



interTwin

D6.2 First release of the DTE core modules

Status: Under EC review
Dissemination Level: public



Funded by the
European Union


Disclaimer: Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. Neither the European Union nor the granting authority can be held responsible for them

Abstract

Key Words | DTE, Core, development, integration

The deliverable includes the status of the development and integration of all WP6 software products available with the first release.



Document Description			
D6.2 First release of the DTE core modules			
Work Package number 6			
Document type	Deliverable		
Document status	Under EC review	Version	1
Dissemination Level	Public		
Copyright Status	 <p>This material by Parties of the interTwin Consortium is licensed under a Creative Commons Attribution 4.0 International License.</p>		
Lead Partner	CSIC		
Document link	https://documents.egi.eu/document/3951		
DOI	https://doi.org/10.5281/zenodo.10224213		
Author(s)	<ul style="list-style-type: none"> • Isabel Campos (CSIC) • Germán Moltó (UPV) • Alexander Jacob (EURAC) • Pablo Orviz (CSIC) • Miguel Caballer (UPV) • Matteo Bunino (CERN) • Alexander Zochbauer (CERN) • Sandro Fiore (UNITN) 		
Reviewers	<ul style="list-style-type: none"> • Renato Santana (EGI) • Matthias Schramm (TU Wien) 		
Moderated by:	<ul style="list-style-type: none"> • Sjomara Specht (EGI) 		
Approved by	AMB		

Revision History			
Version	Date	Description	Contributors
V0.1	12/09/2023	ToC	Isabel Campos (CSIC)
V0.2	24/10/2023	Input from all the tasks	Germán Moltó (UPV) Alexander Jacob (EURAC) Pablo Orviz (CSIC) Miguel Caballer (UPV) Matteo Bunino (CERN) Alexander Zoechbauer (CERN) Sandro Fiore (UNITN)
V0.3	26/10/2023	Introduction and summaries included	Isabel Campos (CSIC)
V0.4	06/11/2023	Sent to internal reviewers	Isabel Campos (CSIC)
V0.5	11/11/2023	Internal review	Matthias Schramm (TU Wien), Renato Santana (EGI)
V0.6	23/11/2023	Version Ready for QA	Isabel Campos (CSIC)
V1.0	24/11/2023	Final	

Terminology / Acronyms	
Term/Acronym	Definition
DT	Digital Twin, a digital representation of an actual physical product, system, or process that serves as the effectively indistinguishable digital counterpart of it for practical purposes, such as simulation, integration, testing, monitoring, and maintenance
DTE	Digital Twin Engine, a platform to build DTs
CLI	Command line interface
GUI	Graphical user interface
API	Application Programming Interface, aka programmatic interface of a computer system through which other computer systems can interact with it
REST API	API that conforms to the design principles of REST, or representational state transfer architectural style
CI/CD	In software engineering, CI/CD is the combined practices of continuous integration and continuous delivery
AI	Artificial Intelligence
ML	Machine Learning is a branch of AI and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy

Terminology / Acronyms: <https://confluence.egi.eu/display/EGIG>



Table of Contents

1	Introduction	7
1.1	Scope	7
1.2	Document Structure.....	7
2	Core Components	8
2.1	Components for Task 6.1.....	8
2.1.1	OSCAR.....	8
2.1.2	dCNiOS.....	10
2.1.3	openEO.....	11
2.1.4	Ophidia.....	12
2.1.5	yProv.....	13
2.2	COMPONENTS FOR TASK 6.2	14
2.2.1	SQAaaS platform	14
2.2.2	SQAaaS GitHub Actions	15
2.2.3	SQAaaS CLI.....	16
2.3	COMPONENTS FOR TASK 6.3	17
2.3.1	openEO.....	18
2.4	COMPONENTS FOR TASK 6.4	19
2.4.1	Infrastructure Manager.....	19
2.4.2	Big Data Analytics TOSCA templates.....	19
2.4.3	Configuration artefacts	21
2.5	COMPONENTS FOR TASK 6.5	21
2.5.1	itwinai.....	21
3	Summary of integration status	23

Table of Tables

<i>Table 1 - List of core components under release by task.</i>	8
--	----------



Executive summary

This report describes the status of development of the core components that have been identified at this stage of the project to support transversal functionalities when deploying Digital Twins. The design considerations have been addressed in a previous deliverable (D6.1)¹.

