Software Provisioning and Infrastructure (UMD/CMD)

Mario David (LIP), João Pina (LIP), Carlos Manuel (LIP), Jorge Gomes (LIP), Samuel Bernardo (LIP), Pablo Orviz (IFCA)

EGI Repositories

The EGI Software Repository provides access for the middleware distributions (**used by WLCG sites)** together with some Community Repositories

- Constitutes a contribution from IBERGRID to the WLCG infrastructure

Composed by:

- **Software Quality Assurance -** validation of the products released as part of the UMD
  - Software Quality criteria - **definition of the criterias** (done once per product)
  - Software validation of the criteria - **run tests** against the define criterias (done for every release)
  - Done by IBERGRID since 2012 now LIP + CSIC (ex partners: UPV, CESGA)
  - Contract paid by EGI under competitive call
- **Repositories Infrastructure -** provides a **unified** point of access for the middleware distributions (IBERGRID since 2021)

EGI Repositories

Two main categories:

- The Unified Middleware Distribution (**UMD**): redistribution of traditional middleware

- **IGTF** Distribution of Authority Trust Anchors: packages with the trust anchor information

OS Support
- AlmaLinux9 (EL9) : **UMD-5**

What's new

- Complete redesign of the service:
  - Current infrastructure based on Nexus Repository OSS
  - Release workflow implemented in GitHub and Jenkins pipelines.
- Software Quality assurance - Full-automated approach
  - **Faster release cycle** with higher automation and less manual intervention

New Functionalities:
- IPV6 compliant
- New backend behind a proxy (HAProxy)
  - https://repository.egi.eu/sw/production/ (prod)
  - https://repository.egi.eu/software/umd/ (dev)
- High Availability (server side)
- Full automatic process until testing
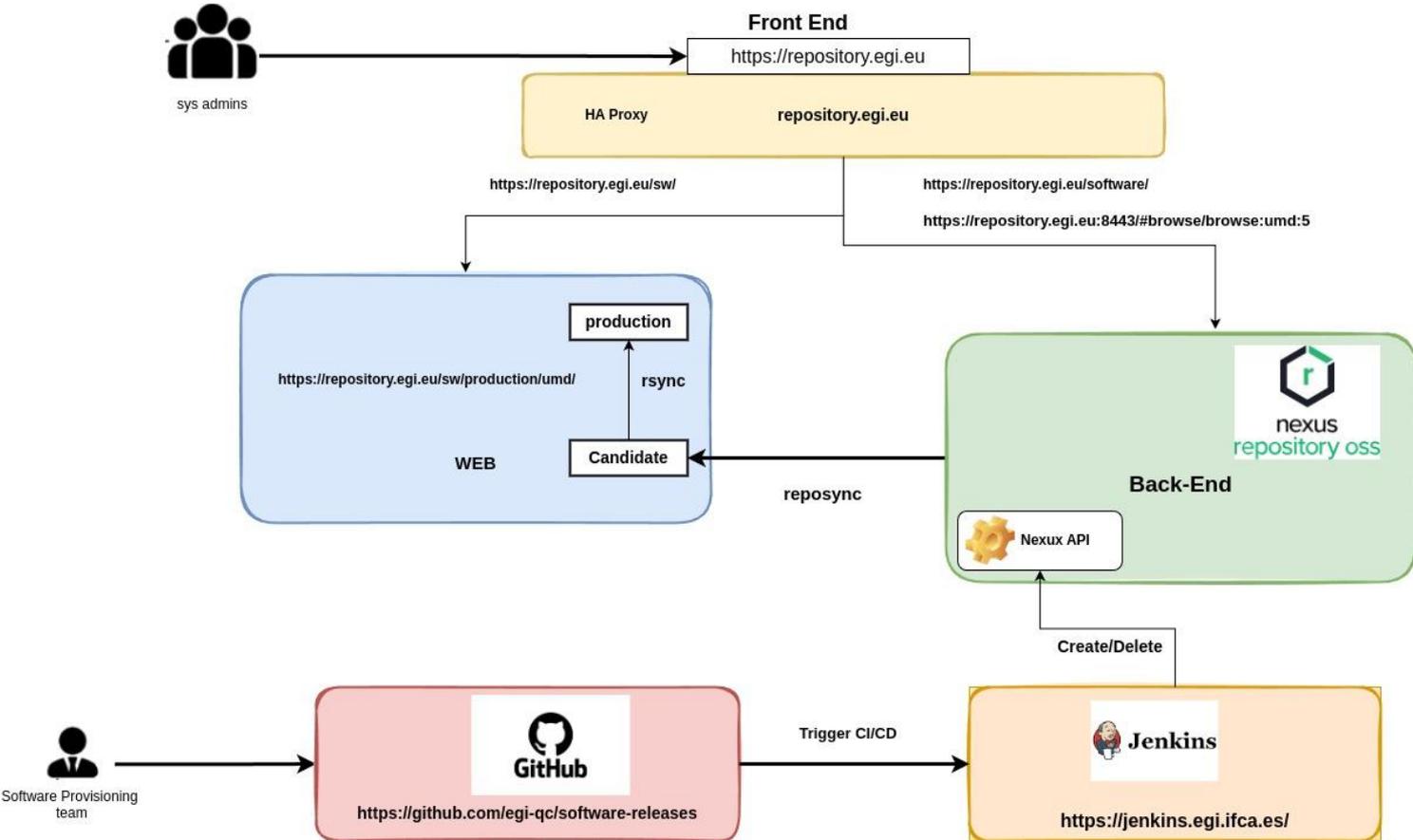- Testing -> production (manual git commit)

