Quality Assurance Models in the framework of EOSC-Synergy

Ibergrid 2022

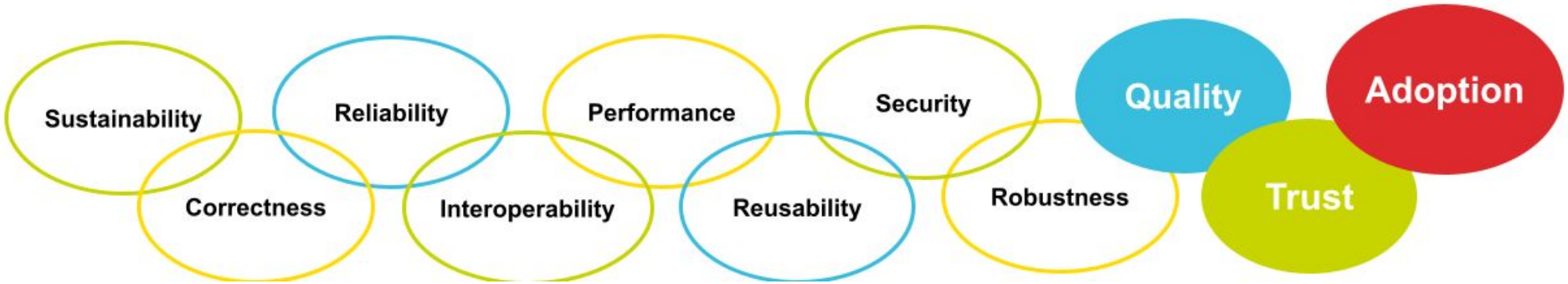
Mario David - <david@lip.pt>
Pablo Orviz - <orviz@ifca.unican.es>
Samuel Bernardo - <samuel@lip.pt>
Jorge Gomes - <jorge@lip.pt>
Isabel Campos - <isabel.campos@csic.es>
João Pina - <jpina@lip.pt>

EOSC-SYNERGY receives funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 857647

1 - Introduction

Importance of Software and Service Quality assessment in Research

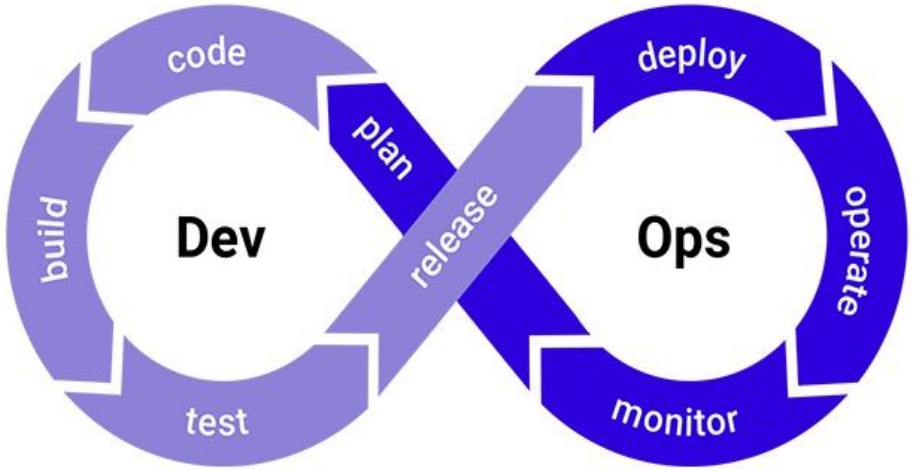
- Quality assessment is an important trait for software and for services.
- It allows users and managers to have higher trust on the Software and Services during their use and operation:
  - They expect that the software and related services will work as supposed.
  - Give the expected results and meet their requirements.
- It also contributes to the maintainability, stability and sustainability of the software and services.
- It contributes to facilitating the collaboration between software developers and promotes good practices for Software development.
- It promotes good practices for service development and operation.



Software Development methodologies: DevOps methodology

DevOps is:

- Is a set of **good practices**.
- Supported by **Continuous Integration/Continuous Delivery/Continuous Deployment - CI/CD(D)**.
- Enhances collaboration between the departments or groups.



It's characterized by the following high level phases:

- Development: *plan* ↦ *design* and *code*.
- Quality assurance: *build*, *test*, *release* and *deployment* ↦ CI/CD(D).
- Operations.

Short periods between phases ↦ Fast + Automation.

Quality Models - I

Software product quality models assess the quality properties of software products:

- A quality model is a set of Quality properties, or Quality Criteria that can be assessed for a given Software product.

The most relevant quality model:

- Standard defined in the ISO/IEC 25010:2011(en) [https://www.iso.org/standard/35733.html]
  - Systems and software engineering, denoted: "Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models".
- It replaces the ISO/IEC 9126-1:2001.

Quality models in the framework of European projects

[https://eosc-portal.eu/providers-documentation/eosc-provider-portal-resource-maturity-classification]:

- Based on Maturity Levels of the Software or service.
- The EOSC guide proposes characteristics to help assess the maturity of a service via the operational definition of the Technology Readiness Level (TRL) indicators:
  - TRL 7 - Beta: "System prototype demonstration in operational environment".
  - TRL 8 - Production: "System complete and qualified"
  - TRL 9 - Production: "Actual system proven in operational environment"

Quality Models - II

CESSDA's Software Maturity Model (SMM) [DOI: 10.5281/zenodo.2591055]:

- Approach for assessing the maturity of the components of the technical Research Infrastructure (RI).
- The Software Maturity grade is based on the Reuse Readiness Levels (RRLs), as developed by NASA Earth Science Data Systems.
- Each criteria is graded with 5 levels.
- The criteria are about: Documentation, Intellectual Property, Extensibility, Modularity, Packaging, Portability, Standards Compliance, Support, Verification and Testing, Security, Internationalization and Localization, Authentication and Authorization.

