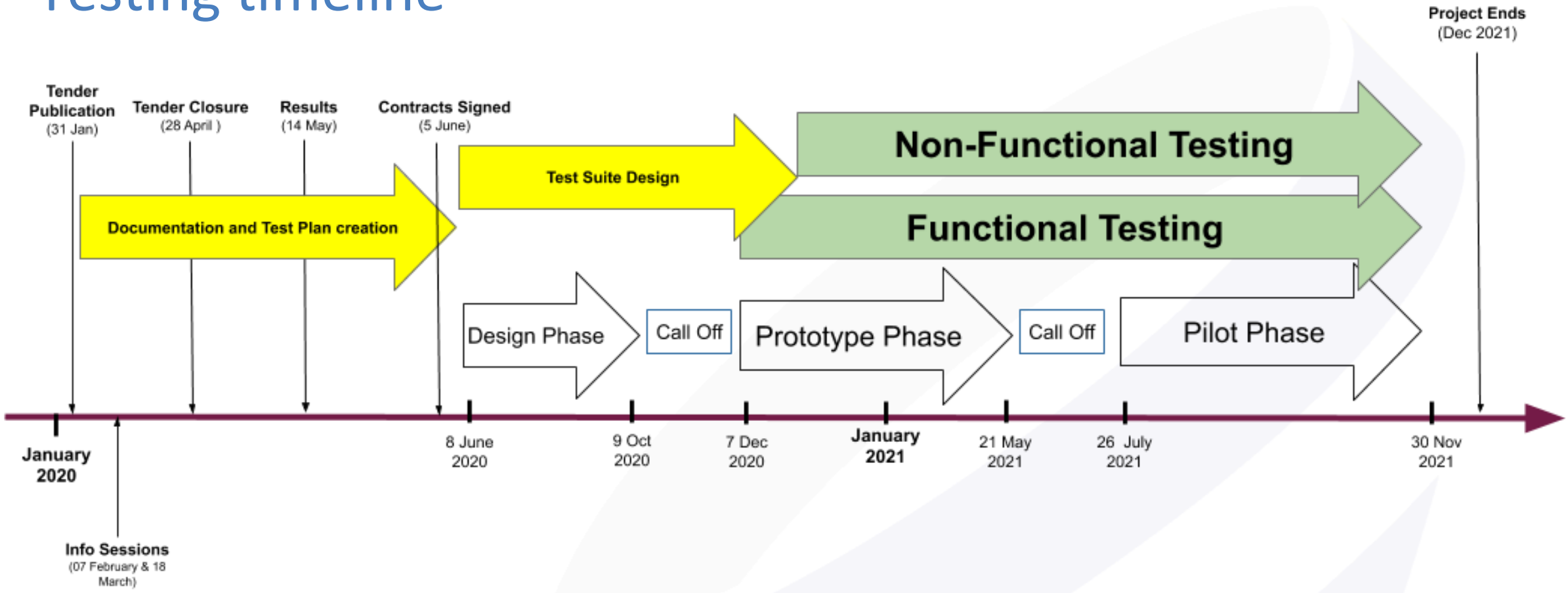
Starting Date: 1st of January 2019

Duration: 36 Months

Coordinator: CERN (Lead Procurer)