Real time data ingestion and analysis will be supported by OSCAR and dCNiOS. Workflow management is being developed in the frameworks of openEO API and of the Ophidia tools. The yProv tool will enhance workflows with data provenance information. Model quality and validation will be supported by the evolution of the SQAaaS service towards automated validation.

The interplay between the infrastructure and the required big data analytics core services is dealt with by the Infrastructure Manager (IM), complemented by a set of TOSCA templates and configuration artefacts repositories. Several already existing tools and APIs are integrated here as a core component. Additionally, the AI workflow itwinai is being developed here from scratch (see task 6.5 in table I), a python-based library that streamlines AI workflows, while integrating with HPC and cloud resources.

The first testing strategies are also briefly laid out regarding integration at the infrastructure level.

¹ <https://zenodo.org/record/8036987>



1 Introduction

1.1 Scope

This deliverable summarises the status of development of core components relevant to implement the architecture of a Digital Twin Engine in the framework of interTwin.

1.2 Document Structure

In [Section 2](#) we introduce the list of core services under development. We provide details on their functionalities, release notes and future plans. In [Section 3](#) we provide a summary on the status of integration with the rest of the project Work Packages.



2 Core Components

The list of core components currently under development is shown in **Table 1**. The design considerations are available in deliverable D6.1.

Table 1 - List of core components under release by task.

Task 6.1	Task 6.2	Task 6.3	Task 6.4	Task 6.5
<ul style="list-style-type: none"> - OSCAR - dCNiOS - openEO - Ophidia - yProv 	<ul style="list-style-type: none"> - SQAaaS - SQAaaS - Github action - SQAaaS CLI 	<ul style="list-style-type: none"> - openEO 	<ul style="list-style-type: none"> - Infrastructure Manager (IM) - TOSCA templates - Configuration artefacts 	<ul style="list-style-type: none"> - itwinai

2.1 Components for Task 6.1

2.1.1 OSCAR

Component name	OSCAR https://oscar.grycap.net
Description	OSCAR is an open-source platform to support the event-driven serverless computing model for data-processing applications. It can be automatically deployed on multi-clouds, and even on low-powered devices, to create highly parallel event-driven data-processing serverless applications along the computing continuum. These applications execute on customised runtime environments, provided by Docker containers which run on elastic Kubernetes clusters.
Value proposition	Users can set up an OSCAR cluster themselves on any available cloud infrastructure. The automatically scalable cluster can be used to scale file-based on-demand processing (e.g. automatically when a file is uploaded to an object store such as MinIO). Furthermore, it allows communication via HTTP-based calls for programmatic interaction with auto-scaled, stateless, user-defined services.

D6.2 First release of the DTE core modules

Users of the Component	Scientific users require data-driven processing on multiple cloud back-ends. Non-expert users can interact via high-level web-based GUIs, while advanced users can use a command-line interface (CLI).
User Documentation	https://oscar.grycap.net/blog/
Technical Documentation	https://docs.oscar.grycap.net/
Responsible	GRyCAP-I3M-UPV - products@grycap.upv.es Contact point: Germán Moltó - gmolto@dsic.upv.es
License	Apache 2.0
Source code	https://github.com/grycap/oscar

2.1.1.1 Release notes

OSCAR has been adopted by interTwin project in order to implement a generic framework for real-time data acquisition and processing that builds on event-triggered execution of workflows, to i) improve response times and minimise data transfers, and to ii) support new event sources.

For the first release, OSCAR has been extended to provide support to Apache Nifi, an open-source system that supports powerful and scalable directed graphs of data routing, transformation, and system mediation logic. The goal is to use Apache Nifi as an intermediate event-ingestion platform (buffering) to decouple the rate at which files are uploaded to an object/file, from the rate at which these data files are processed. In order to facilitate this integration, the component dCNiOS was created (see next subsection). This required i) an adaptation to the internal event structure that is used by Nifi and ii) an adaptation to perform dynamic data fetching from the FaaS supervisor component in OSCAR, to prevent data movements from the data storage system into Nifi (and then into OSCAR).