DevOps approach [*Humble, J. and Farley, D. (2011), Continuous delivery: Reliable software releases through build, test, and deployment automation*]:

- Links development and operations for software components, through the use of a Continuous Integration and Continuous Delivery pipeline.
- CAMS model; Culture, Automation, Measurement and Sharing.
- It does not have a single standard, but takes the best practices from several standards.
- Metrics are based on several sources and are posed as questions.
- These questions/metrics focus on the number of features delivered, the time a feature needs to be delivered or the number of releases to deliver these features.
- Map most of the metrics with the Product Quality model of the ISO/IEC 25010:2011.

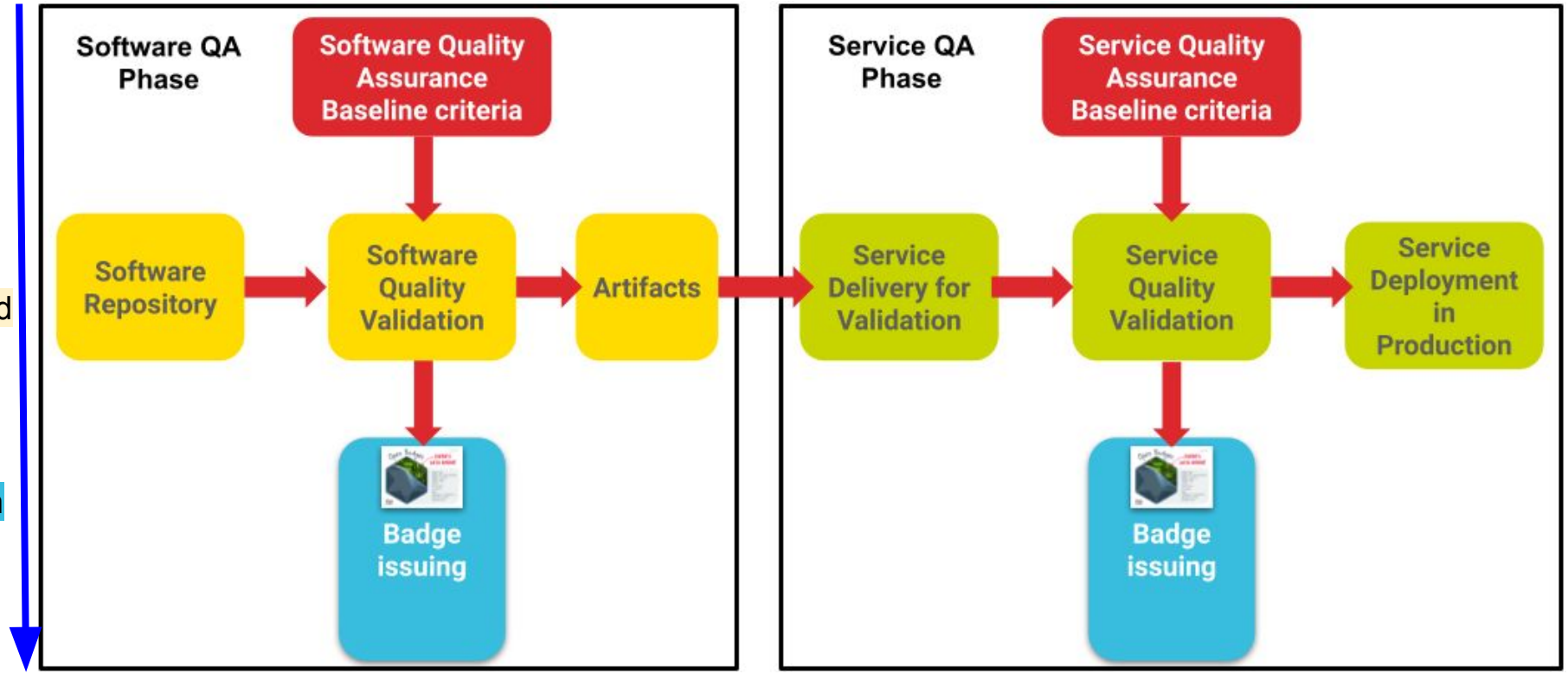
Vision and high level architecture

The vertical view represent the high level architecture of the Quality Assurance developed in EOSC-Synergy.

1. Quality Assurance baseline documents: Detail a set abstract quality metrics.

2. SQA as a Service (SQAaaS): Implement Quality assessment based on the baselines documents.

3. Issue badges: as a proof that a given Software or Service has passed the Quality Assurance criteria.

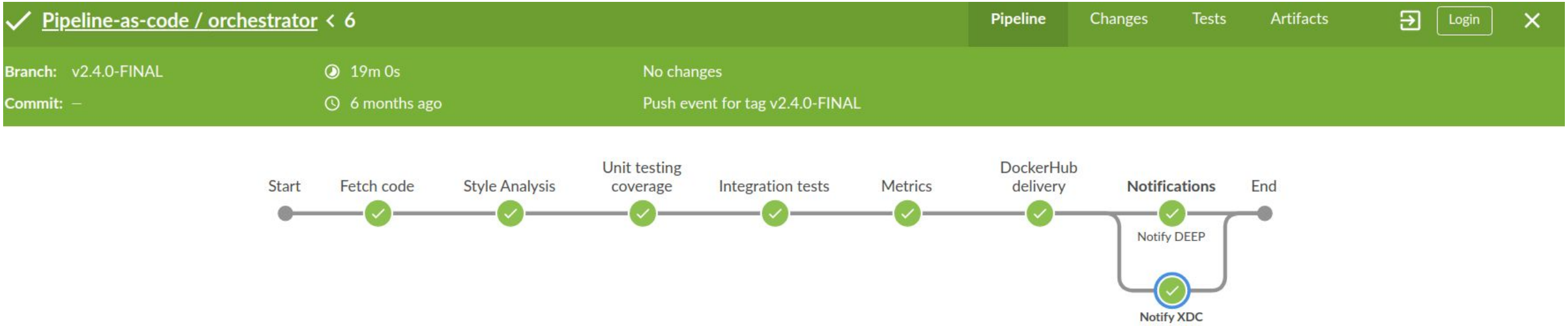


Continuous Integration/Continuous Delivery/Continuous Deployment - CI/CD(D)

- CI ↦ Continuous Integration:
  - Coding.
  - Building: includes Automation ↦ produces artifacts.
  - Testing: includes Automation and SW Quality Assurance (of produced artifacts).
- Code Review: manual step, comments/approval/voting by partners/colleagues/contributors.
- CD ↦ Continuous Delivery ↦ Deployment:
  - Delivery: Artifacts are released - ready for usage (by users or system administrators) in production.
  - Deployment: refers mainly to services ↦ Installation, configuration, service (re)start.

