interTwin logo


**D7.9 DTE Thematic modules development and integration report**

**Status: Under EC Review**

**Dissemination Level: Public**

|  |  |
| --- | --- |
| Abstract | |
| **Key Words** | Digital Twins, Thematic Modules, Radio Astronomy, Gravitational Waves, High Energy Physics, Environmental Monitoring, Climate Research, Machine Learning, Development, Integration, Software Release, Open-source |
| interTwin co-designs and implements a prototype of an interdisciplinary Digital Twin Engine (DTE), an open-source platform integrating software components for modelling and simulation to support application-specific Digital Twins (DTs). Work Package 7 contributes to this effort by developing domain-specific software components, referred to as thematic modules, aligned with the use cases defined in WP4.  This final report provides an overview of the development status, integration, and maturity of the thematic modules across both the environmental and physics domains. It documents the final implemented functionalities and describes how the modules interface with the DTE infrastructure developed in WP5 and the core services from WP6. The report consolidates the progress made throughout the project and sets the foundation for the deployment and reuse of the thematic modules within a federated DTE ecosystem. | |

|  |  |  |  |
| --- | --- | --- | --- |
| Document Description | | | |
| **D7.9 DTE Thematic modules development and integration report** | | | |
| **Work Package number 7** | | | |
| **Document type** | Deliverable | | |
| **Document status** | UNDER EC REVIEW | **Version** | 1.0 |
| **Dissemination Level** | Public | | |
| **Copyright Status** | This material by Parties of the interTwin Consortium is licensed under a [Creative Commons Attribution 4.0 International License](http://creativecommons.org/licenses/by/4.0/). | | |
| **Lead Partner** | EODC | | |
| **Document link** | [**https://documents.egi.eu/document/4098**](https://documents.egi.eu/document/4098) | | |
| **DOI** | [**https://zenodo.org/records/16574058**](https://zenodo.org/records/16574058) | | |
| **Author(s)** | * Charis Chatzikyriakou (EODC) * Gaurav Sinha Ray (CSIC) * Javad Komijani (ETHZ) * Isabel Campos (CSIC) * Yurii Pidopryhora (MPG) * Sara Vallero (INFN) * Francesco Sarandrea (INFN) * Lorenzo Asprea (INFN) * Donatello Elia (CMCC) * Marco De Carlo (CMCC) * Shahbaz Alvi (CMCC) * Sandro Fiore (UNITN) * Bjorn Backeberg (DELTARES) * Willem Tromp (DELTARES) * Iacopo Ferrario (EURAC) * Suriyah Dhinakaran (EURAC) * Michele Claus (EURAC) * Alexander Jacob (EURAC) * Christian Pagé (CERFACS) * Atef Ben Asser (IPSL) * Mathieu Vrac (IPSL) * Matthias Schramm (TUW) * Clay Harrison (TUW) * Vera Maiboroda (CNRS) * Sofia Vallecorsa (CERN) * Matteo Bunino (CERN) | | |
| **Reviewers** | * Mario David (LIP) * Thomas Geenen (ECMWF) | | |
| **Moderated by:** | * Charis Chatzikyriakou (EODC) * Andrea Anzanello (EGI) | | |
| **Approved by** | Andrea Manzi (EGI) on behalf of TCB | | |

|  |  |  |  |
| --- | --- | --- | --- |
| Revision History | | | |
| **Version** | **Date** | **Description** | **Contributors** |
| V0.1 | 12/05/2025 | Created template | Andrea Manzi (EGI), Charis Chatzikyriakou (EODC) |
| V0.2 | 10/07/2025 | Full draft ready for internal review | All authors |
| V0.3 | 25/07/2025 | Reviewers’ comments addressed, ready for TCB review. | All authors |
| V0.4 | 28/07/2025 | TCB reviewer’s comments addressed, ready for QA. | All authors |
| **V1.0** |  | **Final** |  |