2.1.1.2 Future plans

OSCAR will be evolved, during the interTwin project, to accommodate new sources of events. It has been explored the usage of TriggerMesh to support incoming events from Amazon S3, a widely used data lake and object storage system from Amazon Web Services (AWS), the leading public Cloud provider. TriggerMesh will allow us to create a gateway between public data lakes and thus to support long-term data persistence. It also enables the dynamic deployment of OSCAR clusters on any cloud infrastructure - such as the EGI's federated cloud or as any public cloud provider - on which the project-defined workflows can be triggered. Also, an integration between OSCAR and the interLink component will be performed in order to achieve event-driven support for data processing delegation into HPC supercomputers.



2.1.2 dCNiOS

Component name	dCNiOS
Description	dCNiOS (dCache + NiFi + OSCAR) is a new open-source command-line tool to easily manage the creation of event-driven data processing flows.
Value proposition	When files are uploaded to dCache, events are ingested by Apache NiFi, which can queue them up depending on the (modifiable at runtime) ingestion rate. Then, they are delegated for processing into a scalable OSCAR cluster, where a user-defined application, based on a Docker image, can process the data file.
Users of the Component	Technology Integrators
User Documentation	http://github.com/interTwin-eu/dcnios
Technical Documentation	http://github.com/interTwin-eu/dcnios
Responsible	GRyCAP-I3M-UPV - products@grycap.upv.es Contact point: Germán Moltó - gmolto@dsic.upv.es
License	Apache 2.0 License
Source code	https://github.com/grycap/dcnios

2.1.2.1 Release notes

This is a new development performed in interTwin, facilitating the integration of dCache within OSCAR. Apart from the command-line application available in the GitHub repository above, this development has also been supplemented i) by the corresponding TOSCA templates and ii) by ansible roles that are required to deploy an Apache NiFi cluster via the Infrastructure Manager (IM) described in [section 2.4.1](#) and [section 2.4.2](#). Any user can self-deploy such a cluster via the IM Dashboard - <https://im.egi.eu>.

2.1.2.2 Future plans

dCNiOS will be further extended according to the use case requirements. It currently supports user / password authentication to connect to the components. OIDC support may be integrated if it is deemed necessary.



2.1.3 openEO

Component name	openEO
Description	openEO is an application programming interface (API) that supports i) the management of workflows, ii) job handling, and iii) linking to data sources and processing capabilities on compatible cloud platform providers in a standardised way.
Value proposition	openEO can be extended to support execution of containerized software packages as execution of specific processes following the OGC API processes approach. openEO has many implementations, but a specific set of them can be selected for handling workflows in interTwin.
Users of the Component	Digital Twin Developers
User Documentation	https://openeo.org/documentation/1.0/
Technical Documentation	https://openeo.org/documentation/1.0/developers/
Responsible	Eurac, EODC, Münster University, Alexander Jacob, Christoph Reimer, Brian Pondi alexander.jacob@eurac.edu

2.1.3.1 Release notes

For the interTwin project a specific deployment has been composed that i) bases on the available open-source components under the openEO's GitHub repository and that ii) allows the further integration of new component types into the workflows, based on the openEO process graphs. Those new types to be integrated are set up as the so-called application packages that are defined by the EOEPKA architecture² and that are used in the OGC standard³ (e.g. as the OGC API processes). A first prototype has been developed integrating one specific processor in this way, based on the wflow hydrological model⁴. Furthermore, parts of the use case on flood mapping based on the Global Flood Mapping algorithm, have also been ported to openEO (see Task 7.5).

