Initial plan for Quality Assurance and data FAIRness

Deliverable ID	D 7.1- Initial plan for Quality Assurance and data FAIRness
Work Package Reference	WP7
Issue	1.1
Due Date of Deliverable	30/11/2022
Submission Date	09/12/2022
Dissemination Level[1]	[PU]
Lead Partner	LIP
Contributors	CSIC, IISAS, UPV
Grant Agreement No	**101058593**
Call ID	**HORIZON-INFRA-2021-EOSC-01-04**

Funded by the European Union

[1] **PU** = Public, **PP** = Restricted to other programme participants (including the Commission Services),
RE = Restricted to a group specified by the consortium (including the Commission Services),
CO = Confidential, only for members of the consortium (including the Commission Services)

Prepared by	Reviewed by	Approved by
Mario David	Alvaro Lopez, Valentin Kozlov	PMB

Issue	Date	Description	Author(s)
v0.1	2nd Nov. 2022	ToC	Mario David
v0.2	18th Nov. 2022	Content update (section 3)	Mario David
v0.3	5th Dec. 2022	Sections 3, 4 and 5	Mario David, Viet Tran
v0.4	5th Dec. 2022	Sections 1 and 6	Mario David
v0.5	6th Dec. 2022	Review	Alvaro Lopez, Valentin Kozlov
v1.0	7th Dec. 2022	Implement comments, additional review	Mario David
v1.1	9th Dec. 2022	Final version, small fixes	Mario David

Table of contents

1. Introduction **4**

1.1. Work Package organization 4

2. Quality Assurance and FAIRness of data **5**

2.1. Minimal set of criteria to be assessed by SW and Services in AI4EOSC 5

2.2. FAIR data indicators 6

3. Services for QA verification and FAIR assessment **7**

3.1. Services for QA 7

3.2. CI/CD pipelines for AI/ML/DL models 10

3.2. FAIR assessment 10

4. Infrastructure resources for development **11**

4.1. Computing resource providers 11

4.2. AAI integration 11

4.3. Infrastructure tools and services 11

4.4. Infrastructure for software development and integration 12

5. Software release process and procedures **12**

5.1. Introduction 12

5.2. SW release process 13

5.3. Testbeds for development and users preview 14

5.4. Services for the SW release process 15

6. Final Remarks **15**

References **15**

Glossary **17**

1. Introduction

This deliverable describes the initial plan that will guide the software and service quality assurance towards its release and management (Task 7.1). The automatic FAIR assessment of the different digital objects produced by the project, e.g. data, datasets, metadata and AI/ML/DL models (Task 7.2). It also describes the Cloud resources providers that are partners of the projects, and will be responsible for the testbed's deployment and management with the services composing the AI4EOSC platform (Task 7.3). In this regard two different deployments are planned, one for development and another for preview and integration testing with external services.

The design and architecture of the AI4EOSC platform is currently underway by WP3, and will drive the implementation and deployment at the level of the preview and integration testbed by WP7.

The services developed by WP4 and WP5 should follow the Software and Service Quality Assurance (SQA and SvcQA) criteria and CI/CD services proposed and operated by WP7. Moreover, those software components and services are first deployed and tested in the development testbed and after, in the preview and integration testbed.

The AI/ML/DL applications developed within the use cases in WP6 as well as external use cases, can be deployed and tested in the preview and integration testbed, while the models, metadata and data will use the WP7 infrastructure for FAIR automatic assessment.

Regarding the FAIR data indicators, we will adopt the RDA "FAIR Data Maturity Model. Specification and Guidelines" [R7]. The RDA FAIR Data Maturity Model, defines 7 indicators for Findability (F), 12 for Accessibility (A), 12 for Interoperability (I) and 9 for Reusability (R), pertaining to both data and metadata. Each indicator is defined with one of three levels of importance: Essential, Important and Useful.

The preview testbed can be used by WP2 for the dissemination of the project's results, tutorials and demonstrators.

Therefore, WP7 has direct collaboration and relations with all WPs of the project, as such it plans to participate in the discussions and decisions of the other WPs.

1.1. Work Package organization

WP7 has four partners, three of them are Cloud resource providers integrated into the EGI Fedcloud infrastructure (CSIC, LIP and IISAS) and the fourth is UPV, the main developer of the Infrastructure Manager (IM) and of the SQAaaS web dashboard.