CI/CD(D): Example

A real example of a Jenkins CI/CD pipeline

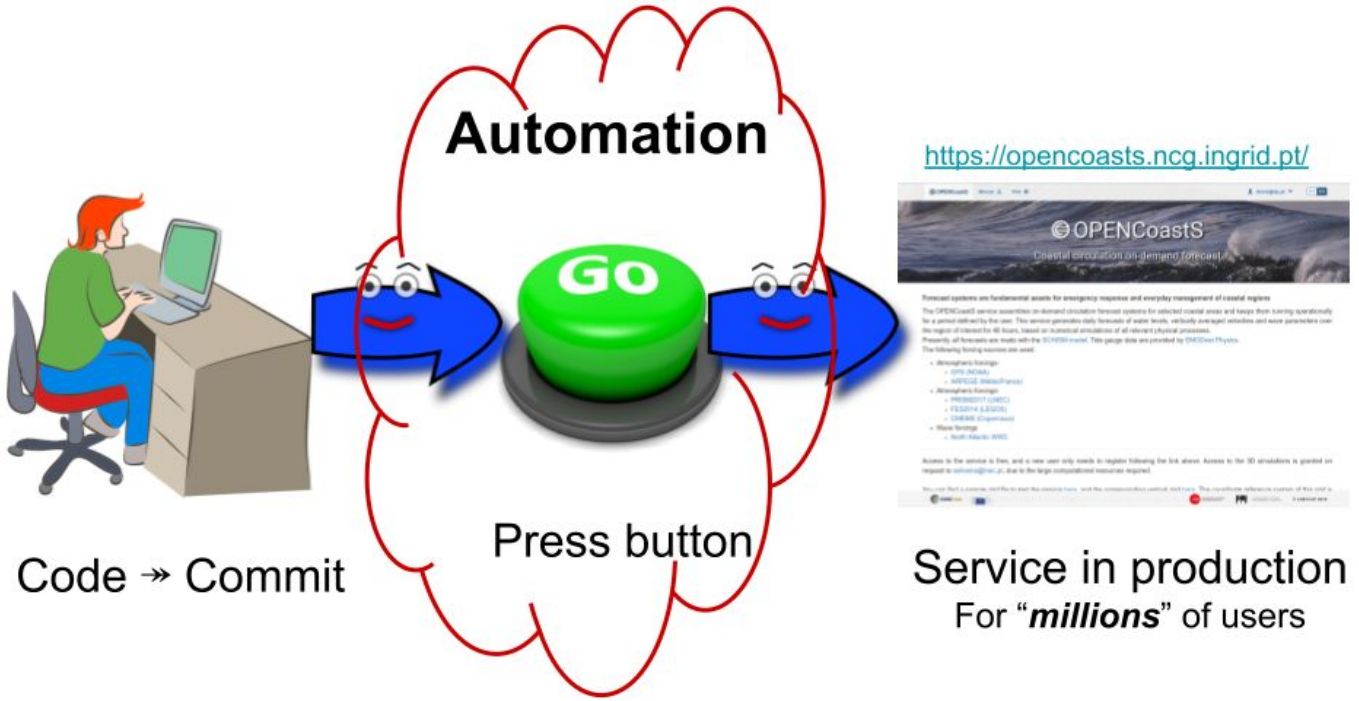


DevOps and automation

Scary but in the real world/practice not *all* steps or phases are automatic or automated, for example:

- "Code Review" is a manual step.
- In many cases - the process is stopped in the "Delivery" step.
- In many cases - the automated deployment is performed in a "Staging" or "Pre-production" or
- Preview" infrastructure, (not on the production service/infrastructure).

Vision when considering DevOps and automation.

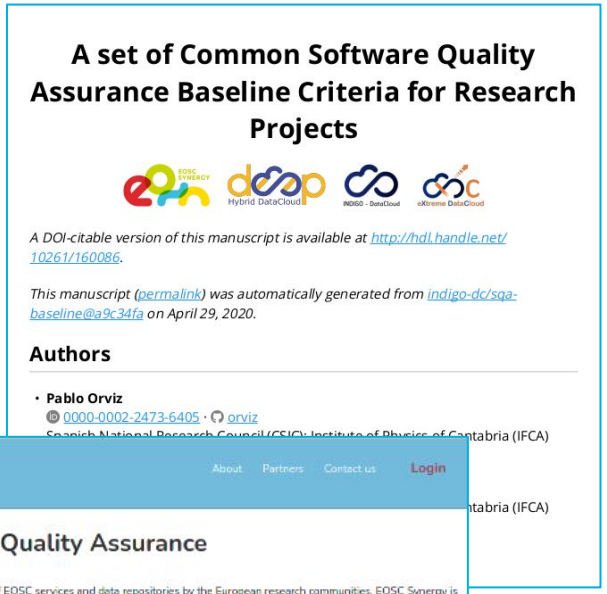


2 - SQA

The SW Quality Assurance baseline

The SW Quality Assurance applies to Software source code best practices and procedures.

