EGI Quality Assurance

Everything that concerns Quality Assurance should go here, e.g.:

- QC Definition processes
- QC Verification processes
- Reference material

Criteria definition

The Quality Criteria (QC) Definition Task is responsible of generating the Quality Criteria Documents that drive the Quality Criteria Verification.

Quality Criteria definition is a continuous process driven by the requirements of users and operations communities, however the verification of new software releases is done against fixed date releases of the QC documents, due every 6 months. Between major these releases, two draft versions are made available for lightweight review.

The QC releases are done in coordination with the EGI Roadmap and Technology releases.

There are 3 different possible states for the Quality Criteria documents:

- **FINAL**, meaning that the documents are actively used for verification of software products
- **DRAFT**, for documents that are in preparation and will be used in the future for verification.
- **DEPRECATED**, for documents that are no longer used in verification.

Please check the dissemination information in order to ensure that you are using the correct version of the QC documents.

Quality Criteria and UMD Releases

The QC documents collect criteria for capabilities of a specific UMD Major Release. Only one **FINAL** version of QC documents is active for any given UMD Release. In the case of having more than one major UMD release active, i.e. new products are verified and made available in different UMD repositories, the verification will use the **FINAL** version of QC documents for each UMD release. Check the dissemination information for determining which QC version should be used for the verification.

Quality Criteria Change Management

Changes in the criteria documents are triggered by one of the following sources of changes:

- Requirements from User Community
- Requirements from Operations Community (especially software incidents found in production)
- Deficiencies in criteria found in Verification or Stage Rollout
- Recommendations and issues found by the Software Vulnerability Group
- Analysis of reference implementations of UMD Capabilities defined in the Roadmap
- Review and analysis of feedback from Technology Providers

Any change to criteria is tracked in the *Related Information* field that includes any links to direct source of change for the criterion (e.g. RT ticket with user requirement) and the *Revision Log* field, that records the historic information about the changes in the criterion.

Moreover, the Quality Criteria Release Notes include an overview of the most relevant changes produced in the whole set of criteria documents.

Quality Criteria Metrics

Number of UMD Roadmap Capabilities defined through validation criteria: This metric tracks the coverage of UMD Capabilities with Quality Criteria. The goal is that for each UMD Capability a number of Quality Criteria are recorded that together define the quality of software contributed against that UMD Capability. It is recorded as a percentage of covered capabilities Number of software incidents found in production that result in changes to quality criteria**: This metric tracks the quality of the defined criteria. Any incidents (bugs) found in EGI Infrastructure should result result in a change of the Quality Criteria. The metric considers also security incidents found by the SVG.**

Number of "change triggers" that result into an update of the Quality Criteria (Internal): Similar to the previous one (and includes it), this internal metric tracks the quality of the defined criteria by counting the number of change triggers (as defined previously in change management section) produce changes in the Quality Criteria.

Criteria Verification

The main objective of the TSA2.3 is to verify the quality of the software provided by the TP before entering the SR phase and going to production. By doing so we prevent that software that might work enters into the SR and even goes into production but that doesn´t follow the quality criteria defined in TSA2.2. Some of the reasons for doing the verification before the software enters the stage rollout are:

- Check that the bugs reported in the previous release of the software have been corrected (work in collaboration with DMSU) by the TP.

- Software can work well in the SR but might not have all the functionalities required

- Software might not be safe, well documented, or have the necessary installation rules or licenses

The Verification Process

When a new product is available, the TP has to follow the NSRW. Once the software is correctly uploaded to the repository, the release enters into the verification phase. The requirements are that the TP has to provide all the necessary information to the verifier (QCV) so that the QCV can assess that the TP has tested in advance the quality of the software. Depending on the type of release, different actions will be taken by the QCV, Verification process is described in detail in the EGI Verifier Guideline.

QC Verification Reports

RT workflow and Verification Templates links are available in this page. First of all verifiers must check QC service mapping to know which test must be verified for each product. This service mapping is available here QC Verification service mapping and Verification/Executive Summary templates are available here: QC Verification Templates.

Verifiers must fill all required fields for each product and write a Verification process summary into Executive Summary, this summary should include:

- A short summary of the installation and configuration process: Installation and configuration were successful?, If not explain any issue or bug found. If the product was rejected explain why.
- If its necessary Verifiers should include a comment for StageRollout: Configuration changes or minor issues found verifying the product.
- If a new QC is necessary and is not included, Verifiers must write a comment to SA2.2 to change current QC.

UMD Release Candidate Testing

Before each UMD release the verification team checks the new RC. To perform this task SA2.3 VMs have included the RC_testing script. This script is available after each SA2 VM instantiation (into /root/configuration_templates/tools directory). This script is able to detect any package inconsistency after UMD RC repo configuration. The RC testing process is as follows:

- Verifier should instantiate different Linux flavours VM. For UMD case SL5, SL6 and debian.
- Install UMD RC repo files from UMDx RC page.
- Check RC_tester "PRODUCTS" array. **This array must be updated to include all UMD products!**
- It is recommended to use *screen* program before RC_tester execution. RC tests take about 2h/3h to finish (depends on OS used and the number of products).
- Run *RC_tester* and follows its instructions.
- *RC_tester* generates several logs into log directory, after its execution SA2 verifier should check:
  - *installator_OK.log*: List of metapackages installed correctly.
  - *installator_ERROR.log*: List of metapackages which contain any issue.
  - *<PRODUCT_NAME>_OUTPUT.log*: Complete Product installation log.
  - *<PRODUCT_NAME>_postupdate.log*: Info about product update execution. It detects any issue updating current UMD products.