WP7 plans to organize monthly video conferences, since the IaaS resources are already in place and part of production infrastructures and some of the AI4EOSC services that are an evolution from the DEEP project, are being deployed. More frequent meetings will be organized as the need arises.

As described in the AI4EOSC deliverable D1.1 - Project handbook, a mailing list for WP7 (https://listas.csic.es/wws/subscribe/ai4eosc-wp7) is already created and used for all internal communications.

Regarding WP7 management, we will use Confluence for Wiki pages, documentation and information for other partners and WPs. Jira will be used to track the software components releases.

2. Quality Assurance and FAIRness of data

The document "A set of Common Software Quality Assurance Baseline Criteria for Research Projects" [R5] (from here on referred to as SQA baseline), details the quality criteria and best practices deemed important in the software development phase of components within the EOSC ecosystem.

The baseline document was created during the European project INDIGO-DataCloud [R1] and used in the followup projects DEEP-Hybrid-DataCloud [R2] and eXtreme DataCloud [R3], where they have proved valuable for adopting the best practices for software development produced in the scientific European arena. In the EOSC-Synergy project, the document was further developed and improved regarding the criteria and its description.

Also in the framework of the EOSC-Synergy project, a second baseline document was created and evolved; "A set of Common Service Quality Assurance Baseline Criteria for Research Projects" [R6] (from here on referred to as SvcQA baseline), targeting the quality of services.

AI4EOSC will follow these recommendations, as a starting point, for the software development of project components and services.

It should be noted that the criteria specified in those documents are abstract and agnostic with respect to the technologies, tools and services that may be used to verify such criteria. Therefore, section 3 describes the tools and services that will be used to assess the validity of the Quality Criteria as well as the automatic FAIR assessment of digital objects used or produced by the project.

2.1. Minimal set of criteria to be assessed by SW and Services in AI4EOSC

Table 1 shows the minimal set of criteria that should be assessed by each software and service component developed or maintained by the project. One should note that other criteria may be assessed as well at the decision of the SW and service developers.

Codename	Description
QC.Acc01	Following the open-source model, the source code being produced MUST be open and publicly available to promote the adoption and augment the visibility of the software developments.
QC.Acc02	Source code MUST use a Version Control System (VCS).
QC.Acc03	Source code produced within the scope of a broader development project SHOULD reside in a common organization of a version control repository hosting service.

QC.Wor01	The main branch in the source code repository MUST maintain a working state version of the software component.
QC.Wor02	New changes in the source code MUST be placed in individual branches.
QC.Man01	An issue tracking system facilitates structured software development. Leveraging issues to track down both new enhancements and defects (bugs, documentation typos) is RECOMMENDED.
QC.Man02	A pull or merge request provides a place for review and discussion of the changes proposed to be part of an existing version of the code.
QC.Rev01	Code reviews MUST be done in the agreed peer review tool within the project
QC.Ver01	Semantic Versioning [R10] specification is RECOMMENDED for tagging the production releases.
QC.Lic01	As open-source software, source code MUST adhere to an open-source license to be freely used, modified and distributed by others. Non-licensed software is exclusive copyright by default. Licenses MUST be physically present (e.g. as a LICENSE file) in the root of all the source code repositories related to the software component.
QC.Lic02	License MUST be compliant with the Open Source Definition [R11].
QC.Doc01	Documentation MUST be treated as code.
QC.Doc02	Documentation MUST use plain text format using a markup language, such as Markdown or reStructuredText.
QC.Doc03	Documentation MUST be online and available in a documentation repository.
QC.Doc04	Documentation MUST be updated on new software versions involving any substantial or minimal change in the behaviour of the application.
QC.Doc05	Documentation MUST be updated whenever reported as inaccurate or unclear.
QC.Doc06	Documentation MUST be produced according to the target audience, varying according to the software component specification.
QC.Sty01	Each individual software product MUST comply with community-driven or de-facto code style standards for the programming languages being used.
QC.Sec01	Secure coding practices MUST be applied into all the stages of a software component development lifecycle. Compliance with Open Web Application Security Project (OWASP) secure coding guidelines [R12], is RECOMMENDED.
QC.Sec02	Source code MUST use automated linter tools to perform static application security testing (SAST) [R13], that flag common suspicious constructs that may cause a bug or lead to a security risk (e.g. inconsistent data structure sizes or unused resources).
QC.Sec04	World-writable files or directories MUST NOT be present in the product's configuration or logging locations.
QC.Del01	Production-ready code MUST be built as an artifact that can be efficiently executed on a system.
QC.Del02	The built artifact MUST be uploaded and registered into a public repository of such artifacts.
QC.Dep01	Production-ready code MUST be deployed as a workable system with the minimal user or system administrator interaction leveraging software configuration management (SCM) tools.
SvcQC.Api01	API testing MUST cover the validation of the features outlined in the specification.
SvcQC.Api02	API testing MUST include the assessment of the security-related criteria
SvcQC.Sec01	The Service public endpoints and APIs MUST be secured with data encryption.
SvcQC.Sec02	The Service SHOULD have an authentication mechanism.
SvcQC.Sec03	The Service SHOULD implement an authorization mechanism.
SvcQC.Sec04	The Service MUST validate the credentials and signatures.