- The SQA baseline is a set of abstract criteria that should be applied to the process of SW development.
- Each criterion has a severity according to the keywords defined in RFC2119:
  - **"MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL"**.
- Each criterion is binary, with a value of "0" or "1" such that when implemented in practice Software Quality Assurance as a Service [https://sqaaas.eosc-synergy.eu/], it "**passes**" or it "**does not pass**" that criterion.

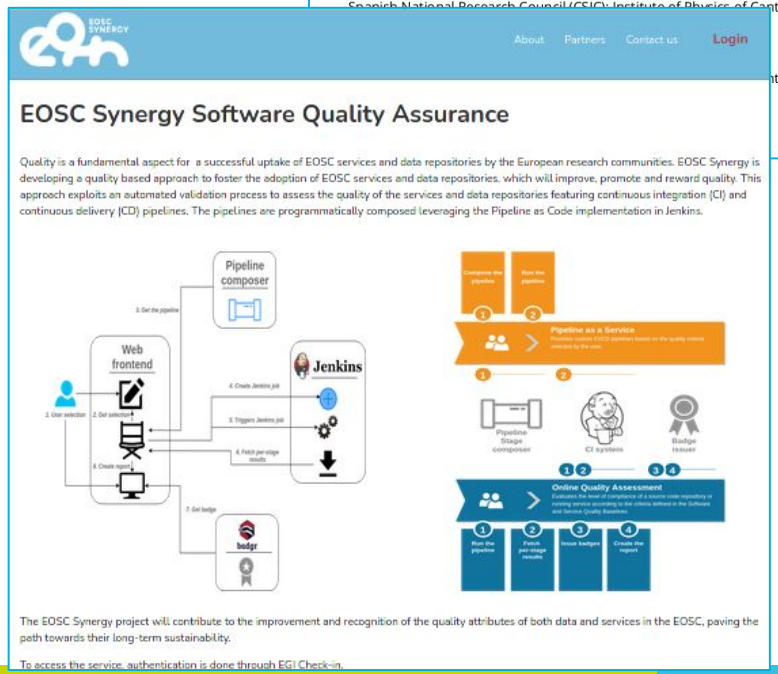

A set of Common Software Quality Assurance Baseline Criteria for Research Projects

A DOI-citable version of this manuscript is available at http://hdl.handle.net/10261/160086.

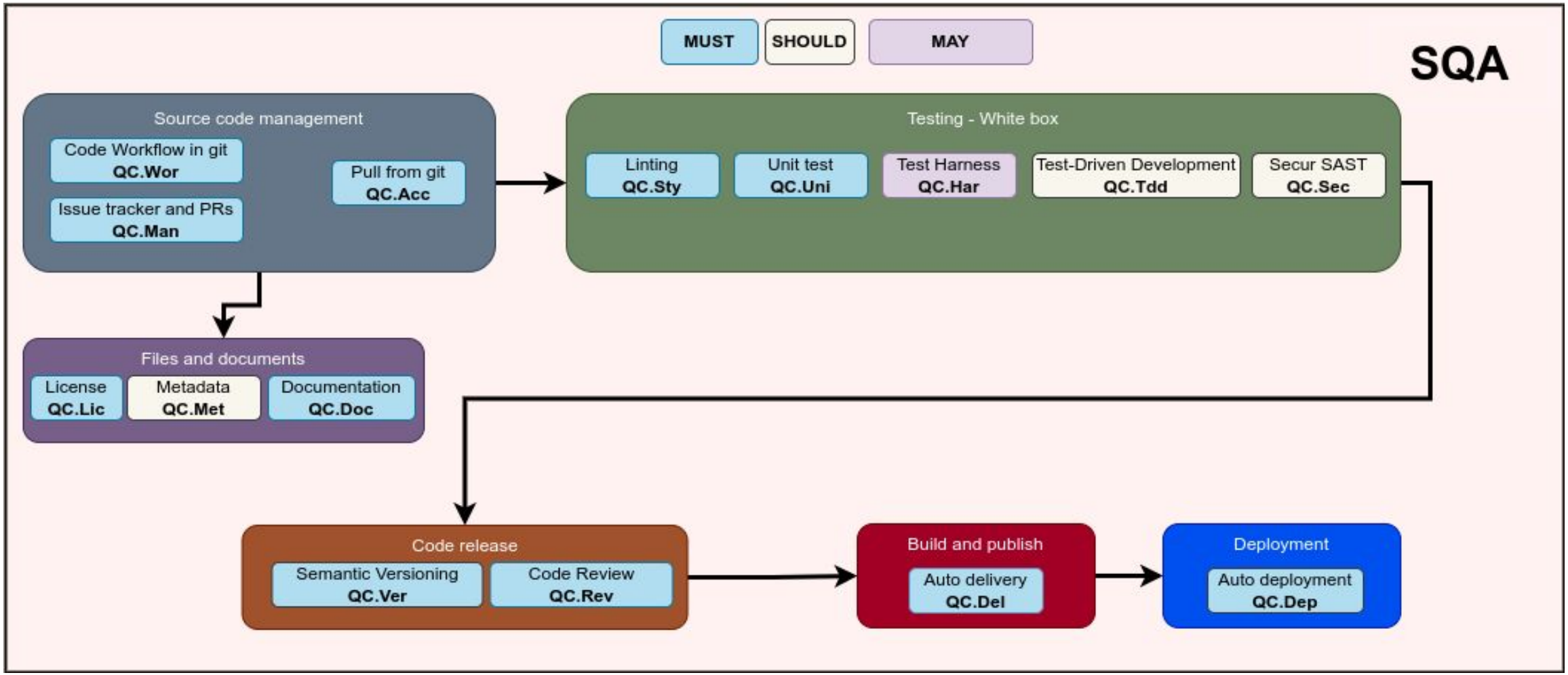
This manuscript (permalink) was automatically generated from indigo-dc/sqa-baseline@a9c34fa on April 29, 2020.