Yum repositories (UMD-5 structure)

1. **`release`**: Main repository with all the packages (**= to production**).

2. **`testing`** : testing repository with rpm not fully validated in production environment. RPM belonging to release declared as production ready will be moved from testing to release.

3. **`contrib`**: repository for specific communities or special services. (**NEW**)
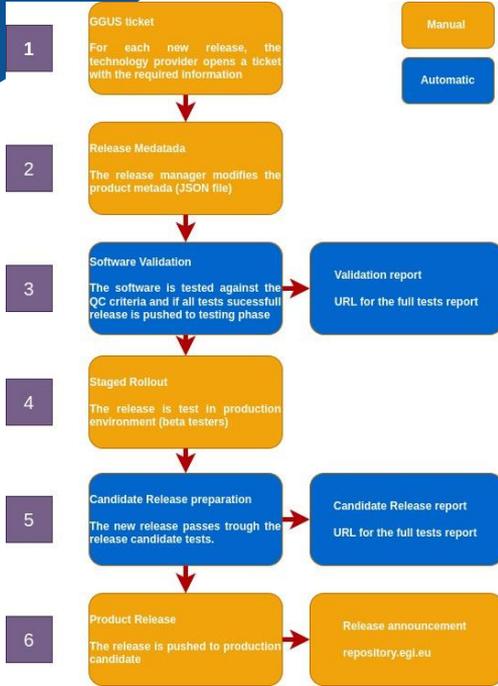
Dropped repositories: Staged Rollout and **Release Candidate**

- UMD-5 **Production since (October 2024)**
  - **https://repository.egi.eu/umd/distribution.html?id=5.0.0#5.0.0**
  - **Install instructions:**
  - **https://repository.egi.eu/umd/installation-notes.html?id=5**