² <https://eoepka.org/>

³ <https://www.ogc.org/standards>

⁴ <https://deltares.github.io/Wflow.jl/stable/>



2.1.3.2 Future plans

This first process can act as a template for other thematic modules, to be defined and set up the same way and then to be executed. Ideally, new components should also be developed to interact with the other core components (e.g. with the iTwinAI package; see Task 6.5) That could probably also fully be described in this way. Also, a connection to the OSCAR and dCNiOS event-driven triggering of workflows is foreseen, so that event can trigger the execution of openEO process graphs. As a next step, a TOSCA template for this environment should be developed together with Task 6.4.

2.1.4 Ophidia

Component name	Ophidia/PyOphidia
Description	PyOphidia provides the Python bindings for Ophidia, a High-Performance Data Analytics framework
Value proposition	Ophidia framework is an open-source solution for the analysis of scientific multi-dimensional data, joining HPC paradigms and Big Data approaches. It provides an environment targeting High Performance Data Analytics through parallel and in-memory data processing, data-driven task scheduling and server-side analysis. The framework supports the execution of complex analytics workflows in the form of DAGs of Ophidia operators.
Users of the Component	Data scientists, developers
User Documentation	https://pyophidia.readthedocs.io/en/latest/
Technical Documentation	https://pyophidia.readthedocs.io/en/latest/installation.html
Responsible	CMCC, Ophidia dev team: ophidia-info@cmcc.it
License	GPLv3
Source code	https://github.com/OphidiaBigData/PyOphidia

2.1.4.1 Release notes

The Ophidia framework, and in particular its Python bindings, PyOphidia, have been extended in order to provide a preliminary support for Ophidia workflows coded in Common Workflow Language (CWL): <https://github.com/OphidiaBigData/PyOphidia/tree/feature-cwl/PyOphidia/utils>. In this pre-release new Python modules are provided for translating workflows from CWL format to the native Ophidia one. The new capability is based on the CWLtool command



D6.2 First release of the DTE core modules

line interface, which performs this translation. The new capabilities also integrate direct submission of the translated workflow to the Ophidia server.

2.1.4.2 Future plans

A stronger integration of the capabilities for supporting CWL format is envisioned in future releases. Moreover, support for workflows including non-Ophidia tasks will be explored. Besides, full documentation of the new capabilities, including examples of usage will be added.

2.1.5 yProv

Component name	yProv
Description	yProv is an open-source service to support provenance management within scientific workflows. It relies on the W3C PROV family of standards, a RESTful interface and a graph database back-end based on Neo4J. The yProv web service is implemented in Python by using the Flask micro-framework which is based on the Jinja2 Template Engine and Werkzeug WSGI Toolkit. The service is domain-agnostic, though its primary case studies in the project come from the climate change domain (i.e. climate analytics workflows). The service aims at implementing the micro-provenance concept, to navigate within the provenance space across different dimensions (e.g., horizontal & vertical).
Value proposition	Users can exploit the yProv service to manage (i.e. store, retrieve, explore, visualise) the provenance information associated with scientific datasets, thus getting a better understanding about specific datasets. The value proposition is about (i) stronger traceability, transparency, and trust (through a richer set of metadata) and (ii) multidimensional exploration/navigation of provenance metadata information (i.e., multi-level).
Users of the Component	Scientific users, both producers and consumers of datasets. End users can interact via the yProv RESTful API to manage (i.e., CRUD operations) the provenance information.
User Documentation	https://github.com/HPCI-Lab/yProv/blob/main/README.md



D6.2 First release of the DTE core modules

Technical Documentation	https://github.com/HPCI-Lab/yProv/blob/main/tech-doc.MD
Responsible	University of Trento Contact point: Sandro Fiore (sandro.fiore@unitn.it)
License	GPLv3
Source code	https://github.com/HPCI-Lab/yProv

2.1.5.1 Release notes

yProv has been adopted in InterTwin to implement provenance support within scientific workflows, starting from some case studies identified in the environmental domains (i.e. climate data analytics workflows). Being a new effort, the first release is still pre-operational, though it already provides a preliminary set of core functionalities.