Table 1: List of criteria that should be assessed by the project SW and services. Codenames starting with "QC" are from the SQA baseline [R5], while code names starting with "SvcQC" are from the SvcQA baseline [R6] for the SW that is applicable.

The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in the Table 1 are to be interpreted as described in RFC 2119 [R9].

2.2. FAIR DATA INDICATORS

One of the project's aims is fostering a FAIR and open platform targeted at users and developers working on AI/ML/DL applications. As such it is expected that the platform offers tools that allow to assess any corresponding subproducts for the FAIR principles, in particular: data, datasets, models, metadata and publications.

In addition, an automatic FAIR assessment tool is necessary to ensure compliance with the FAIR principles. That means that different digital objects produced along the workflow of the use cases will be properly identified to be findable and they will include proper metadata to be described including information about how they can be accessed and reused. Furthermore, the formats adopted will be open to enhance interoperability.

3. Services for QA verification and FAIR assessment

3.1. Services for QA

The implementation and assessment of the Quality Assurance of SW and Services of the project, will take advantage of already deployed services and tools from the EOSC-Synergy project [R4].

Figure 1 shows the several tools and services used for a typical QA for a given SW component that is a service, the initial list of services and corresponding endpoints will be kept updated in the Confluence wiki pages for WP7.

GitHub is used for:

- Version Control System (VCS) for the source code management and workflow.
- New features or bug fixes undergo through the process of a Pull Request (PR), where a CI/CD pipeline is triggered and executed.
- Source code review process by other collaborators or external parties.
- Issue tracker, where bugs are reported or new features are requested and tracked.
- Optionally for documentation of the SW components.
- Infrastructure as Code (IaC) such as Dockerfiles, docker-compose files and Ansible Roles or playbooks, or other Configuration Management System (CMS) files.

The SQA as a Service (SQAaaS) has three aims:

- CI/CD pipelines for assessment of a given SW component with the objective to get an award badge from EOSC-Synergy. For the badges, we rely on the OpenBadges specification and its implementation through the open-source Badgr platform and the GDPR-compliant instance provided by Concentric Sky.
- Custom composition of Quality Criteria with the objective to create a Jenkins pipeline. The Jenkins file and all necessary configurations are created so that they can be put in the Github source code repository.
- The criteria quality attributes to be verified are based on the SQA and SvcQA baselines.

Jenkins is the service used to execute the CI/CD pipelines created by SQAaaS and triggered by GitHub PRs.

In the case of SQA pipeline, in general the last step is the build of an artefact, for example a Docker image, that can be published in a Docker Hub or self hosted repository. While the case of SvcQA pipeline, starts by deploying and configuring the service. In the example shown in the figure, the service is deployed by the Infrastructure Manager (IM) [R17]. It pulls the image from Docker Hub, and executes playbooks based on Ansible roles hosted in Ansible Galaxy (an online platform which provides pre-packaged Ansible Roles to foster sharing and reuse). When the service is up and running, automated API and security checks can be performed by the Jenkins pipeline.

Most of these services (SQAaaS, IM, Jenkins and the services and platforms developed by the project), are deployed on top of the EGI Federated cloud [R15], that provides the Infrastructure as a Service (IaaS) computing, storage and network resources.