Architecture

Front End

https://repository.egi.eu

HA Proxy **repository.egi.eu**

sys admins

https://repository.egi.eu/sw/

https://repository.egi.eu/software/

https://repository.egi.eu:8443/#browse/browse:umd:5

production

https://repository.egi.eu/sw/production/umd/ **rsync**

WEB **Candidate**

reposync

nexus
repository oss

Back-End

Nexux API

Create/Delete

Software Provisioning
team

GitHub

https://github.com/egi-qc/software-releases

Trigger CI/CD

Jenkins

https://jenkins.egi.ifca.es/

Workflow

Workflow

Architecture

7

Git
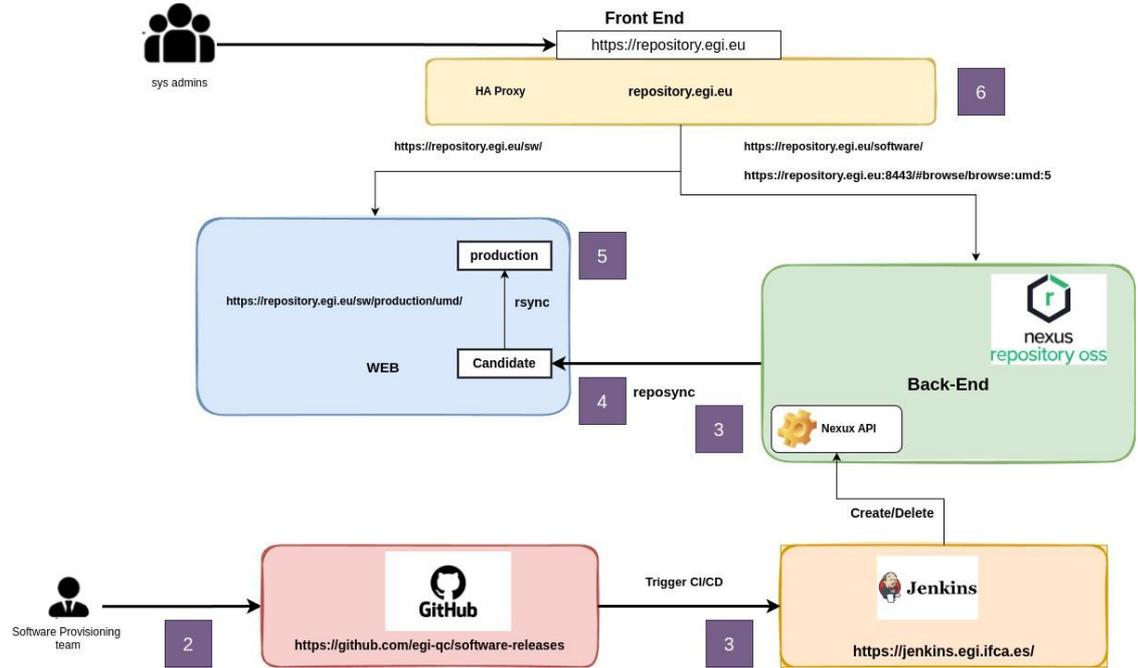
Tracking of the workflow all based on git

Software provisioning process:
https://github.com/egi-qc

Software Releases:
https://github.com/egi-qc/software-releases

Ansible playbooks for install software:
https://github.com/egi-qc/umd-verification

Workflow

3

Pull Request PR-130

Full project name: egi-qc/software-releases/PR-130

Last Successful Artifacts
📄 release.json   1.81 KB   👁 view

Stage View

	Declarative: Checkout SCM	Declarative: Agent Setup	Install dependencies	Detect release changes	Get release info	Collect the list of packages	Download the packages to a temporary directory	Add UMD GPG key	Upload packages to testing	Trigger validation	Generate JSON release file	Trigger Release Candidate validation	Production download packages to a temporary directory	Upload packages to production
Average stage times: (Average full run time: ~1min 51s)	1s	852ms	3s	349ms	892ms	863ms	3s	2s	3s	1min 28s	693ms	0ms	0ms	0ms
#1 Oct 14 14:54 No Changes	1s	852ms	3s	349ms	892ms	863ms	3s	2s	3s	1min 28s	693ms			

Permalinks

- Last build (#1), 4 min 23 sec ago
- Last stable build (#1), 4 min 23 sec ago
- Last successful build (#1), 4 min 23 sec ago
- Last completed build (#1), 4 min 23 sec ago

S	W	Name
✓	☀	package-install
✓	☀	release-candidate
✗	☁	cas
✓	☁	grid-clients
✗	☁	xrootd
✗	☁	frontier-squid
✗	☁	arc
✗	☁	gridftp
✓	☁	cloud-info-provider
✗	☁	dpm
✓	☀	run
✓	☁	gridsite
✗	☁	bdi-site
✗	☁	fts
✓	☁	caso
✓	☀	argus
✗	☁	cream
✗	☁	user-interface
✗	☁	ooi
✓	☁	worker-node
✗	☁	bdi-top
✗	☁	keystone-voms

Icon: S   M   L

Moving product to testing repository

https://github.com/egi-qc/umd-verification

✓ **Build #268** (Oct 14, 2024, 1:55:04 PM)

Build Artifacts
📄 qc_conf.stdout   71.17 KB   👁 view

No changes.

Started by upstream project egi-qc » software-releases » PR-130 build number 1
originally caused by:
- Pull request #130 opened at 1:54:17 PM on Oct 14, 2024

Docker build metadata:
- **Cloud:** docker-cloud
- **Container ID:** 350654e9ec6e35a31ed817cc7e5456e4ede5c416d020ab487d3a535cb64e845F

This run spent:
- 12 sec waiting;
- 1 min 15 sec build duration;
- 1 min 28 sec total from scheduled to completion.