European Commission

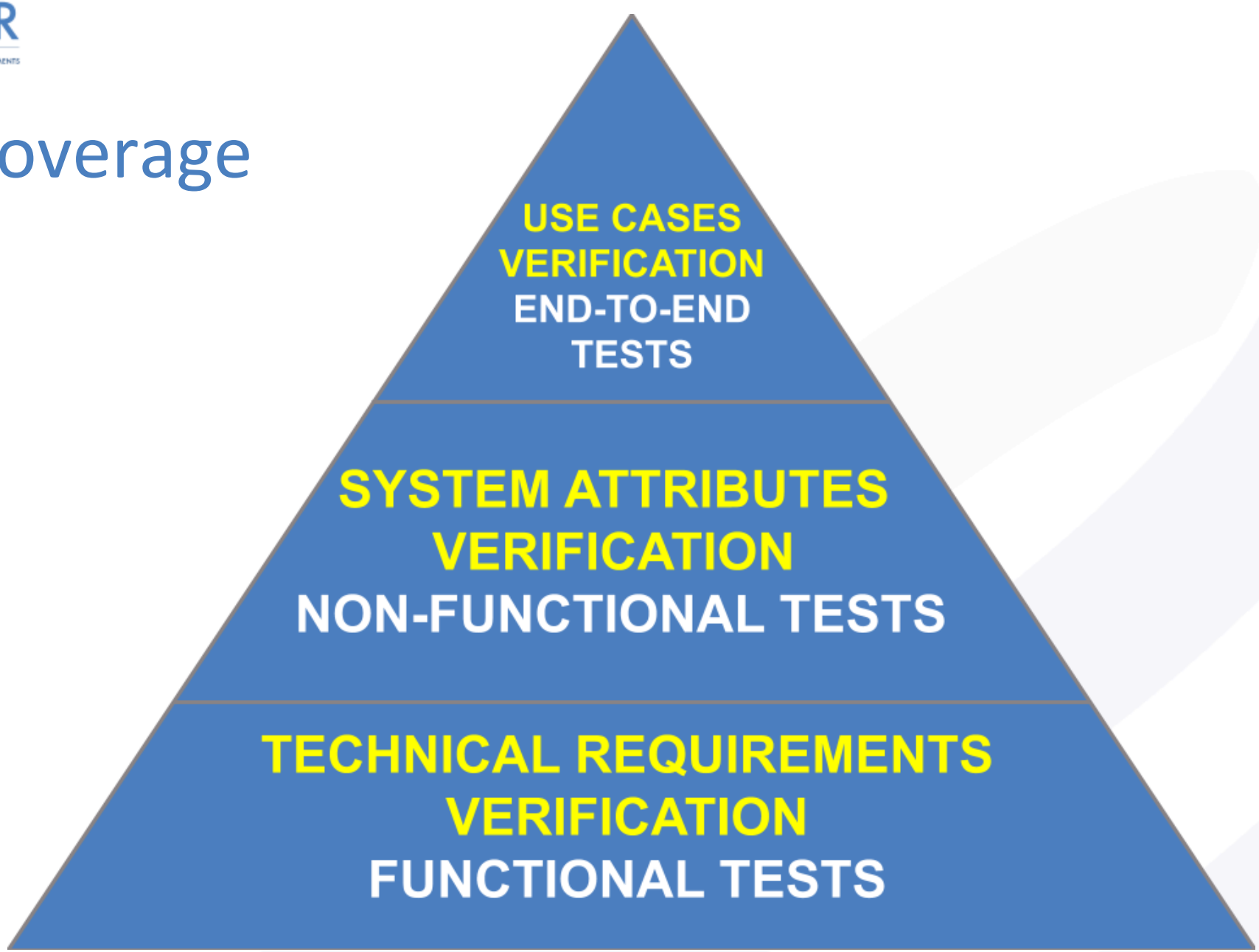
Testing timeline



Testing Approach

- Agile principles
- Frequent feedback
- Continuous Testing Environment
- Open Test catalogue
- Automated reporting
- Human readable results