Authors

Pablo Orviz
0000-0002-2473-6405 · orviz



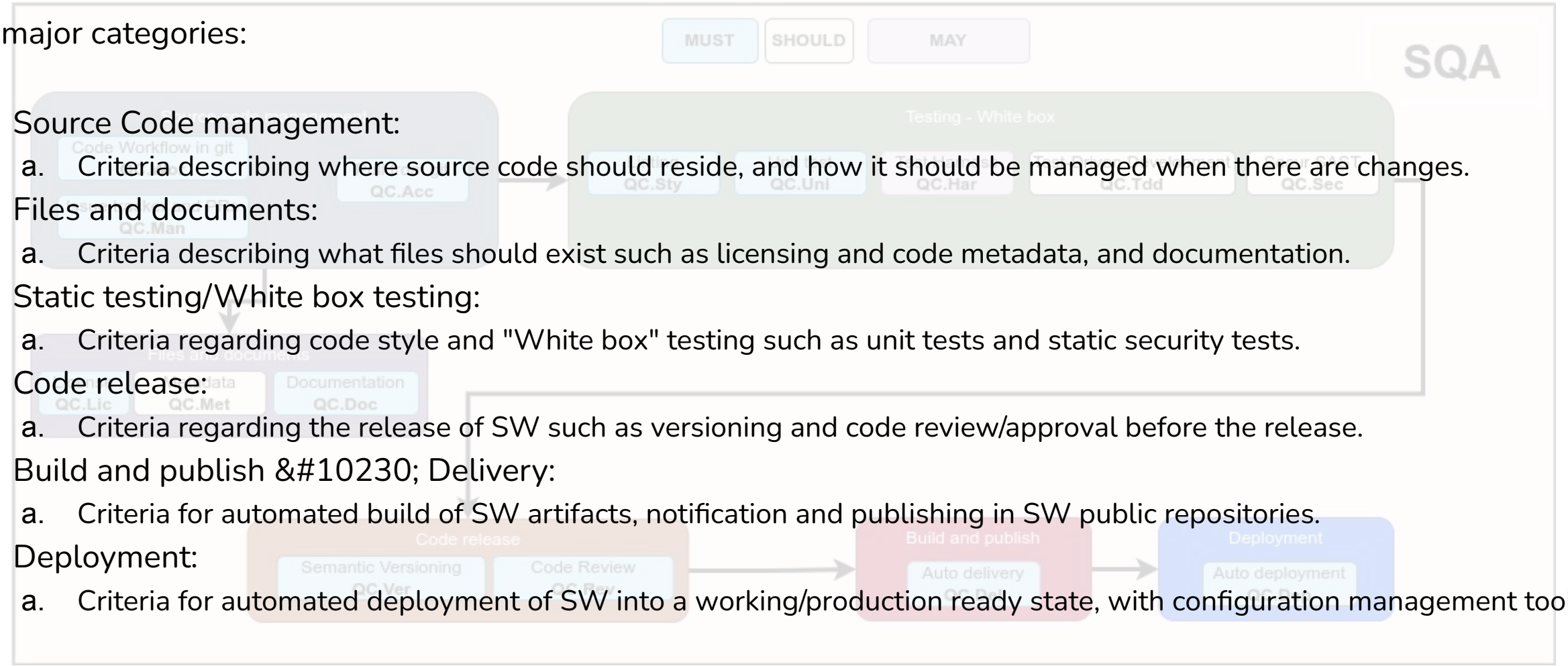
Quality model



Quality model

Six major categories:

1. Source Code management:
   a. Criteria describing where source code should reside, and how it should be managed when there are changes.
2. Files and documents:
   a. Criteria describing what files should exist such as licensing and code metadata, and documentation.
3. Static testing/White box testing:
   a. Criteria regarding code style and "White box" testing such as unit tests and static security tests.
4. Code release:
   a. Criteria regarding the release of SW such as versioning and code review/approval before the release.
5. Build and publish ⟶ Delivery:
   a. Criteria for automated build of SW artifacts, notification and publishing in SW public repositories.
6. Deployment:
   a. Criteria for automated deployment of SW into a working/production ready state, with configuration management tools.



Awarding badges according to criteria passed

EOSC-Synergy approach defines three different badge classes: bronze, silver and gold.

Mapping between quality criteria and EOSC-Synergy digital badges

	Software Classification			
Quality Criteria	QC Code	Quality Badges		
		Gold	Silver	Bronze
Code Accessibility	QC.Acc			
Licensing	QC.Lic			
Code Style	QC.Sty			
Code metadata	QC.Met			
Unit Testing	QC.Uni			
Documentation	QC.Doc			
Security	QC.Sec			
Code Workflow	QC.Wor			
Semantic Versioning	QC.Ver			
Code Management	QC.Man			
Automated Delivery	QC.Del			

3 - ServiceQA

Contextualization of a Service

Service, represents the following:

