
# Software Quality Assurance (SQA) Report

13-17 Jun 2016

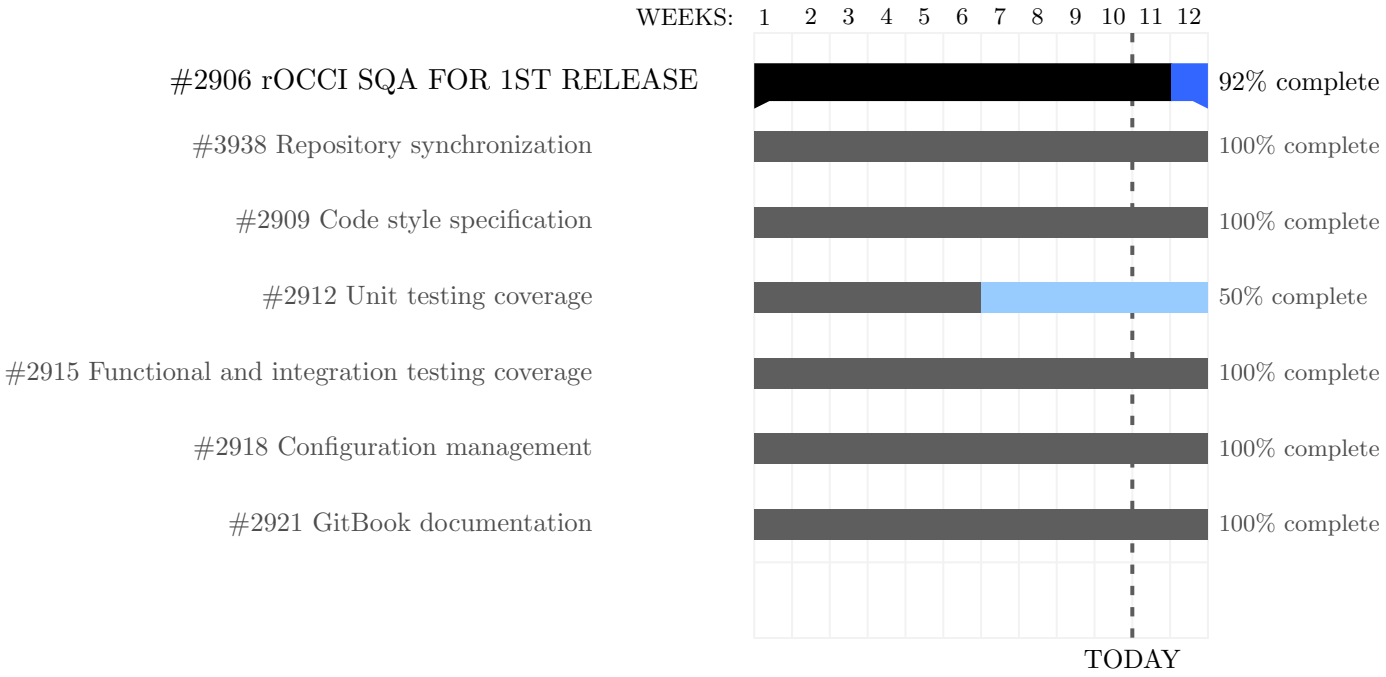
# rOCCI

SQA Progress Status COMPLETE

92% done

| | |
|---|---|
| GitHub repository | COMPLETE |
| Code style adherence | COMPLETE |
| Code coverage | IN PROGRESS - WP3 |
| Functional/integration testing | IN PROGRESS - WP3 |
| GitBook documentation | COMPLETE |
| Automated deployment | IN PROGRESS - WP3 |

# Part I
# Task Progress for the 1st Release




## 1 Repository synchronization

*Products contributing to INDIGO-DataCloud project must have their code avaiable under GitHub's `indigo-dc` organization.*

Repository exists under `indigo-dc` GitHub organization:

- https://github.com/indigo-dc/rOCCI-server.git

## 2 Code Style

*Products contributing to INDIGO-DataCloud project are expected to be adhered to a community or de-facto standard code style definition. Exceptions can be made to the selected standard. Custom style guides are accepted but nonetheless not recommended.*

| | |
|---|---|
| Code style definition | Ruby Style Guide |
| Community/de-facto standard | Yes |
| Exceptions | just lint cops checked |
| Richness | ¡to check¿ Errors None Warnings None link |

### 2.1 Build status

Last build status on Jenkins CI rocci-codestyle.