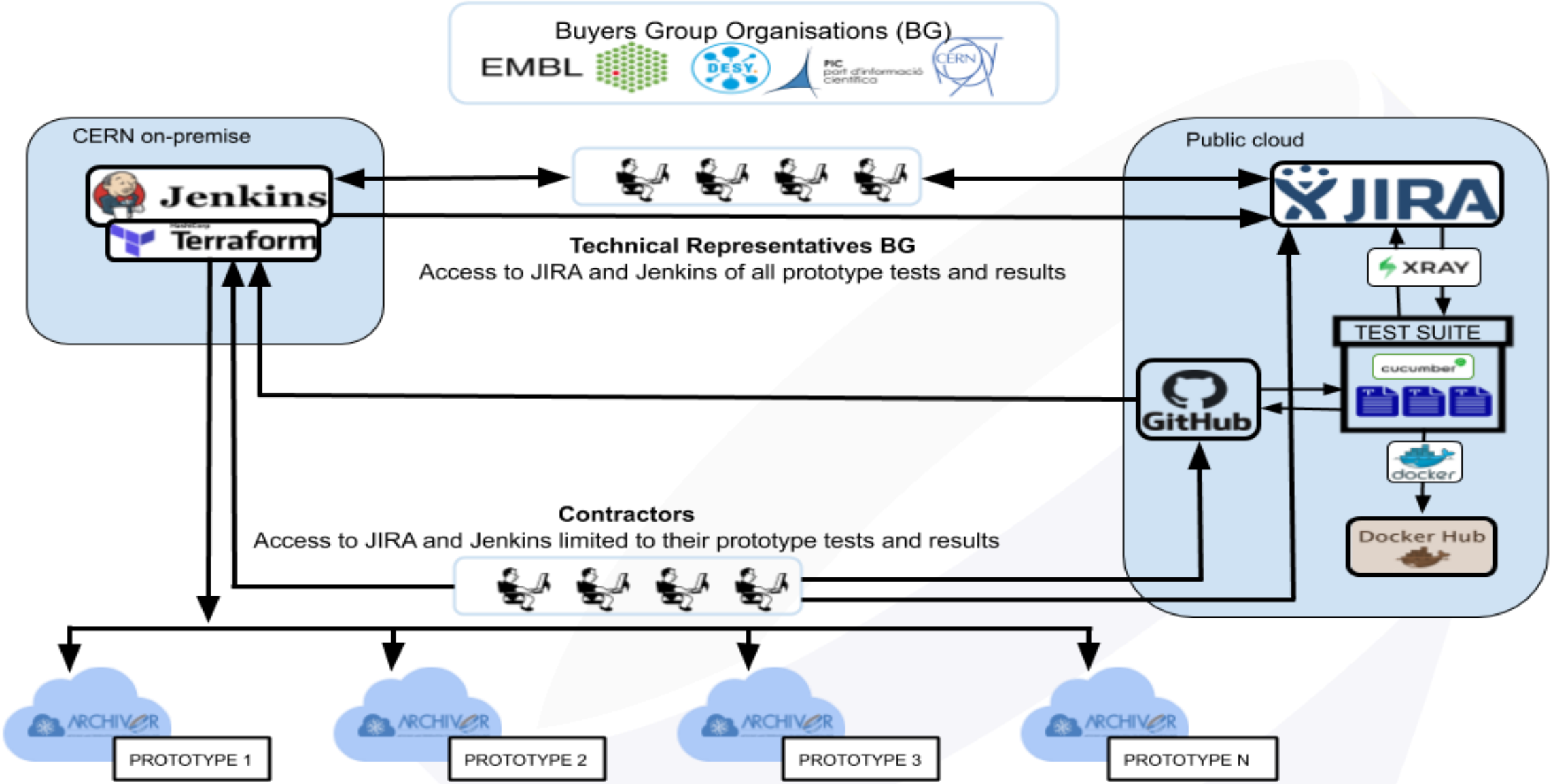
Test coverage



Architecture principles

- Platform independent
- Broad Support
- Isolated
- Configurable
- Documented
- Abstracted
- Modular
- Adjustable
- Accessible
- Open source