2.1.5.2 Future plans

yProv will be evolved during InterTwin in order to accommodate additional requirements. A cloud-enabled version of the service based on containers will be implemented; an automated service deployment over Kubernetes clusters, will be performed in the project by leveraging cloud components, like IM. A more extensive and user-friendly CLI will be developed to better address usability. The multi-dimensional support will also be integrated to address the vertical exploration of provenance within complex scientific workflows, thus fully implementing the micro-provenance concept. Finally, a long-running service instance will be established within EOSC as a reference service for the community.

2.2 COMPONENTS FOR TASK 6.2

2.2.1 SQAaaS platform

Component name	SQAaaS https://sqaas.eosc-synergy.eu
Description	Platform for quality assessment and awarding of multiple digital objects (source code, services, data)
Value proposition	Provide a module for quality validation within the interTwin's DTE
Users of the Component	DTE developers & users
User Documentation	https://docs.sqaas.eosc-synergy.eu https://indigo-dc.github.io/jenkins-pipeline-library



D6.2 First release of the DTE core modules

Technical Documentation	https://github.com/eosc-synergy/sqaaas-api-spec
Responsible	Pablo Orviz < orviz@ifca.unican.es > Samuel Bernardo < samuel@lip.pt > David Arce < darce@i3m.upv.es >
License	GPL-3.0-only
Source code	https://github.com/eosc-synergy/sqaaas-api-server https://github.com/eosc-synergy/sqaaas-web https://github.com/indigo-dc/jenkins-pipeline-library

2.2.1.1 Release notes

Several minor releases have been delivered for each SQAaaS component during the InterTwin first 12 months (API server: 2.11.x through 2.17.2, Web: 2.0.0 through 2.9.0). Those related to InterTwin project requirements include:

- Support for assessing private repositories through GitHub personal tokens,
- Enhancement of data FAIR analysis by offering hints to improve not fulfilled criteria,
- Improve metadata of digital badges.

2.2.1.2 Future plans

The next steps for T6.2 are:

- Extend the current type of assessments (custom assessments) to deal with specific requirements on model validation and data quality of the existing use cases, in particular those participating in T6.2 and T6.5.
- Integrate the SQAaaS platform with the WfMS technologies being supported by T6.1, such as CWL.

2.2.2 SQAaaS GitHub Actions

Component name	SQAaaS assessment GitHub Action
Description	Trigger SQAaaS quality assessment service from GitHub actions
Value proposition	Integrate interTwin's GitHub organisation with the SQAaaS platform for software quality (incl. workflow and model code)
Users of the Component	DTE developers & users



D6.2 First release of the DTE core modules

User Documentation	https://github.com/EOSC-synergy/sqaaas-assessment-action https://github.com/EOSC-synergy/sqaaas-step-action
Technical Documentation	https://github.com/eosc-synergy/sqaaas-gh-action https://github.com/EOSC-synergy/sqaaas-step-action
Responsible	Pablo Orviz < orviz@ifca.unican.es >
License	GPL-3.0-only
Source code	https://github.com/EOSC-synergy/sqaaas-assessment-action https://github.com/EOSC-synergy/sqaaas-step-action

2.2.2.1 Release notes

The current release includes two GitHub actions (`sqaaas-assessment-action` and `sqaaas-step-action`) that enable the automated assessment of source code, including workflow and model code, by triggering the SQAaaS platform. More precisely, the main action (`sqaaas-assessment-action`) is in charge of interacting with the SQAaaS API, running the appropriate HTTP requests to conduct the source code assessment. As an output of this action, a summary containing the quality criteria being analysed is provided, and, in the event that a certain level of these criteria has been fulfilled, the corresponding digital badge that recognizes those achievements.

As a complement, the step action (`sqaaas-step-action`) adds the capability to define customised steps as part of the evaluation of a quality criterion within the SQAaaS source code assessment. This is required, for instance, for the testing criteria, where diverse testing frameworks might be used (e.g. Python's `pytest`). Additionally, this action serves the purpose of covering pre/post requirements that might be needed as part of the quality criteria validation. An example could be setting up the environment as a 'pre' condition before proceeding with the actual testing process (e.g. following the Python example: `conda`, `virtualenv`, etc.).