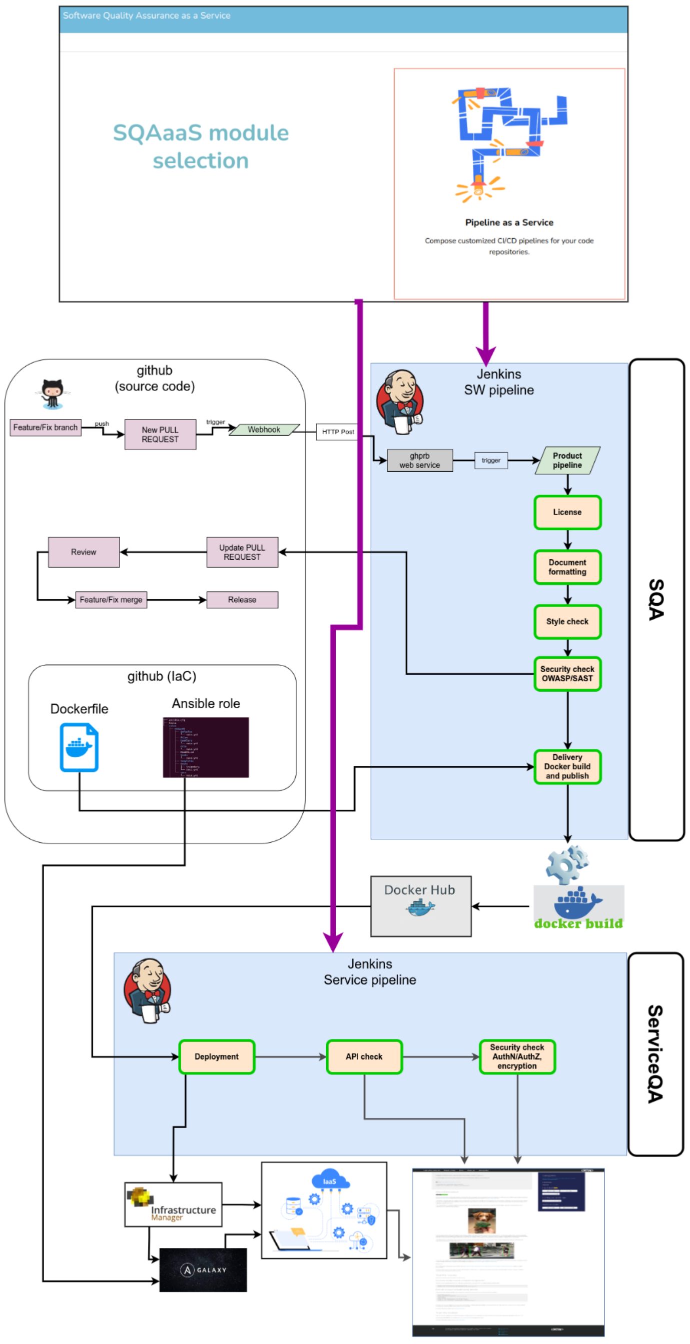


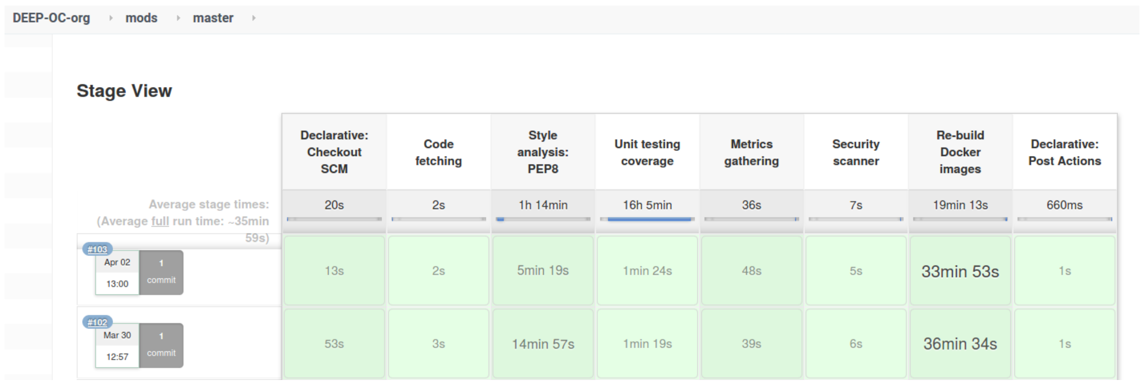
Figure 1: Diagram depicting a full SW and Service Quality Assurance workflow.

3.2. CI/CD PIPELINES FOR AI/ML/DL MODELS

The QA services described in the previous sub-section (3.1) can also be used to execute CI/CD pipelines for the AI/ML/DL models developed within the project, as well as for external use cases.