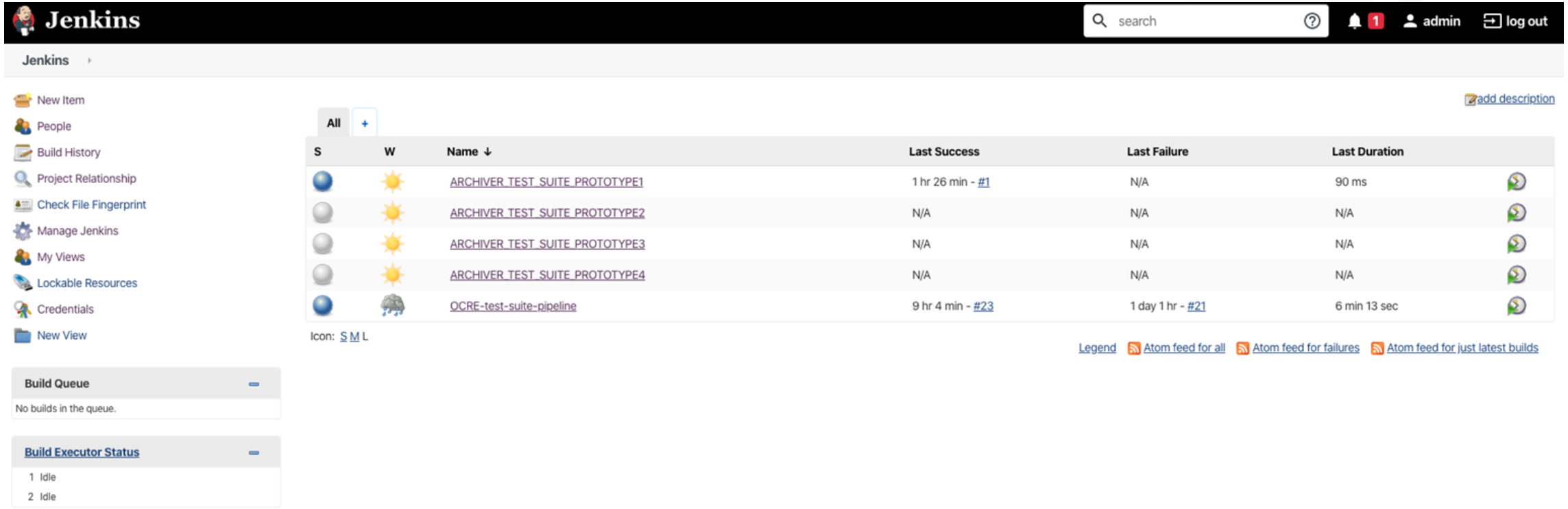
Continuous Testing Environment



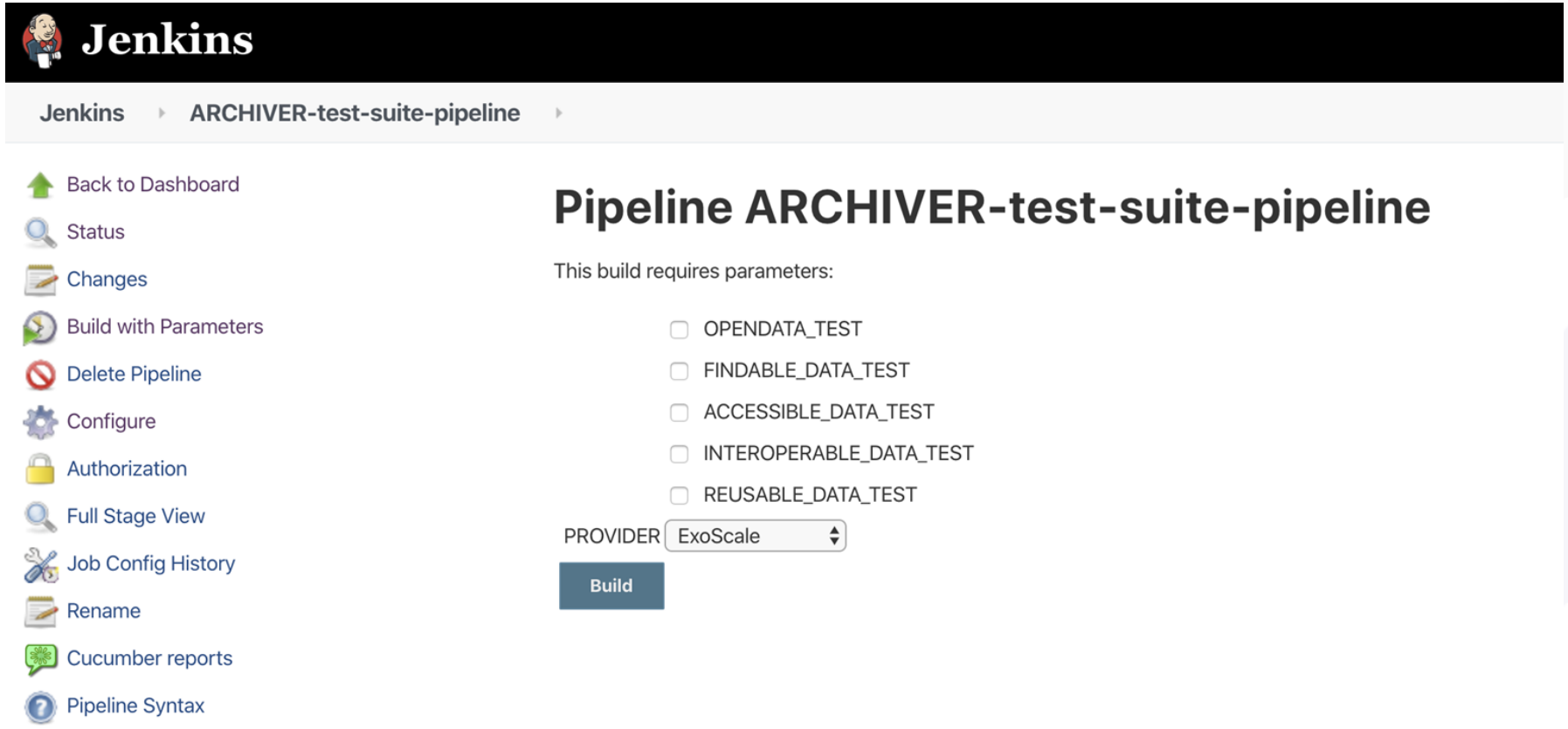
Testing Process

- Tests to be provided by the scientific partners
- Test plan to be co-designed with the contracted companies
- Contractors to test first!
- Confidentiality of test results will be assured

Jenkins access matrix



Jenkins Test Suite triggering



Jenkins Test Suite triggering

Pipeline ARCHIVER-test-suite-pipeline

Recent Changes

Stage View

	SCM	Set-up	Test	Import results to Xray	Generate HTML report
Average stage times: (Average full run time: ~12s)	853ms	138ms	1s	1min 10s	321ms
#42 May 11 10:35 No Changes	2s	141ms	3s	5s	572ms