Revision: a5bd4c71a797a6037fa8a51b5a491d528e41a473
Repository: https://github.com/egi-qc/umd-verification
- refs/remotes/origin/master

https://jenkins.egi.ifca.es/job/Quality
CriteriaValidation/job/package-insta
ll/268/artifact/logs/qc_conf.stdout/*v
iew*/

9

Workflow

Moving product to prod repository

Pull Request PR-129

Full project name: egi-qc/software-releases/PR-129

Stage View

https://github.com/egi-qc/ansible-release-candidate

https://jenkins.egi.ifca.es/job/QualityCriteriaValidation/job/release-candidate/409/artifact/logs/qc_conf.stdout/*view*/

Evolution

Future:

- Integration with SQAaaS from EOSC-Synergy
  - **Allow full pipeline tests**
- Allow it to be used by **external users**:
  - EGI / EOSC marketplace
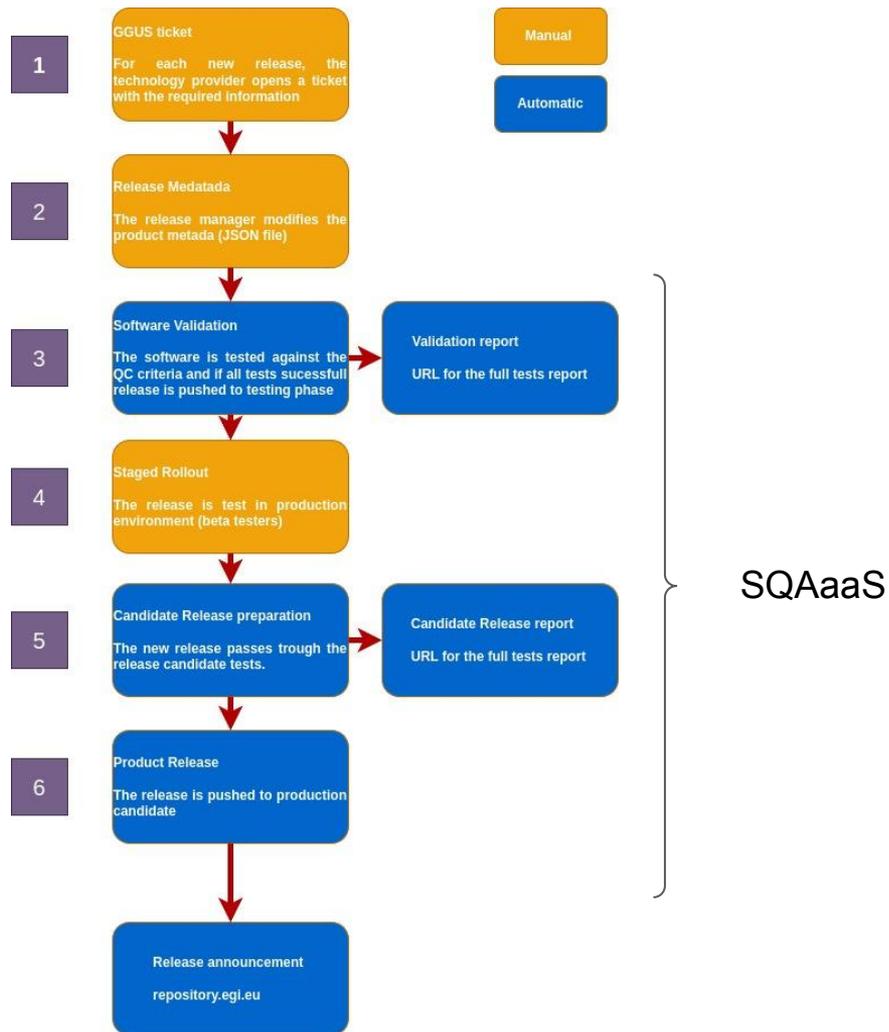
Yum repositories (new structure)

Production

[UMD-5]

name=UMD 5 main repository (Alma Linux 9)

baseurl=https://repository.egi.eu/sw/production/umd/5/al9/release/$basearch

enabled=1

priority=1

gpgcheck=1

gpgkey=https://repository.egi.eu/repository/umd/5/UMD-5-RPM-PGP-KEY

EPEL 9 required

UMD higher priority than EPEL

Priorities already part of DNF

Need crb almalinux9 enabled

No more base / updates >> release

Yum repositories (new structure)

Testing

[UMD-5-testing]

name=UMD 5 nexus testing (Alma Linux 9)

baseurl=https://repository.egi.eu/sw/production/umd/5/al9/testing/$basearch

enabled=0

priority=1

gpgcheck=1

gpgkey=https://repository.egi.eu/repository/umd/5/UMD-5-RPM-PGP-KEY

Packages under testing but already validated

Yum repositories (new structure)