2.2.2.2 Future plans

Extend the main GitHub action ([sqaaas-assessment-action](#)) to cope with other types of quality assessments currently provided by the SQAaaS platform, in particular the validation of FAIR principles for data.

2.2.3 SQAaaS CLI

Component name	SQAaaS CLI
----------------	------------



D6.2 First release of the DTE core modules

Description	Command line interface to interact with SQAaaS API from scripts
Value proposition	Integrate interTwin's workflows within pipeline steps, creating assessments, getting outputs, checking pipeline status
Users of the Component	DTE developers & users
User Documentation	https://gitlab.a.incd.pt/eosc/eosc-synergy/sqaas-cli/-/tree/cli-2.6.0
Technical Documentation	https://gitlab.a.incd.pt/eosc/eosc-synergy/sqaas-cli/-/tree/cli-2.6.0
Responsible	Samuel Bernardo < samuel@lip.pt > Pablo Orviz < orviz@ifca.unican.es >
License	GPL-3.0-only
Source code	https://gitlab.a.incd.pt/eosc/eosc-synergy/sqaas-cli/-/tree/cli-2.6.0

2.2.3.1 Release notes

The SQAaaS CLI is a work in progress, to answer the custom assessment required for DTE workflow integration into the SQAaaS platform. The CLI approach provides a way to launch commands from the workflows and run the required steps.

Release 2.6.0 reflects the SQAaaS production schema is not supporting the custom assessment yet. But it answers the quality checking task, providing the commands to create an assessment, get assessment outputs, run pipelines, and get pipeline status.

2.2.3.2 Future plans

Further developments towards the custom assessment (CA), that will answer the DTE requirements for the model validation. We foresee two possible use cases: trigger assessment from GitHub or from a workflow step. The new features will comprehend the configuration (from a file, template, or web interface), register, and uniquely identify the CA from a URI representation.

2.3 COMPONENTS FOR TASK 6.3

The Core components for data fusion are partially aligned with the core components for the workflow management tools like openEO.



2.3.1 openEO

Component name	openEO
Description	openEO is an application programming interface (API), supporting management of workflows and handling of jobs, as well as linking to available data sources and processing capabilities in a harmonised way - front-end users don't need knowledge on the data structure.
Value proposition	The openEO syntax is already used for a number of specific predefined processes, dealing with fusion of raster data. Those use cases can be further extended for integrating vector data and possibly data coming as output from different models from both the physical and data driven domain as well as preparing data in a harmonised way for ingestion into such models.
Users of the Component	Digital Twin Developers
User Documentation	https://openeo.org/documentation/1.0/
Technical Documentation	https://openeo.org/documentation/1.0/developers/ , https://processes.openeo.org/
Responsible	Eurac, EODC, Münster University, Alexander Jacob, Christoph Reimer, Brian Pondi alexander.jacob@eurac.edu

2.3.1.1 Release notes

openEO-based processes for fusion of raster data are already used in interTwin for the preparation of various sources of raster data in the environmental use cases, specifically dealing with early warning of extreme flood and drought events. TU Wien and Eurac are using those process libraries for monitoring flood and drought events; Deltares is currently evaluating their suitability for its own processes.

2.3.1.2 Future plans

The firstly deployed openEO-based processes shall be used as templates for identifying the potential of other thematic modules, whether they can be defined and set up the same way and then be executed. Additionally, it needs to be seen if other common



D6.2 First release of the DTE core modules

functionalities from data fusion can be extracted from the thematic modules and be promoted to the level of the generic core engine.

2.4 COMPONENTS FOR TASK 6.4

2.4.1 Infrastructure Manager

Component name	Infrastructure Manager (IM)
Description	The Infrastructure Manager (IM) is an open-source Infrastructure as Code (IaC) tool that provides both an XML-RPC and REST API to receive virtual infrastructure provision requests for their deployment on IaaS cloud back-ends. These requests may come from a web-based graphical user interface such as the IM-Dashboard, through the IM command-line interface (CLI) or via an HTTP-based client.
Value proposition	Users can self-provision any kind of complex virtual infrastructure on whichever cloud infrastructure they can access.
Users of the Component	Scientific users, requiring Big Data Analytics tools to be deployed on different Cloud back-ends.
User Documentation	https://imdocs.readthedocs.io/en/latest/
Technical Documentation	https://imdocs.readthedocs.io/en/latest/
Responsible	GRyCAP-I3M-UPV - products@grycap.upv.es Contact point: Miguel Caballer - micafer1@upv.es
License	GPL 3.0
Source code	https://github.com/grycap/im

2.4.1.1 Release notes

IM has been adopted in interTwin for the deployment of the Big Data Analytics tools.