In collaboration with WP6, the AI/ML/DL applications are in general hosted in GitHub repositories and, therefore, some automated checks can be performed on such applications using specific Jenkins pipelines.

Previous work from the DEEP-Hybrid-DataCloud project [R2] is shown in Figure 2. The main nodes of the pipeline execute the following checks: "code style", "unit tests coverage", "static security (SAST)" and "Docker image build". Thus it is expected that the project's AI/ML/DL application developers implement the checks for a similar pipeline. It is also expected that the AI/ML/DL pipelines can be further improved according to the collected user requirements in the course of the project.


DEEP-OC-org › mods › master ›

Stage View

	Declarative: Checkout SCM	Code fetching	Style analysis: PEP8	Unit testing coverage	Metrics gathering	Security scanner	Re-build Docker images	Declarative: Post Actions
Average stage times: (Average full run time: ~35min 59s)	20s	2s	1h 14min	16h 5min	36s	7s	19min 13s	660ms
#103 Apr 02 13:00 1 commit	13s	2s	5min 19s	1min 24s	48s	5s	33min 53s	1s
#102 Mar 30 12:57 1 commit	53s	3s	14min 57s	1min 19s	39s	6s	36min 34s	1s

Figure 2: Jenkins pipeline execution example for DEEP repository https://github.com/deephdc/mods.

3.2. FAIR ASSESSMENT

The assessment of FAIRness of data, metadata and models produced by the AI/ML/DL applications, should be an automated process. Furthermore, the project will use a data repository service that enables this FAIR automated process.

In this regard, we will use the SQAaaS from EOSC-Synergy for data FAIR automated assessment since it already integrates the tool FAIR-EVA [R20]. This tool was developed by EOSC-Synergy to check the FAIRness level of digital objects from different repositories or data spaces. It requires the object identifier (preferably persistent and unique identifier) and the repository to be checked. It provides a generic and agnostic way to assess digital objects [R8].

FAIR EVA has been developed to assess RDA FAIR indicators under different scenarios.

4. Infrastructure resources for development

4.1. Computing resource providers

The computing resources needed for development, testing and integration are provided by the following IaaS cloud providers:

- The CSIC Scientific Cloud platform has more than 11000 vCPU cores, 1PB of CEPH based storage and more than 100 state of the art GPUs (NVIDIA V100, NVIDIA T4). The CSIC Scientific Cloud is ISO/IEC 9001:2005 certified.
- IISAS-FedCloud OpenStack site consists of 776 Intel CPU cores, 16 NVIDIA GPUs, 2.5 TB RAM, 144 TB of CEPH based storage, InfiniBand interconnection network, VM images with pre-installed NVIDIA drivers, CUDA and Docker provided for all users.
- INCD, the Portuguese National Distributed Computing Infrastructure. INCD is a digital research infrastructure that delivers computing and data services to the scientific community, supporting national and international projects. It consists of 2300 AMD cores, 7.7 TB of RAM and 320 TB of CEPH based storage.

The amount of resources to be allocated for the project from each provider is under discussion and follows as well from the use case requirements.

4.2. AAI integration

All cloud providers mentioned above are fully integrated with and working in production in EGI Federated Cloud [R15], it will ease as well the integration of other external cloud providers from this infrastructure. Authentication and authorization will be migrated to EGI Check-in (instead of DEEP IAM used in the DEEP-HybridDataCloud project).

A new VO ai4eosc.eu is being created in the EGI Checkin service for all members of the AI4EOSC project and for external users that want to test and integrate their applications in the AI4EOSC platform. Different roles will be created in the VO for different user groups:

- Platform operators: who can deploy and maintain the AI4EOSC platform on IaaS resource providers. They can manipulate infrastructure resources on IaaS providers such as creating VMs and deploying the platform software in the VMs.
- Application developers: who can submit jobs or applications to the platform. They cannot modify the platform or interact with IaaS computing resources, they are restricted to the execution of application containers on the platform.
- End users: who can log into the application services deployed in the platform.