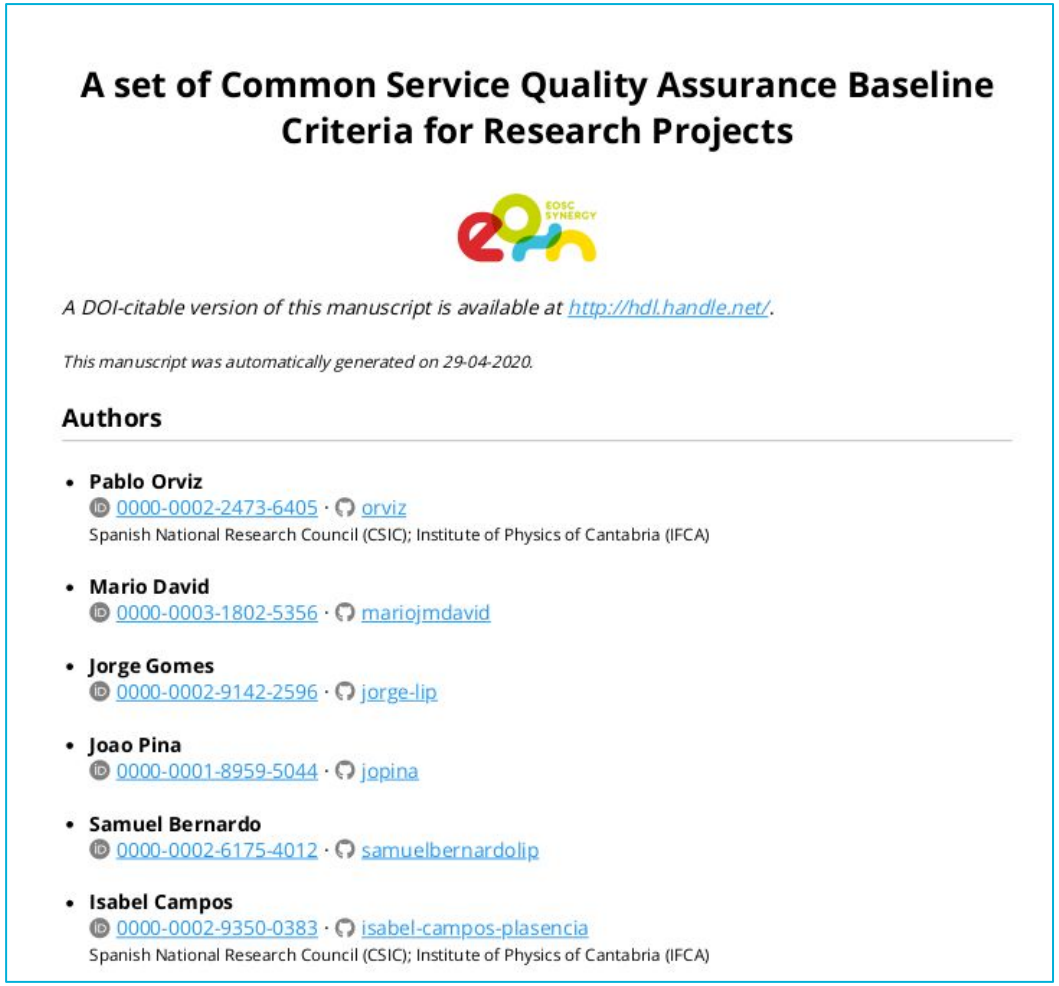
- **Web Service** [https://techterms.com/definition/web_service]:
  - A **Web Service** is an application or data source that is accessible via a standard web protocol (HTTP or HTTPS).
  - **Web Services** are designed to communicate with other programs, rather than directly with users.
  - Most **Web Services** provide an API, or a set of functions and commands, that can be used to access the data.
- **Web Application** [https://techterms.com/definition/web_application]:
  - A **Web Application** or **Web App** is a software program that is delivered over the Internet and is accessed through a web browser.
- **Platform / Service Composition** [https://csrc.nist.gov/glossary/term/Service_Composition]:
  - Aggregation of multiple small services into larger services, according to a service-oriented (SOA) and/or microservices architecture.
  - An integrated set of **Web Services**, **Web Applications** and software components.

Examples are: Web portals, Scientific portals and gateways, data repositories.

The Service Quality Assurance baseline

The Service Quality Assurance applies to Service development and operation best practices and procedures.

- The ServiceQA baseline is a set of abstract criteria that should be applied to the process of Service development deployment and operation.
- Criteria properties are the same as for the SQA baseline.
- Criteria can be assessed/verified by the SQAaaS.
- These criteria complements the criteria described in the "Software Quality Assurance baseline"



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

A DOI-citable version of this manuscript is available at http://hdl.handle.net/.

This manuscript was automatically generated on 29-04-2020.

Authors

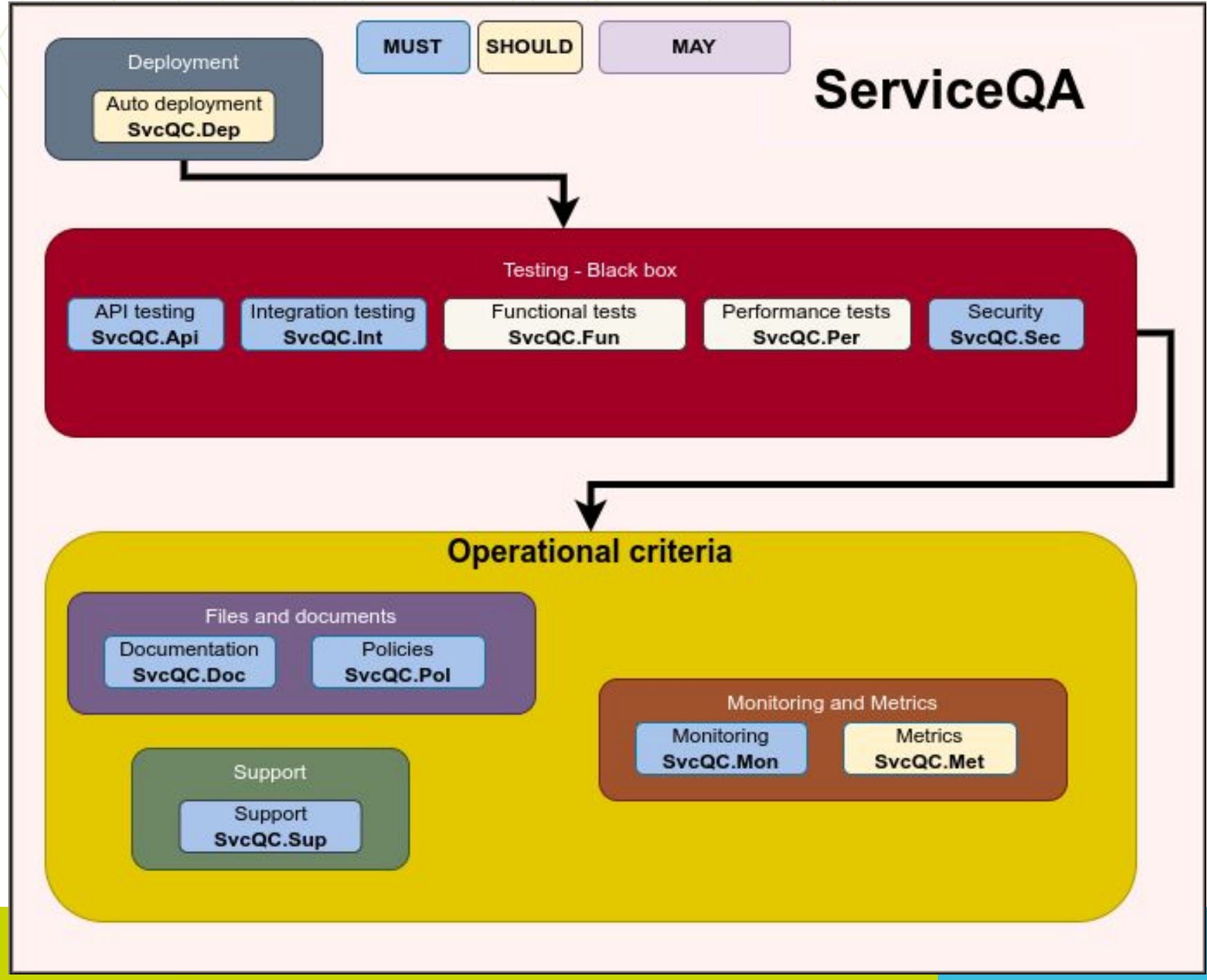
- **Pablo Orviz**
  0000-0002-2473-6405 · orviz
  Spanish National Research Council (CSIC); Institute of Physics of Cantabria (IFCA)
- **Mario David**
  0000-0003-1802-5356 · mariojmdavid
- **Jorge Gomes**
  0000-0002-9142-2596 · jorge-lip
- **Joao Pina**
  0000-0001-8959-5044 · jopina
- **Samuel Bernardo**
  0000-0002-6175-4012 · samuelbernardolip
- **Isabel Campos**
  0000-0002-9350-0383 · isabel-campos-plasencia
  Spanish National Research Council (CSIC); Institute of Physics of Cantabria (IFCA)

