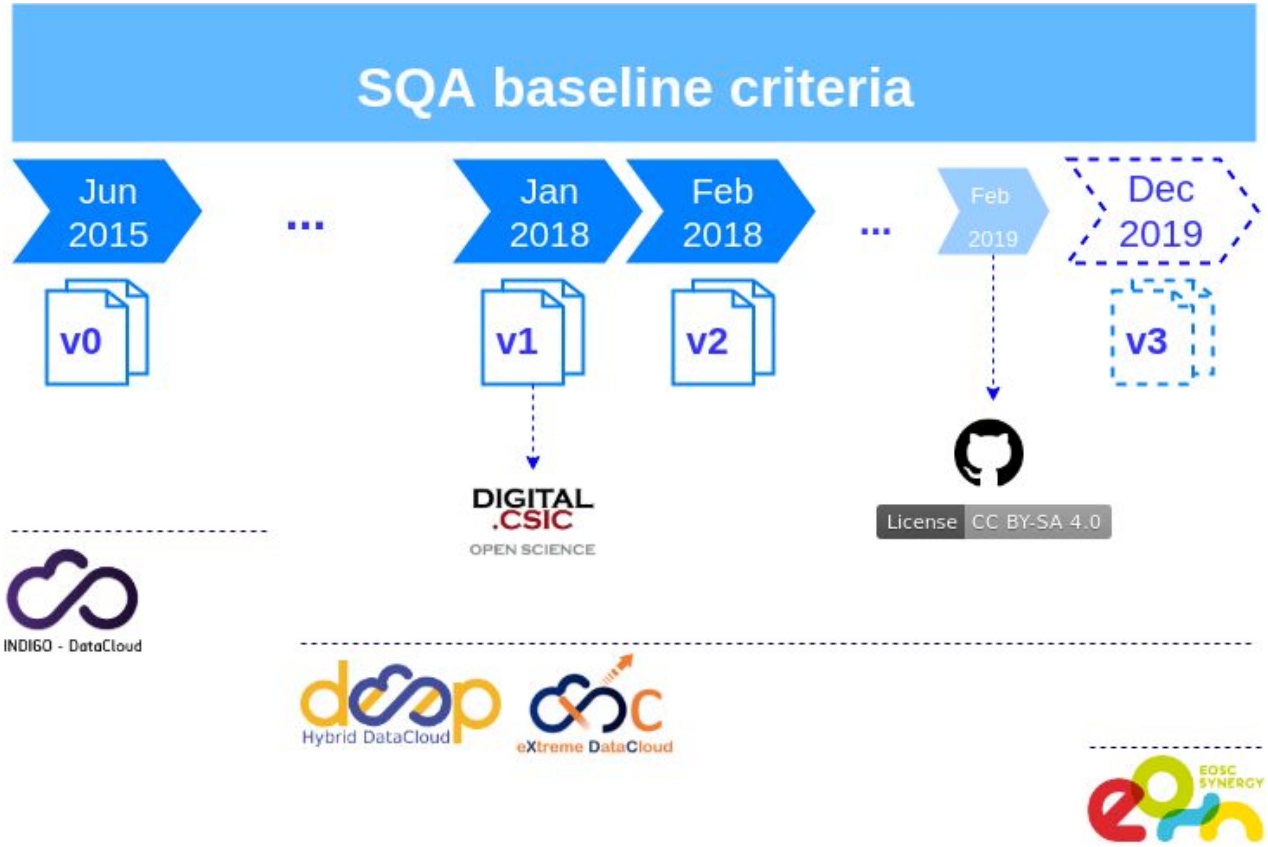
Baseline criteria for achieving software quality within the European research ecosystem

Pablo Orviz (CSIC)
Mario David (LIP)
Cristina Duma (INFN)

Motivation

- Filling an uncovered gap in European research software engineering ecosystem (origin → 2015)
- Drive the adequate development, timely delivery and reliable operation of software assets (origin → collaboration-driven sw development projects)
- Pragmatic approach, schematic content
- Act at the early stages of software lifecycle
    - Focus on source code verification
- Build a knowledge-base for reference
    - Starting point for any software development effort in research