2.4.1.2 Future plans

IM will be evolved in interTwin, steered by upcoming user requirements to the deployments of the Big Data Analytics tools.

2.4.2 Big Data Analytics TOSCA templates

Component name	Big Data Analytics TOSCA templates
----------------	------------------------------------



D6.2 First release of the DTE core modules

Description	As a set of TOSCA templates to deploy Big Data Analytics tools
Value proposition	TOSCA templates enable the description, in a cloud-agnostic way, of the virtual infrastructures needed in the available Big Data Analytics tools.
Users of the Component	TOSCA template developers.
User Documentation	https://confluence.egi.eu/display/interTwin/TOSCA+Templates
Technical Documentation	https://confluence.egi.eu/display/interTwin/TOSCA+Templates
Responsible	UPV
License	Apache 2.0
Source code	https://github.com/grycap/tosca/tree/main/templates

2.4.2.1 Release notes

In this release, the following templates have been created:

- **KubeFlow**: Template to deploy the Kubeflow machine learning (ML) workflows platform on top of Kubernetes.
- **Airflow**: Template to deploy the Apache workflows system on top of Kubernetes.
- **CernVMFS**: Install CernVMFS on a VM and mount a list of CernVM-FS repositories specified by the user.
- **Kafka**: Deploy Kafka distributed event streaming platform on top of a Kubernetes cluster.
- **MLFlow**: Deploy the MLFlow platform to manage the ML lifecycle in a single VM, with possibility to store the artefacts in an external S3 (or MinIO) storage system.

WP5 and WP6 members have tested the templates before the release and plans for the testing with DT use cases have been done

2.4.2.2 Future plans

Some of the templates are in an early stage (Kafka and MLFlow) and need to be correctly tested by users with experience using these tools to validate the functionality of the deployed infrastructure. Other templates are more mature but may need some additions to improve them. Finally, further templates need to be created (e.g. for the openEO compatible back-ends).



2.4.3 Configuration artefacts

Component name	Configuration artefacts
Description	Ansible playbooks and roles and any other artefact needed in the deployment of the selected Big Data Analytics tools. Those ansible playbooks are used by the TOSCA templates defined in section 2.4.2 in order to perform the actual deployments on cloud nodes.
Value proposition	TOSCA templates refer to these artefacts to enable the configuration of the Big Data Analytics tools using the Ansible tool.
Users of the Component	TOSCA template/Ansible playbooks developers.
User Documentation	-
Technical Documentation	-
Responsible	UPV
License	Apache 2.0
Source code	https://github.com/grycap/tosca/tree/main/artifacts

2.4.3.1 Release notes

Artefacts created:

- [KubeFlow](#), [Kafka on docker compose](#), [Kafka on Kubernetes](#), [CernVMFS](#) and [MLFlow](#), see [section 2.4.2.1](#) for the description of the components.

2.4.3.2 Future plans

Some of the artefacts are in an early stage (Kafka and MLFlow) and need to be validated by users with experience, using these tools to also validate the functionality of the deployed infrastructure. Other artefacts are more mature but may need some additions to improve them. Finally, further templates need to be created (e.g. for the openEO compatible back-ends).

