INDIGO - DataCloud

Software Quality Assurance (SQA) Report

22-26 Aug 2016

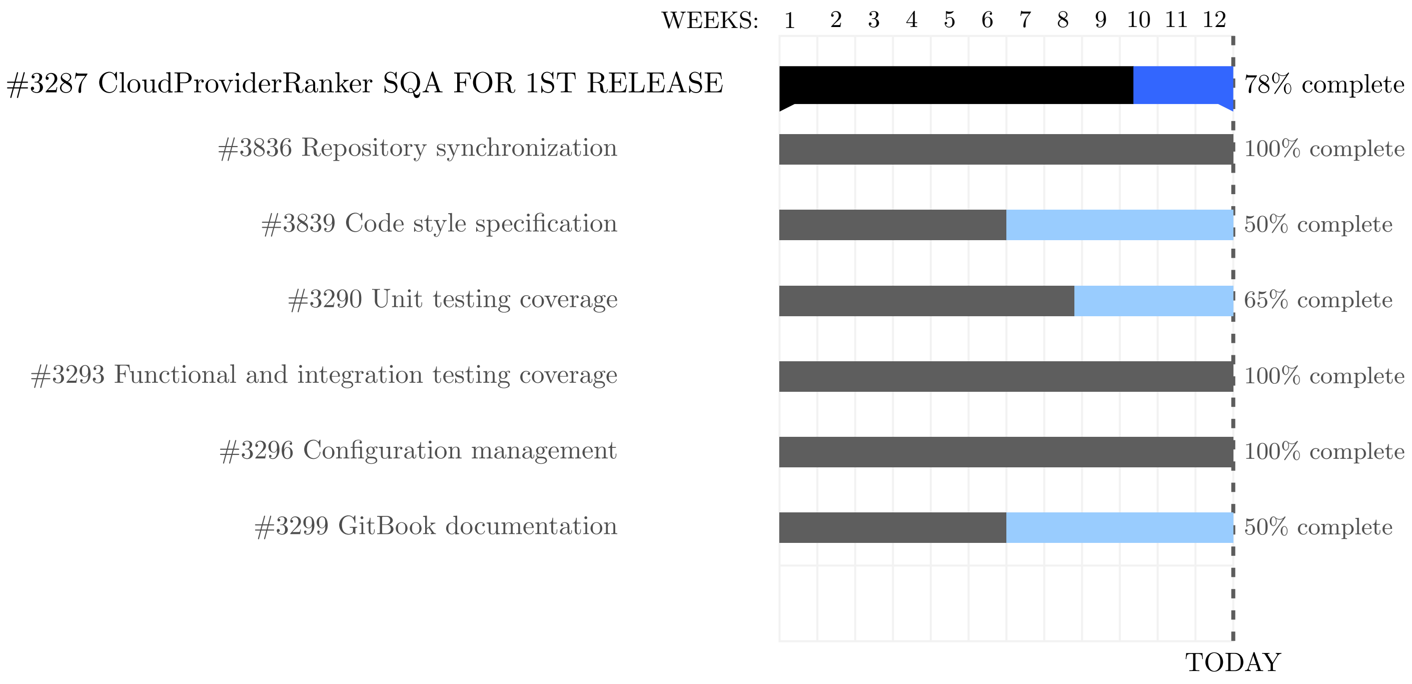
CloudProviderRanker

SQA Progress Status NOT COMPLETE

78% done

GitHub repository	COMPLETE
Code style adherence	NOT COMPLETE
Code coverage	70%
Functional/integration testing	COMPLETE
GitBook documentation	COMPLETE
Automated deployment	COMPLETE

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/CloudProviderRanker.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	Google Java Style
Community/de-facto standard	Yes
Exceptions	0
Richness	54 Errors 54 Warnings 0 link

2.1 Build status

Last build status on Jenkins CI cloudproviderranker-codestyle.