## 3 Unit Testing

*Code coverage will be tracked for the INDIGO-DataCloud related products and must not decrease during the project's duration. Recommended threshold is 70%.*

### 3.1 Observations

- WP3 facing some errors while running the cobertura tool. Setting to 50% until the job is running fine.

## 4 Functional/Integration testing

*Functional testing must cover at least the basic functionalities that the product was requested to fulfill within the INDIGO-DataCloud project scope. Integration testing must cover the interactions with other components. Both types of testing will be automated whenever feasible by integrating them in the project's continuous integration service.*

Tests currently being defined at Jenkins CI service.

### 4.1 Observations

- Pending job definition: The set of functional tests is documented here: https://github.com/arax/kitchen-occi

## 5 GitBook documentation

*Product-related documentation must be uploaded to GitBook's `indigo-dc` central repository. Types of documentation includes a) Developer b) Deployment and Administration c) Command-line Interface (CLI) and Application Program Interface (API) d) User Documentation. All these types may not be applied for every product. Those products that offer functionalities out of the scope of INDIGO-DataCloud project needs may not provide all the spectrum, but links to the official documentation.*

Documentation available under `indigo-dc` GitBook organization:

https://indigo-dc.gitbooks.io/rocci/content/

### 5.1 Types of documentation currently provided

Readme

## 6 Configuration Management

*Those products released by INDIGO-DataCloud project that need to be deployed by the end user must rely on a maintained open-source configuration management tool to provide an automated means to install and configure the product. The recommended tool is* `Ansible`.

| | |
|---|---|
| Tool | puppet |
| Deployment coverage | configuration |
| Manifest link | https://github.com/EGI-FCTF/rOCCI-server/blob/master/examples/etc/puppet/ |

### 6.1 Observations

- Puppet manifest provided for configuring rOCCI-server.

Part II

# How to read this document

## 1 Summary (front) page

Both the overall product's SQA adherence and per-task status codes are explained below:

| | |
|---|---|
| COMPLETE | Task has been successfully completed and fulfills the project's SQA requirements, listed in Deliverable D3.1 and Extensions to Software Quality Assurance documents. |
| NOT COMPLETE | Task has not been completed, yet some missing required bits have not been provided. |
| IN PROGRESS | Task has not been completed, but can proceed as it is. |
| WP3 PENDING | Task has some pending work from WP3 side, meaning that the product team already submitted the required data but it has not been yet consumed by WP3. |

## 2 Task Progress

### 2.1 Code style

| | |
|---|---|
| Code style definition | Name and link of the standard to which the product is adhered. |
| Community/de-facto standard | Whether the adopted standard is community-wide accepted. |
| Exceptions | Number of exceptions from the standard definition. |
| Richness | Number of rules defined in the adopted standard. Additionally (whenever available) the number of errors, number of warnings documented in the standard will be displayed as well as the link to the latest definition. |

### 2.2 Unit testing

This section will display the a) trend graph with the evolution of the code coverage over time and b) the Cobertura report, with the coverage results of different methods. Both are taken from the project's Jenkins continuous integration service.
*Note*: resultant coverage value is the lowest of the ones for the different methods: packages, files, classes, lines, conditionals.

## 2.3 Functional/Integration testing

## 2.4 GitBook documentation

Whenever the documentation of the product is available at the project's GitBook repository, both the a) **link** to the documentation index and b) **type of documentation** provided will be displayed in the report.

## 2.5 Configuration Management

Whenever the product has an recipe to be deployed automatically the following information will be available:

| | |
|---|---|
| **Tool** | Configuration management tool used. |
| **Manifest link** | URL pointing to the manifest/s. |
| **Deployment level** | Whether `installation`, `configuration` or both. |
| **Build status** | Current build status for the project's supported OS distributions. |