Timeline

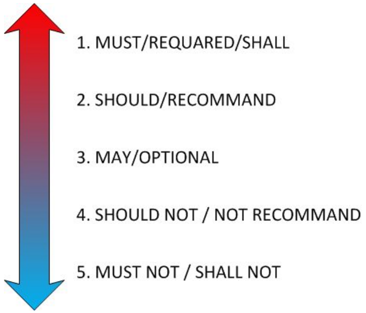


Key dates

- [Jun 2015] INDIGO WP3’s 1st deliverable sets the foundations (***v0***)
- [Jan 2018] Publication on CSIC national repository (***v1***)
- [Feb 2019] Open to contributions
- ***v3*** is on the way

Features

- **RFC 2119** compliant
  - MUST → *Requirements*
  - SHOULD/SHALL/RECOMMENDED, .. → *Best practices*
- Enumeration-like structure, grouped in **categories**:
  - *Code Accessibility & Code Workflow*
  - *Licensing*
  - *Code Style & Code Testing (unit, functional)*
  - *Code Review*
  - *Documentation*
  - *System, Integration & Security Testing*
  - *Automated Deployment*
- **Schematic content**, (future) presence of Annexes to elaborate on topics
- Alignment with **software engineering** methodologies
  - *DevOps culture*
  - *Agile manifesto*



1. MUST/REQUARED/SHALL

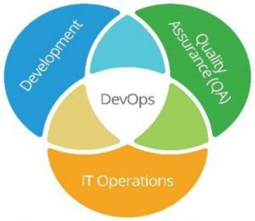
2. SHOULD/RECOMMAND

3. MAY/OPTIONAL

4. SHOULD NOT / NOT RECOMMAND

5. MUST NOT / SHALL NOT

https://indigo-dc.github.io/sqa-baseline/



Quality Basics

- Source code
    - *Open & Accessible* through adequate licensing and code hosting platforms
    - *Change-based* approach to source code management by means of VCSs
        - *Branching* model for addition of features/bug fixes
            - Separate dev from production
        - *Static analysis testing*: style compliance, unit and functional testing, vulnerability scanning..
        - *Code review*
            - Meaningful test cases
- Documentation
    - Use of *markup languages*
    - *Treated as code* (versioned) → code review
- Dynamic analysis testing
    - Security assessment
    - Integration testing
- Automated deployment

Related work

Comparison with similar initiatives/efforts in academic research

- ⇈ **pragmatic approach**
  - *Software Sustainability Institute* https://www.software.ac.uk/
  - *CESSDA Software Maturity Levels* https://doi.org/10.5281/zenodo.2614050
  - *CLARIAH Guidelines for Software Quality* https://github.com/CLARIAH/software-quality-guidelines
- ⇊ **didactic orientation**
  - *The Carpentries* https://carpentries.org/
- ⇈ **management of collaborative developments**
  - none ??

THE CARPENTRIES

Software Sustainability Institute

cessda
Consortium of European
Social Sciences Data Archives

Target audiences

- Collaboration-driven software development
  - Initial scope of application
  - Specific guidelines for collaborative projects
    - Common source code & documentation repositories (organizations)
    - Compatible licenses
    - (in v3) Software release management
- Research software engineers (RSEs)
  - Most best practices addressed to software lifecycle of individual products
    - *Applicable to any software development effort*
  - Pragmatic approach
    - *Suitable to both high and low-skilled developers*

Openness

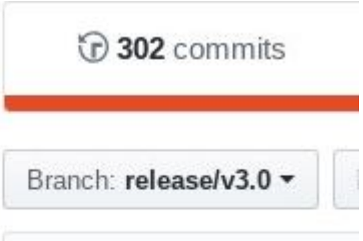
- Opening up the SQA criteria..
    - Grew out of an individual project, Sustained by wider collaboration
        - Best chance to evolve and survive
    - Share practical experience
    - Promote discussion on new topics, Improve/consolidate existing topics
- Social coding..
    - Found at GitHub: **https://github.com/indigo-dc/sqa-baseline**
        - Markdown format (contribution-friendly)
    - Automatic online rendering: **https://indigo-dc.github.io/sqa-baseline/**
        - Last version immediately available

Contribute

How to collaborate?