Current Quality Criteria

The current **FINAL** version of Quality Criteria is available at http://egi-qc.github.io/. Testing procedures for generic criteria are available at EGI_QC_Testing, Specific criteria are available at EGI_QC_Specific

The Verifier Guideline

Verification of software products is managed through EGI's RT, every product will consist of one or more tickets that include the information about the product and links to the repositories where the packages are available.

> ***External Verification:***
> *If you are performing an external verification, you do not need to manage the RT tickets. Instead you should perform the verification and provide the final report.*

In order to start a verification, the verifier must:

1. Set RT ticket(s) **Owner** with the current verifier.
2. Set **UMDRelease** to the appropriate UMD Release (currently 2 or 3) and save the state.
3. Change RolloutProgress to **In Verification** to start the actual verification.

Core Components Verification List

In order to be able to better organize the verification activity a list of core components that MUST to be verified. The list is available here

Installing the products

Each product to be verified must be installed in a controllable environment where tests are easily reproducible. The EGI Verification Testbed may be used for this purpose (check the link for requesting new VMs for the products).

Products **must** be installed (or upgraded) using the packages from the repository linked in the RT ticket. The URL for this repository can be found at the Custom Fields of the RT ticket within attribute *RepositoryURL*. The easiest way of adding the repositoy is using the files available at the *repofiles* directory.

Verification is performed using UMD repositories + OS provided repositories + EPEL (for RH based distributions). No other external repositories may be used.

Once the product is installed (or upgraded), the verifier must perform a basic configuration that allows testing of the product features. This configuration should use the services already existing in the testbed if they are needed. Check EGI Verification Testbed for a list of available services. Support for dteam and ops.vo.ibergrid.eu VOs is expected for all services.

Sample configurations for some of the products are available at configuration-templates github repo.

QC Assessment & Verification report

Each verification must be documented with a final report that includes an executive summary of the result and a summary of the tests performed. The tests to be performed are described in the the current QC document, see also below and at the http://github.com/egi-qc/qc-tests GitHub repository for more information. Verification report template is available at EGI DocDB #1993 in different formats.

The effort dedicated to testing the criteria should be adjusted depending on the type of release:

The effort dedicated to those criteria should be adjusted depending on the type of release:

- **Major** releases (may not be backwards compatible) or first time the product enters EGI's process:
  - Product installation from scratch (optionally also upgrade).
  - Generic QC assessment.
  - Verifier **must** assess product meets its functionality.
  - Test any new functionality.
- Other releases (**Minor** or **Revision**):
  - Package installation or upgrade.
  - Check those features affected by update changes (described at TP release notes). The rest of criteria, should be filled in the report as *Not Tested* and if available a link to the report where the testing of the criterion was performed.

Generic QC

Tests for the generic QC are described in http://egi-qc.github.io/. Each criterion includes in the *What to check* and *What to report* fields describe what needs to be tested and what to include in the verification report respectively. More detailed information about the tests is also available at EGI_QC6_Testing .

Specific QC

Specific QC depends on the product to be verified. There are two criteria in this section:

- Basic Funcionality Test: This includes basic functionality tests of the product that assures that it performs its intended functionality (e.g. a CE is able to submit jobs). These tests do not need to cover the complete functionality or stress the product and should be restricted to the verification testbed. A list of proposed tests for each product is available at EGI_QC6_Specific. If the product does not have any proposed tests yet, the verifier may decide which product functionality to test based on his/her experience with the product.
- New Features/Bug Fixes Test: These tests must focus on new features or fixed bugs of the release. Verifier must check the release notes and test those new features that are feasible to test in the verification testbed within reasonable effort. Take into account that StageRollout will also test the product in a production environment.

Handling issues during verification

If the Verifier finds any problems or issues with the product (any of the criteria is not met or problems to install/configure/operate the product), either they are clarified within the ticket by the verification team by using the verifiers list (sw-rel-qc(at)mailman.egi.eu) or, if the problems needs the interaction of the Technology Provider, a GGUS ticket should be opened.

If a GGUS ticket is needed:

- The ticket should be created with default priority and the short description should begin with **UMD Verification** and then describe the issue.
- Priority may be set to *urgent* if the issue is a show-stopper.
- A link to the GGUS ticket must be included into the *RelatedGGUSTickets* RT field. It should be also listed in the verification report.
- RT ticket *RolloutProgress* should be changed to **Waiting for Response** status. Once the issue is solved, it should be changed again to **In verification**.

Finishing verification

The product is considered as **ACCEPTED** if all the criteria marked as **critical** pass. Otherwise it will be **REJECTED**

Once filled, the verification report must be uploaded as a new doc in DocDB with the following information:

- *Title*: **QC Verification Report: <PRODUCT_NAME>-<VERSION>**.
- The File field is used to upload Verification Report (as PDF preferably).

- *Keywords*: must include **QualityCriteria**, **Verification**.
- *Media type* must be set to **Document**.
- *status* must be set to **FINAL**.
- *View* must be set to **public**.
- *Modify* should be set to **umd-team**.
- *TOPICS* space field must be set to **Software Provisioning Verification Reports** and **WP5**.

Finally, the result of the verification must be set by changing the *RolloutProgress* field of the RT ticket(s).

- If the product is accepted (i.e. meets all critical criteria) then change this field to the **StageRollout** value. This will cause the Rollout teams to continue with the software provisioning process.
- If the product is not accepted, then change the value to **Rejected**, causing the process to stop.
    - If the rejection involves two or more RT tickets, the verifier **must** fill link the verification report for every RT ticket.

When the process is finished (the product is accepted or rejected) the verifier must also fill the "Time Worked" RT field to account the real hours/minutes spent to finish the verification process.