4.3. Infrastructure tools and services

The following tools and services will be used for managing computing resources for the development and testing of the project’s software and services:

- FedCloud client [R16]: universal command-line client for EGI Federated Cloud. It can interact directly with services in EGI Federated Cloud such as the EGI Check-in, the service registry (GOCDB), the secret management; and execute OpenStack commands on sites in the cloud federation.
- Infrastructure Manager [R17]: A service that eases the access and the usability of cloud infrastructures by automating Virtual Machines Instances (VMI) selection, deployment, configuration, software installation, monitoring and update of Virtual Appliances.
- Dynamic DNS [R18]: Provides a dynamic Domain Name System (DNS) service for the EGI Cloud infrastructure. Users can register their own meaningful and memorable host names from a list of provided domains (e.g. fedcloud.eu) and assign to public IPs of their servers hosted in EGI Federated Cloud.

4.4. Infrastructure for software development and integration

The main software repositories for development will be hosted in the AI4EOSC organisation on GitHub [R14]. A read-only mirror of the repositories will be hosted in the GitLab server at IFCA [R19].

The software integration process will be automated through Jenkins CI/CD pipelines. Some of the existing pipelines, e.g. for software quality assurance, FAIRness assessment, are described in Section 3.

5. Software release process and procedures

5.1. Introduction

The AI4EOSC software components and services can be divided into three categories:

1. Software components developed within AI4EOSC, are the services that will compose the platform: Experiment dashboard, MLOps (requires as well external tools), the AI4EOSC Exchange (includes the Model provenance framework, model repository and catalogue) and the AI platform. Components that where initially developed and maintained by the INDIGO-DataCloud and later the DEEP-Hybrid-DataCloud projects such as: DEEPaaS and INDIGO PaaS Orchestractor, as well as some other components such as OSCAR (Open Source Serverless Computing for Data-Processing Applications) [R21].
2. The user AI/ML/DL applications from WP6, that correspond to the “Agrometeorological forecasts”, “Integrated plant protection scenario” and “Automated Thermography” use cases. Also other external users and use cases are planned.
3. Complementary external services and tools in order to have a coherent and complete AI/ML/DL platform where users applications can be executed, such as: The IaaS cloud resources from the EGI Fedcloud [R15], the IM [R17], the EGI AAI Checkin service [R22], data management services such as Nextcloud [R23]

and Hashicorp's NOMAD (application scheduler) + CONSUL (service discovery, health checking, etc.) [R24].

The AI4EOSC conceptual diagram is shown in Figure 3 for reference while the AI4EOSC platform architecture is currently under design in WP3. It will be reflected in the deployment and configuration on top of the IaaS resources.

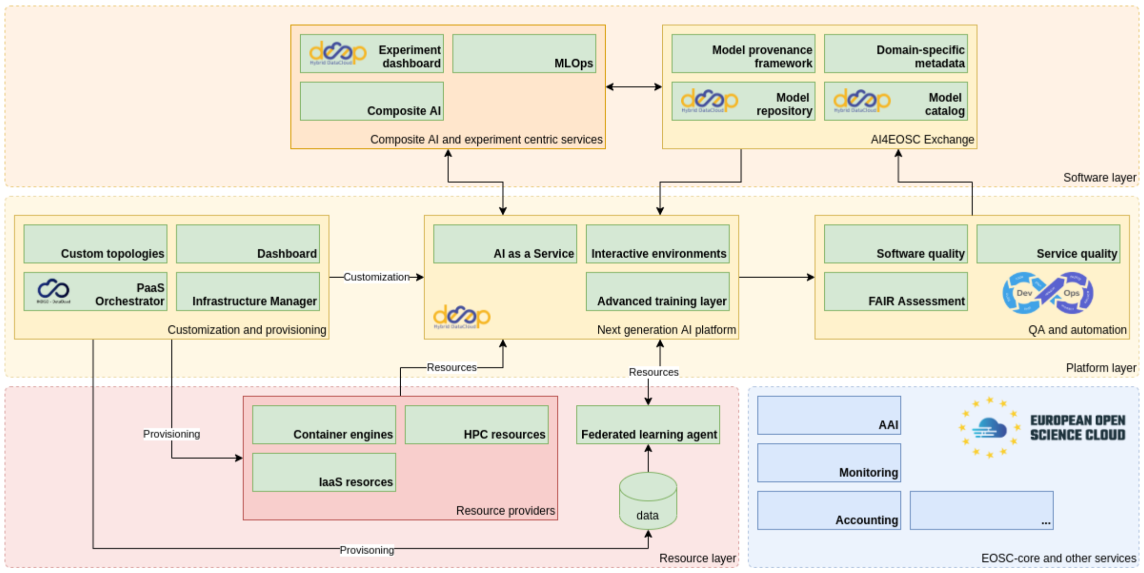


Figure 3: AI4EOSC conceptual diagram from the proposal.