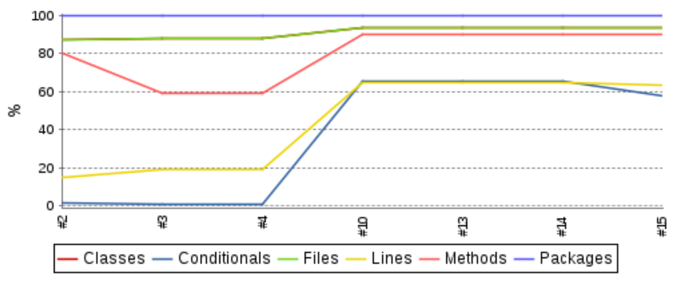
2.2 Observations

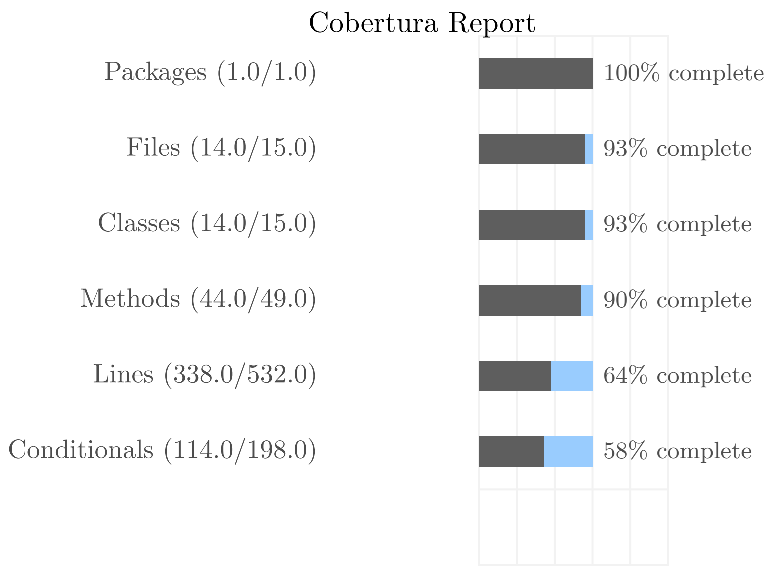
- Progress is set to 50%: code style is defined but the current code is not compliant with it.

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%.

Trend graph





3.1 Build status

Last build status on Jenkins CI cloudproviderranker-unittest.

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.

4.1 Build status

Last build status on Jenkins CI cloudproviderranker-functional.

4.2 Observations

- Jenkins job creates the JAR file for testing the functionalities documented in GitBook.
- Will get the report from https://indigo-dc.gitbooks.io/cloud-provider-ranker/content/chapter1.html

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/cloud-provider-ranker/content/

5.1 Types of documentation currently provided

Readme **Deployment/user documentation**

5.2 Observations

- Points to improve in the documentation
  - Align documentation sections to the ones defined in https://project.indigo-datacloud.eu/projects/v Documentation; in case you need to add different sections, do it at the end.
  - README
    - Explain the interactions with other INDIGO components, whether this is a microservice to be deployed together with the orchestrator?
    - Describe where to deploy it (site service, central infrastructure service serving multiple communities, or a service to be maintained by each individual user community)
  - Deployment guide
    - Building from source: add command/s to build the JAR
    - Installing from packages: add command/s to install the JAR from INDIGO repositories to INDIGO, both for CentOS7 and Ubuntu 14.04
    - Deployment using Docker: add a reference to dockerhub's indigodatacloud/cloudproviderranke repository
    - Other considerations:
      - Identify the TCP ports that need to be exposed and to which services
      - Provide information on how to configure the ranking, this should be simple to understand examples, the information in the ranking algorithm section is too dense and difficult to follow for this purpose.
      - Not clear how to implement a monitoring metric, how to obtain the monitoring information ? from where ?
      - Reference to https://goo.gl/GZnl8 in ranking algorithm, please put this information inside the help (other subsection, annex) and not as external document

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`.

6.1 Observations

- Configuration for CloudProviderRanker server is trivial: zero configuration, or NO configuration at all (puppet or other). It can be launched without any configuration but the editing of two support files in json notation that define weights and priorities used by the internal engine rule.

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.