2.5 COMPONENTS FOR TASK 6.5

2.5.1 itwinai

Component name	itwinai
Description	itwinai is a Python library that streamlines AI workflows,



D6.2 First release of the DTE core modules

	while reducing coding complexity. It seamlessly integrates with HPC resources, making workflows highly scalable and promoting code reuse. With built-in tools for hyperparameter optimization, distributed machine learning, and pre-trained ML models. itwinai empowers AI researchers. It also integrates smoothly with Jupyter-like GUIs, enhancing accessibility and usability.
Value proposition	Different interfaces, to lower the entry barrier for users coming from different fields of expertise: from lower-level python programming to high-level GUI workflow representation. itwinai provides out-of-the-box SOTA AI tools and encourages code reuse, to further simplify and streamline the development of ML workflows, on top of seamless integration with HPC resources.
Users of the Component	DT developers, ML engineers and researchers.
User Documentation	https://intertwin-eu.github.io/itwinai/
Technical Documentation	https://intertwin-eu.github.io/itwinai/
Responsible	CERN, FZ Juelich. Contact points: Matteo Bunino - matteo.bunino@cern.ch , Alexander Zoechbauer - alexander.zoechbauer@cern.ch , Rakesh Sarma - r.sarma@fz-juelich.de
License	MIT License
Source code	https://github.com/interTwin-eu/itwinai

2.5.1.1 Release notes

The itwinai library has been adopted in interTwin for the definition of Advanced AI workflows. Up-to-date release notes can be found in the GitHub repository⁵.

2.5.1.2 Future plans

Future extensions to itwinai in terms of features and functionalities will be based on adoption requirements identified through integration efforts with new use-cases. Furthermore, AI-centric pipelines defined with tools such as Kubeflow, Airflow and others will be continuously adopted in the course of the project. Support for additional distributed learning backends is also envisioned in the future. Also, integration with the interlink package from WP5, workflow composition tool from Task 6.1 and other relevant tasks in WP6 will be performed.

⁵ <https://github.com/interTwin-eu/itwinai/releases>



3 Summary of integration status

At the level of the infrastructure integration (WP5) several activities are taking place in WP6. The Big Data analytics task (T6.4) has been testing KubeFlow and Airflow instances in the JSC cloud that have provided access to their cloud platform. Both tests are still in progress and in a good shape, which made possible several improvements in the templates and artefacts.

Actually, the KubeFlow instance is also being used by T6.5 to test the itwinai prototype. In particular, itwinai released version was encapsulated within a docker container to facilitate seamless integration. Tests were conducted using the WP5 interlink prototype, interLink⁶. WP5 provided access to a Kubernetes cluster located at JSC. A pod was instantiated with an MLFlow server alongside the itwinai module. Subsequently, it was deployed to the Kubernetes cluster, initiating a series of tests to assess offloading capabilities to the compute node. A KubeFlow instance, deployed at the level of the task of Big Data Analytics, is used by itwinai as a testbed for exploring an easily accessible user interface, based on Jupyter notebooks to the itwinai system.

The integration with the real-time tools, notably the integration process with OSCAR (T6.1), is carried out in conjunction with WP5. A test scenario involves the arrival of new data triggering OSCAR to launch a Kubernetes pod containing the data analysis container. Interlinking operations are then offloaded to the high-performance computing (HPC) resources.

Regarding the integration with the project digital twin examples, following the release of the initial stable version of itwinai, integration efforts extended beyond the initial testing use case. Specific integration meetings were scheduled with partners from CERN, VIRGO, CMCC, CERFACS, and EURAC use cases to reaffirm requirements and outline integration strategies. As of the latest update, the integration with the CERN use case has been successfully completed. Progress has been made in the integration process with Virgo and CMCC, although it has not yet reached full completion. Initial integration steps with CERFACS and EURAC have been initiated, marking the early stages of the integration process. In particular also the quality assessment module (T6.2) is being tested using pytest and newly developed SQAaaS GitHub action.

Regarding Data Fusion (T6.3), most data fusion is currently specific to work happening in the digital twin prototypes defined by WP4 and implemented in specific thematic modules developed in WP7. There's the need to understand which thematic modules / parts of thematic modules can be promoted to "core". Currently, most need for data integration has been identified in the environmental related digital twins. Multiple data sources of different types of raster and vector data are used and combined for both physical based modelling and data driven models. A concrete planning can only be done here, just after the first prototype implementations of those modules become available.

⁶ <https://github.com/interTwin-eu/interLink>