5.2. SW RELEASE PROCESS

The SW release process is deemed to be a lightweight process. The software components are developed independent from each other, therefore it's up to the development teams to decide in coordination with WP7, the schedule of the release of any given software component.

The development and implementation of new features are planned with the WP7 team and in accordance with the use case requirements. Over the first project's months, several meetings have been held in WP6 to gather these requirements from each use case.

During development, IaaS resources and other necessary services (c.f. sub-section 5.3), are available to perform tests of any given software component. Release candidates should be also deployed in a preview testbed, so that software components can be integrated and tested with all other services of the AI4EOSC platform. At this stage, the users applications (and use cases), can preview and test the new versions of the components. The preview testbed should always be kept in a working state.

It is assumed that for a given software component to be deployed for testing, to have already passed the complete SQA and SvcQA pipelines described in the previous sections. The last stage of this pipeline is the build of the artifact, that in the case of most AI4EOSC components are in the form of Docker images that will be uploaded to DockerHub.

Regarding the software release procedure description, a dedicated wiki page will be created in the AI4EOSC Confluence space https://confluence.ifca.es/display/AI4.

This page will contain the full list of software components divided into the three above mentioned categories, alongside the responsible team or developers. It is also requested to the development teams to communicate and detail a roadmap for their software component.

The tracking of all software releases will be done with Jira tickets (https://jira.ifca.es/), that will allow WP7 to have a global view, at any given point in time, of the software status.

One should note that the Jira tickets are only used to track the software releases of components. Bugs and feature requests should be tracked in the corresponding source code GitHub repository. While problems with the services or platform, in general from users and operators should be reported in the EGI GGUS trouble ticketing system where a dedicated Support Unit, 3rd level - experts, with members of the AI4EOSC, will be created for this purpose.

There are two planned major releases of the AI4EOSC platform, the first major release is due on project month 18 (February 2024), while the second major release on project month 33 (May 2025). Given the nature of the release process being almost Agile [R26], and the release of any given component being quite flexible, the major release of the AI4EOSC platform is defined as the complete set of components with a given version at the time of the release.

The announcement of all AI4EOSC platform components and corresponding versions, will be done on the AI4EOSC website, including the release date of any given version. For a major release, this will include the information about all components and versions alongside the release date.

5.3. Testbeds for development and users preview

The infrastructure described in section 4.1 is very important for the deployment of the services and platform developed in the project, regarding testing and integration. As such, the testbed infrastructures will use resources operated by the project partners. The deployment and configuration of the services can be done through the IM using TOSCA templates and Ansible roles for those services.

Figure 3 shows the AI4EOSC conceptual diagram from the proposal., the yellow light boxes (top and middle of the diagram), are all services that compose the AI4EOSC platform, except the "QA and automation" (middle right), the lower left (light red) correspond to the cloud resources provided by the WP7 partners and the lower right (light blue box), correspond to the services needed for federation and integration with EGI (AAI, monitoring and accounting).

Given the Cloud flexibility and customizability nature of such infrastructure, there will be separate deployments of the project's developed services, targeted to different actors:

- For the software developers: The development testbed is an infrastructure composed of resources and services used for the development work in the project. It is where components under development are deployed and tested before they are deemed as candidates for a release or update and considered to be deployed in a preview testbed.

- For users preview and testing service integration: The preview testbed should be as stable as possible in order to allow a trusted environment for the user, but as well as for testing integration of new service versions.

Furthermore, two types of resource providers are identified: those who have GPUs for an "advanced scenario" - AI/ML/DL training, and those which provide only CPUs that can be used for inference scenarios.

5.4. Services for the SW release process

The list of services to support the software release process is shown in Table 2 together with the URL endpoint and a short description.

Service	Endpoint	Description
Confluence	https://confluence.ifca.es/display/AI4	Internal wiki project documentation and organization
Jira	https://jira.ifca.es/	Task tracker for the software releases
Mailing list	https://listas.csic.es/wws/subscribe/ai4eosc-wp7	WP7 mailing list
Website	https://ai4eosc.eu/	

Table 2: List of services and corresponding endpoints for the software release process

6. Final Remarks

This document describes the initial plan for the software and services QA, how it will be implemented and assessed. Also how data, metadata and models will be assessed for FAIRness and finally the Cloud infrastructure providers to support all activity in WP7.