External Contribution (new)

[UMD-5-contrib]

name=UMD 5 contrib (Alma Linux 9)

baseurl=https://repository.egi.eu/sw/production/umd/5/al9/contrib/$basearch
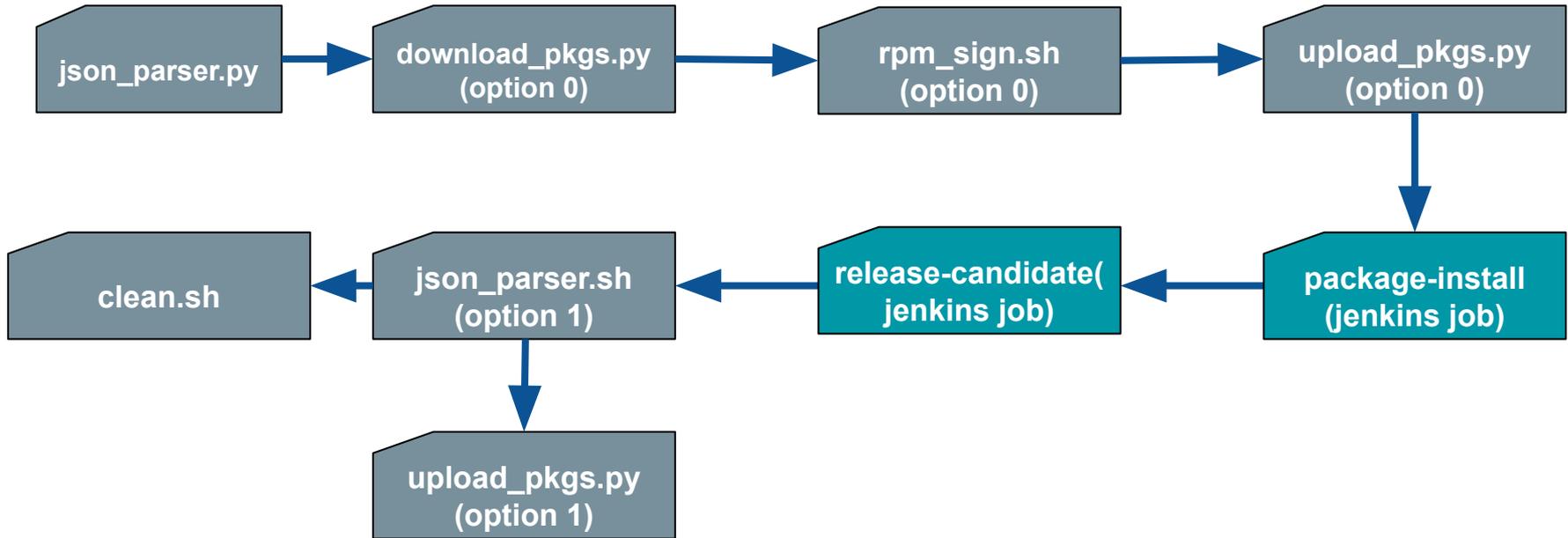
enabled=0

priority=1

gpgcheck=1

gpgkey=https://repository.egi.eu/repository/umd/5/UMD-5-RPM-PGP-KEY

Packages under testing but already validated: Caso

UMD/CMD new Workflow

https://github.com/egi-qc/software-releases

```
json_parser.py  →  download_pkgs.py  →  rpm_sign.sh  →  upload_pkgs.py
                    (option 0)           (option 0)       (option 0)
                                                               ↓
clean.sh  ←  json_parser.sh  ←  release-candidate(  ←  package-install
             (option 1)         jenkins job)            (jenkins job)
                  ↓
             upload_pkgs.py
             (option 1)
```

Approved the Release (Testing/Production repository)

Pipeline

https://github.com/egi-qc/software-releases

The pipeline is as follows:

1. `json_parser.py`: parse json, get the list of files to download and produce list of filenames (packages).

2. `download_pkgs.py` (option 0): download the packages to a temporary directory.

3. `rpm_sign.sh` (option 0): rpm sign each package.

4. `rpm_sign.sh` (option 0): verify signature of each package.

   a. a. verification of the packages

5. `upload_pkgs.py`: upload each package to nexusrepo.

6. `Package-install`: Validate package installations from testing repository and perform functional tests.

7. `release-candidate`: Install all packages in release repo together with the new packages from testing

8. `json_parser.sh`: Produce new json file as asset of the new release

9. `clean.sh`: clean temporary directories.