|  |  |
| --- | --- |
| Terminology / Acronyms | |
| **Term/Acronym** | **Definition** |
| AI | Artificial Intelligence |
| API | Application Programming Interface |
| BIC | Bayesian Information Criterion |
| CEMS | Copernicus Emergency Mapping Service |
| CeCill-C | CEA CNRS INRIA Logiciel Libre |
| CLIC | Compact Linear Collider |
| CMIP6 | Coupled Model Intercomparison Project, Phase 6 |
| CNN | Convolutional Neural Network |
| COG | Cloud Optimized GeoTIFF |
| CSV | Comma Separated Value |
| CVAE | Convolutional Variational Auto-Encoder |
| CWL | Common Workflow Language |
| DAG | Directed Acyclic Graph |
| DestinE | Destination Earth |
| DID | Data Identifier |
| DNN | Deep Neural Network |
| DoA | Description of Action |
| DT | Digital Twin |
| DTE | Digital Twin Engine |
| ECMWF | European Centre for Medium-Range Weather Forecasts |
| EO | Earth Observation |
| ERA5 | ECMWF Reanalysis v5 |
| ESGF | Earth System Grid Federation |
| FAIR | Findable, Accessible, Interoperable, and Reusable |
| FC | File Catalogue |
| FESOM | Finite-Element/volumE Sea ice-Ocean Model |
| FIAT | Fast Impact Assessment Tool |
| FTS | File Transfer Services |
| GAN | Generative Adversarial Network |
| GenNN | Generative Neural Network |
| GNN | Graph Neural Network |
| GNU AGPLv3 | GNU Affero General Public License version 3 |
| GPLv2 | GNU General Public License version 2 |
| GPU | Graphics Processing Unit |
| GW | Gravitational Wave |
| HDF | Hierarchical Data Format |
| HEP | High Energy Physics |
| HL-LHC | High Luminosity - Large Hadron Collider |
| HPAR | Harmonic Parameters |
| HPC | High-Performance Computing |
| HPO | Hyperparameter Optimization |
| HTTP | Hypertext Transfer Protocol |
| IBTrACS | International Best Track Archive for Climate Stewardship |
| icclim | Index Calculation for CLIMate |
| ILDG | International Lattice Data Grid |
| JSON | JavaScript Object Notation |
| LSTM | Long short-term memory |
| MC | Monte Carlo |
| MDC | Metadata Catalogue |
| MCMC | Markov Chain Monte Carlo |
| ML | Machine Learning |
| ML-PPA | Machine Learning-based Pipeline for Pulsar Analysis |
| PLIA | Projected Local Incidence Angle |
| PoC | Proof of Concept |
| QCD | Quantum Chromodynamics |
| QED | Quantum Electrodynamics |
| RA2CE | Resilience Assessment and Adaptation for Critical infrastructurE |
| RSE | Rucio Storage Elements |
| RFI | Radio Frequency Interference |
| SE | Storage Endpoint |
| SEAS5 | ECMWF seasonal forecasts |
| SFINCS | Super-Fast INundation of CoastS |
| SNR | SIgnal-to-Noise Ratio |
| SQL | Structured Query Language |
| STAC | SpatioTemporal Asset Catalog |
| TC | Tropical Cyclone |
| TCB | Technical Coordination Board |
| VGG | Visual Geometry Group (a standard deep CNN) |
| WP | Work Package |
| YAML | Yet Another Markup Language |

Terminology / Acronyms: [**https://confluence.egi.eu/display/EGIG**](https://confluence.egi.eu/display/EGIG)

**Table of Contents**

[Executive summary 10](#_Toc204700908)

[1 Introduction 11](#_Toc204700909)

[1.1 Scope 11](#_Toc204700910)

[1.2 Document Structure 11](#_Toc204700911)

[2 Overview of thematic modules 13](#_Toc204700912)

[2.1 T7.1 Lattice QCD simulations and data management 13](#_Toc204700913)

[2.1.1 Machine Learning for Lattice Simulations with normflow 13](#_Toc204700914)

[2.1.2 The Lattice Data Lake 15](#_Toc204700915)

[2.2 T7.2 Noise simulation for radio astronomy 17](#_Toc204700916)

[2.2.1 Use-case description 17](#_Toc204700917)

[2.2.2 Machine Learning-based Pipeline for Pulsar Analysis (ML-PPA) 18](#_Toc204700918)

[2.3 T7.3 GenNN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for gravitational waves 20](#_Toc204700919)

[2.3.1 Use-case description 20](#_Toc204700920)

[2.3.2 High-level architecture of the DT implementation 21](#_Toc204700921)

[2.3.3 The Training DT subsystem 22](#_Toc204700922)

[2.3.4 The Inference DT subsystem 23](#_Toc204700923)

[2.3.5 The Virgo Data Lake 23](#_Toc204700924)

[2.4 T7.4 Climate analytics and data processing 23](#_Toc204700925)

[2.5 T7.5 Earth Observation Modelling and Processing 24](#_Toc204700926)