In the next steps, after the AI4EOSC platform architecture is defined, it will be implemented and deployed in the preview testbed. The SQA and SvcQA criteria are already known to the developers and in many cases already implemented in their development practices.

References

[R1] INDIGO-DataCloud project: https://www.indigo-datacloud.eu/.

[R2] DEEP-Hybrid-DataCloud project: https://deep-hybrid-datacloud.eu/.

[R3] eXtreme DataCloud project: https://www.extreme-datacloud.eu/.

[R4] EOSC-Synergy project: https://www.eosc-synergy.eu/.

[R5] Pablo Orviz, Alvaro Lopez, Doina Cristina Duma, Mario David, Jorge Gomes, Giacinto Donvito, “A set of Common Software Quality Assurance Baseline Criteria for Research Projects”, 2017, https://digital.csic.es/handle/10261/160086 (Github repository: https://github.com/indigo-dc/sqa-baseline).

[R6] Orviz Fernández, Pablo ; Mario David; Jorge Gomes; Joao Pina; Samuel Bernardo; Campos Plasencia, Isabel ; Germán Moltó; Miguel Caballer, “A set of Common Service Quality Assurance Baseline Criteria for Research Projects”, June 2020: DOI http://dx.doi.org/10.20350/digitalCSIC/12533 (Github repository: https://github.com/EOSC-synergy/service-qa-baseline).

[R7] FAIR Data Maturity Model Working Group. (2020). FAIR Data Maturity Model. Specification and Guidelines (1.0): https://doi.org/10.15497/rda00050.

[R8] Fernando Aguilar, Jorge Gomes, Isabel Bernal, Wilko Steinhoff, Vyacheslav Tykhonov, “EOSC-SYNERGY EU DELIVERABLE D3.5: Final report on technical framework for EOSC FAIR data principles implementation”, Oct. 2022: http://hdl.handle.net/10261/281891.

[R9] Key words for use in RFCs to Indicate Requirement Levels, Scott O. Bradner, Internet Engineering Task Force (1997-03): https://datatracker.ietf.org/doc/rfc2119/.

[R10] Semantic Versioning: https://semver.org.

[R11] Open Source Definition: https://opensource.org/osd.

[R12] Open Web Application Security Project (OWASP) secure coding guidelines: https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/migrated_content.

[R13] Static Application Security Testing (SAST): https://owasp.org/www-community/Source_Code_Analysis_Tools.

[R14] AI4EOSC Software repository: https://github.com/AI4EOSC.

[R15] EGI Federated cloud: https://www.egi.eu/service/cloud-compute/.

[R16] FedCloud client: https://fedcloudclient.fedcloud.eu/.

[R17] Infrastructure Manager:https://www.grycap.upv.es/im.

[R18] Dynamic DNS: https://nsupdate.fedcloud.eu/.

[R19] IFCA Advanced Computing GitLab server: https://gitlab.ifca.es/.

[R20] FAIR EVA - Evaluator, Validator and Advisor. DOI: 10.20350/digitalCSIC/14559

[R21] Open Source Serverless Computing for Data-Processing Applications (OSCAR): https://oscar.grycap.net/.

[R22] AAI EGI Checkin service: https://aai.egi.eu/registry/.

[R23] Nextcloud: https://nextcloud.com/.

[R24] NOMAD and CONSUL: https://developer.hashicorp.com/nomad.

[R25] EGI GGUS trouble ticketing system: https://ggus.eu/.

[R26] The 12 Principles behind the Agile Manifesto: https://www.agilealliance.org/agile101/12-principles-behind-the-agile-manifesto/.

GLOSSARY

Acronym	Description
API	Application Programming Interface
CI/CD	Continuous Integration / Continuous Delivery
CLI	Command Line Interface
DO	Digital Object
DOI	Digital Object Identifier
EOSC	European Open Science Cloud
FAIR	Findable Accessible Interoperable Reusable
IaC	Infrastructure as Code
IM	Infrastructure Manager
OWASP	Open Web Application Security Project
OSS	Open Source Software
PR	Pull Request
RDA	Research Data Alliance
SAST	Static Application Security Testing
SCM	Software Configuration Management
SQA	Software Quality Assurance
SvcQA	Service Quality Assurance
SQAaaS	Software Quality Assurance as a Service
VCS	Version Control System