Goals

The baseline, harnesses the capabilities of the quality factors in the underlying software to lay out the principles for attaining quality in the enabled services. The Service Quality baseline aims at fulfilling the following goals:

1. Complement with a DevOps approach the existing approaches to assess and assure the quality and maturity of services.
2. Build trust on the users by strengthening the reliability and stability of the services, with a focus on the underlying software, thus ensuring a proper realization of the verification and validation processes.
3. Ensure the functional suitability of the service by promoting testing techniques that check the compliance with the user requirements.
4. Improve the usability by identifying the set of criteria that fosters the service adoption.
5. Promote the automated validation of the service quality criteria.

Quality model



Quality model - I

The criteria are organized in major categories as follows:

Automated:

1. Deployment:
    a. Infrastructure as Code (IaC) for deployment and configuration → Criteria for Automated Deployment.
2. Dynamic testing - Black box testing
    a. Criteria regarding code style and "Black box" testing such as; API Testing, Integration Testing, Functional tests, Performance tests, Security.

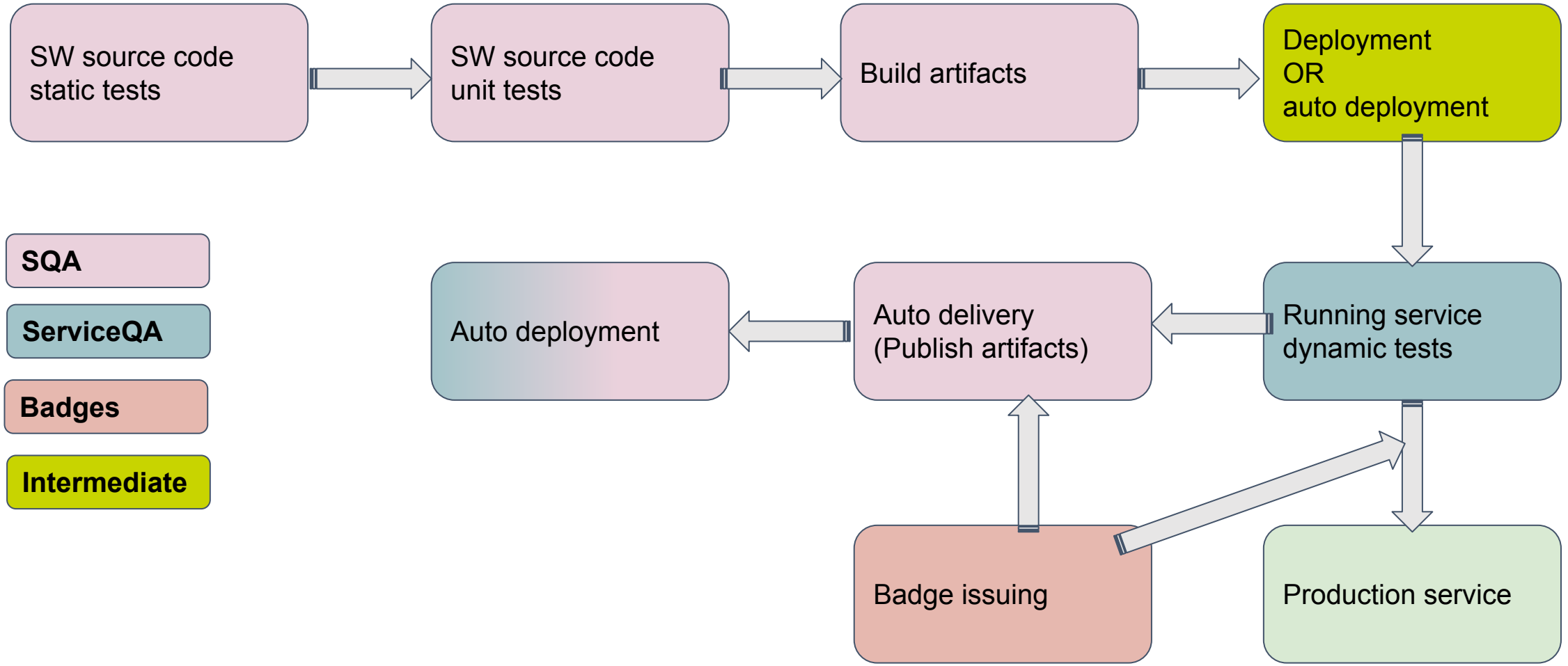
Quality model - II

The criteria are organized in major categories as follows:

Operational:

1. Files and documents:
    a. Criteria regarding documentation of the Service and Policies.
2. Support:
    a. Criteria about the existence of support service for issue tracking, communication with users, reporting of problems.
3. Monitoring and Metrics:
    a. Criteria regarding monitoring is particularly important for a service in production, it can include the execution of some or all tests from Blackbox testing. While metric collection is important for reporting the usage of the service including accounting of resources.

Integrating SQA and Service QA: With open-source code

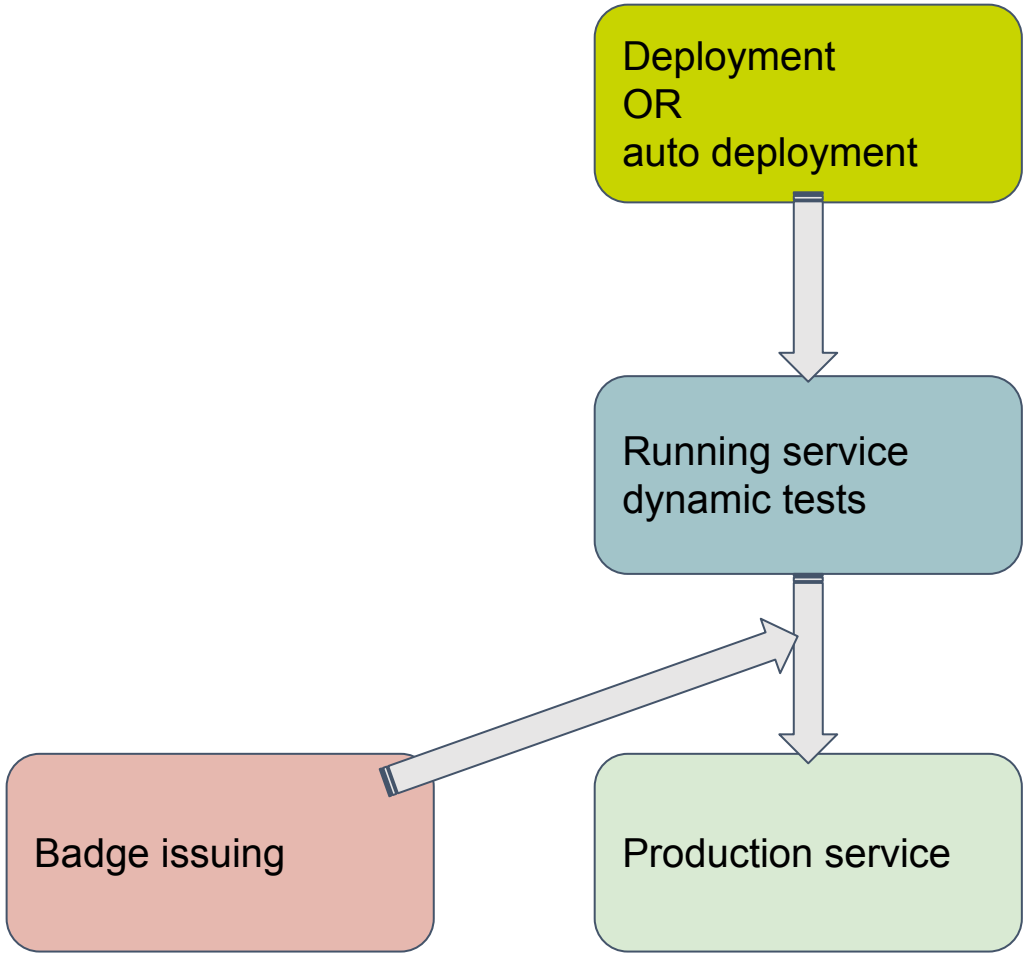


Integrating SQA and Service QA: With Commercial/Closed source SW

ServiceQA

Badges

Intermediate



Awarding badges according to criteria passed

Awarding badges according to criteria passed

	Service Classification			
Quality Criteria	Criteria code	Quality Badges		
		Gold	Silver	Bronze
Automated Deployment	**SvcQC.Dep**			
API Testing	**SvcQC.Api**			
Integration Testing	**SvcQC.Int**			
Functional tests	**SvcQC.Fun**			
Performance tests	**SvcQC.Per**			
Security tests	**SvcQC.Sec**			
Documentation	**SvcQC.Doc**			

Where are the QA baselines

SQA:

- https://github.com/indigo-dc/sqa-baseline
- Latest: v4.0
- Contribute:
  - https://github.com/indigo-dc/sqa-baseline/blob/master/CONTRIBUTING.md
- URL: http://hdl.handle.net/10261/160086
- DOI: 10.20350/digitalCSIC/12543

ServiceQA:

- https://github.com/EOSC-synergy/service-qa-baseline
- Latest: v2.0
- Contribute:
  - https://github.com/EOSC-synergy/service-qa-baseline/blob/master/CONTRIBUTING.md
- URI: http://hdl.handle.net/10261/214441
- DOI: 10.20350/digitalCSIC/12533

Gracias !

Obrigado !

Danke !

Dziękuję !

Vďaka !

Dekuji !

Bedankt !

Merci !

Thanks !

Coming next: Software Quality Assurance