Through GitHub issues and Pull Requests..

- 3 types of issues:
  - **Typo** report
  - **Inconsistency** report
  - **Enhancement** request
- Typos and Inconsistencies are (usually) considered emergency fixes
  - Merged immediately in production version
- Next release is maintained in a separate branch
  - Branch ID: ***release/v****<next-version-number>*
  - (usually) contains only Enhancements

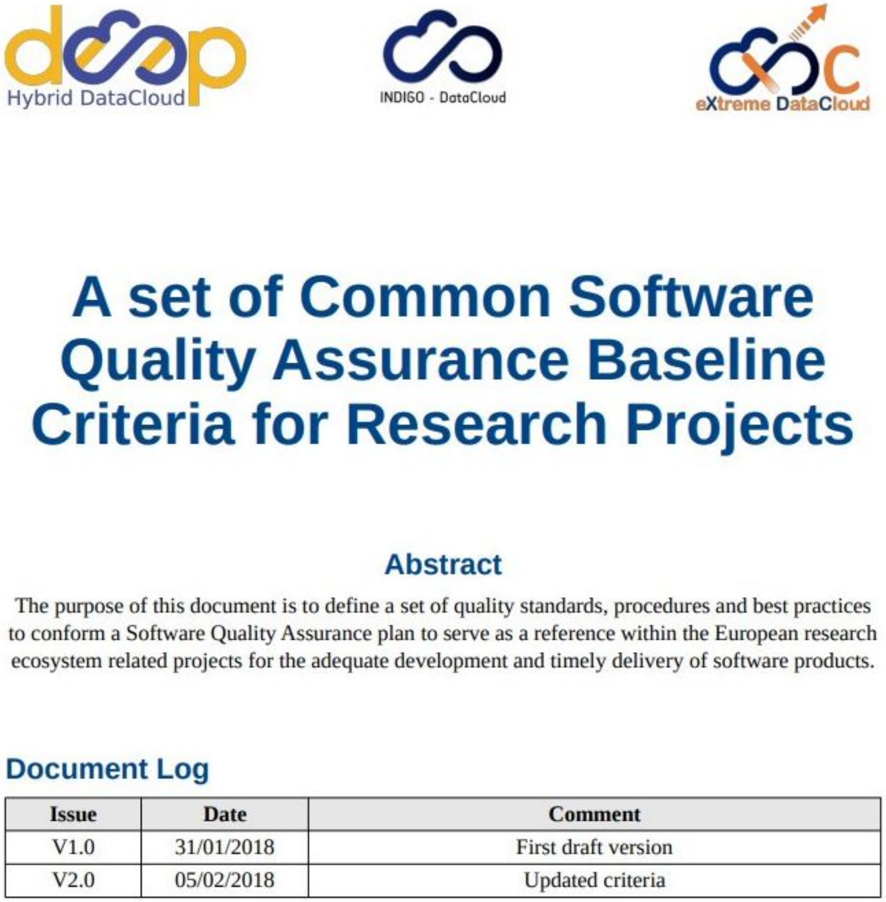


Support

How to request support?

Through GitHub issues..

- Issue type
    - **Consultancy** service



A set of Common Software Quality Assurance Baseline Criteria for Research Projects

Abstract

The purpose of this document is to define a set of quality standards, procedures and best practices to conform a Software Quality Assurance plan to serve as a reference within the European research ecosystem related projects for the adequate development and timely delivery of software products.

Document Log

Issue	Date	Comment
V1.0	31/01/2018	First draft version
V2.0	05/02/2018	Updated criteria

Gracias!

Obrigado!