Jenkins Test Suite reports

Feature Report

	Steps						Scenarios			Features	
Feature	Passed	Failed	Skipped	Pending	Undefined	Total	Passed	Failed	Total	Duration	Status
OpenData	89	0	0	0	0	89	12	0	12	1.778	Passed

Feature OpenData

Tags: AR-250
Scenario Verify file out of dataset 0.122

Tags: AR-249
Scenario Verify title is matching 0.040

Tags: AR-244
Scenario Verify number of files present 0.021

Tags: AR-248
Scenario Verify validation info is present 0.020

Tags: AR-253
Scenario Verify dataset semantics table 0.024

Localhost Test Suite triggering

```
jakubs-macbook-pro:archiver_cucumber jakub$ behave ./features/OpenData.feature
Feature: OpenData # features/OpenData.feature:1

  @AR-250
  Scenario: Verify file out of dataset                                    # features/OpenData.feature:4
    Given the provider 'ExoScale'                                         # features/steps/steps.py:47 0.000s
    When retrieve DOI of record which id equals 1 from opendata           # features/steps/OpenDataSteps.py:31 0.151s
    When retrieve DOI of record which id equals 1 from provider           # features/steps/OpenDataSteps.py:31 0.103s
    When retrieve title from metadata of record with the given DOI from opendata # features/steps/OpenDataSteps.py:39 0.256s
    When retrieve title from metadata of record with the given DOI from provider # features/steps/OpenDataSteps.py:39 0.259s
    When retrieve doi from metadata of record with the given DOI from opendata   # features/steps/OpenDataSteps.py:39 0.227s
    When retrieve doi from metadata of record with the given DOI from provider   # features/steps/OpenDataSteps.py:39 0.172s
    Then opendata and provider title elements are the same                # features/steps/OpenDataSteps.py:267 0.000s
    Then opendata and provider doi elements are the same                  # features/steps/OpenDataSteps.py:267 0.000s
```

Localhost Test Suite triggering

```
jakubs-macbook-pro:archiver_cucumber jakub$ behave ./features/OpenData.feature --no-capture --format plain
Feature: OpenData

  Scenario: Verify file out of dataset
    Given the provider 'ExoScale' ... passed in 0.000s
    When retrieve DOI of record which id equals 1 from opendata ... passed in 0.121s
    When retrieve DOI of record which id equals 1 from provider ... passed in 0.071s
    When retrieve title from metadata of record with the given DOI from opendata ... passed in 0.180s
    When retrieve title from metadata of record with the given DOI from provider ... passed in 0.171s
    When retrieve doi from metadata of record with the given DOI from opendata ... passed in 0.204s
    When retrieve doi from metadata of record with the given DOI from provider ... passed in 0.285s
    Then opendata and provider title elements are the same ... passed in 0.000s
    Then opendata and provider doi elements are the same ... passed in 0.000s
```

Cloud Jira integration

Projects / ARCHIVER / AR panel

Backlog

JU Only My Issues Recently Updated

VERSIONS EPICS

Execution results [1588085316638]	JU	AR-241
Execution results [1588085862294]	JU	AR-242
Execution results [1588110565226]	JU	AR-243
Verify number of files present		AR-244
Verify dataset full download		AR-246
Verify validation info is present		AR-248
Verify search for author		AR-247
Verify title is matching		AR-249
Verify file out of dataset		AR-250
Verify search for dataset name		AR-252
Verify dataset semantics table		AR-253
Verify dataset file size		AR-251
Verify dataset partial download		AR-254

AR-244

Verify number of files present

Description

Add a description...

Test Details

Cucumber

Test Repository

Scenario

```
1 Given the provider 'ExoScale'
2 When retrieve count of files for record wh
3 When retrieve count of files for record wh
4 Then opendata and provider have the same a
```

Cloud Jira integration

Test Details

Test Type: Cucumber

Scenario Type: Scenario

Scenario:

```
1 Given the provider 'ExoScale'
2 When retrieve count of files for record which id equals 4910 from opendata
3 When retrieve count of files for record which id equals 4910 from provider
4 Then opendata and provider have the same amount of files
```

Results

Context	Duration	Status
-	21 millisec	PASSED
Steps		
Given the provider 'ExoScale'	0 millisec	PASSED
When retrieve count of files for record which id equals 4910 from opendata	10 millisec	PASSED
When retrieve count of files for record which id equals 4910 from provider	10 millisec	PASSED
Then opendata and provider have the same amount of files	0 millisec	PASSED

Get involved

Any feedback is welcome!

CONTACT: developers@archiver-project.eu