[2.5.2 High-level Architecture of the DT Implementation 25](#_Toc204700927)

[2.5.4 Thematic Modules Integration 26](#_Toc204700928)

[2.6 T7.6 Hydrological model data processing 27](#_Toc204700929)

[2.6.1 Use-case Description 27](#_Toc204700930)

[2.6.2 High-level Architecture of the DT Implementation 27](#_Toc204700931)

[2.6.3 Modelbuilder Workflow 28](#_Toc204700932)

[2.6.4 Scenario Workflow 29](#_Toc204700933)

[2.6.5 Data Integration and Infrastructure 29](#_Toc204700934)

[2.6.6 Summary 30](#_Toc204700935)

[2.7 T7.7 Fast particle detector simulation 30](#_Toc204700936)

[2.7.1 Use-case description 30](#_Toc204700937)

[2.7.2 High-level Architecture of the DT Implementation 30](#_Toc204700938)

[2.8 Summary of supported Digital Twins 33](#_Toc204700939)

[3 Thematic modules and final development and integration activities 35](#_Toc204700940)

[3.1 T7.1 Lattice QCD simulations and data management 35](#_Toc204700941)

[3.1.1 openQxD 35](#_Toc204700942)

[3.1.2 normflow 36](#_Toc204700943)

[3.1.3 Functionalities developed since the last release 36](#_Toc204700944)

[3.1.4 Integrations with other DTE components 36](#_Toc204700945)

[3.1.5 Integrations with DT Applications 37](#_Toc204700946)

[3.1.6 Pilots and testing activities 37](#_Toc204700947)

[i. Both components have been piloted on the VEGA EuroHPC GPU partition resources dedicated to the project. 37](#_Toc204700948)

[3.2 T7.2 Noise simulation for radio astronomy 37](#_Toc204700949)

[3.2.1 PulsarDT 37](#_Toc204700950)

[3.2.2 PulsarDT++ 38](#_Toc204700951)

[3.2.3 PulsarRFI\_Gen 39](#_Toc204700952)

[3.2.4 PulsarRFI\_NN 40](#_Toc204700953)

[3.2.5 Functionalities developed since the last release 40](#_Toc204700954)

[3.2.6 Integrations with other DTE components 41](#_Toc204700955)

[3.2.7 Integrations with DT Applications 41](#_Toc204700956)

[3.2.8 Pilots and testing activities 41](#_Toc204700957)

[3.3 T7.3 GenNN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for gravitational waves 41](#_Toc204700958)

[3.3.1 GlitchFlow 41](#_Toc204700959)

[3.3.2 ANNALISA 42](#_Toc204700960)

[3.3.3 Functionalities developed since the last release 43](#_Toc204700961)

[3.3.4 Integrations with other DTE components 43](#_Toc204700962)

[3.3.5 Integrations with DT Applications 44](#_Toc204700963)

[3.3.6 Pilots and testing activities 44](#_Toc204700964)

[3.4 T7.4 Climate analytics and data processing 44](#_Toc204700965)

[3.4.1 ML TC detection 44](#_Toc204700966)

[3.4.2 ML4Fires 45](#_Toc204700967)

[3.4.3 eddiesML 46](#_Toc204700968)

[3.4.4 xtclim 47](#_Toc204700969)

[3.4.5 downscaleML 48](#_Toc204700970)

[3.4.6 emergence.compound 49](#_Toc204700971)

[3.4.7 Esgpull\_rucio 50](#_Toc204700972)

[3.4.8 Functionalities developed since the last release 51](#_Toc204700973)

[3.4.9 Integrations with other DTE components 52](#_Toc204700974)

[3.4.10 Integrations with DT Applications 53](#_Toc204700975)

[3.4.11 Pilots and testing activities 53](#_Toc204700976)

[3.5 T7.5 Earth Observation Modelling and Processing 54](#_Toc204700977)

[3.5.1 openeo-processes-dask 54](#_Toc204700978)

[3.5.2 openeo-pg-parser-networkx 54](#_Toc204700979)

[3.5.3 raster-to-stac 55](#_Toc204700980)

[3.5.4 dask-flood-mapper 56](#_Toc204700981)

[3.5.5 Functionalities developed since the last release 57](#_Toc204700982)

[3.5.6 Integrations with other DTE components 57](#_Toc204700983)

[3.5.7 Integrations with DT Applications 57](#_Toc204700984)

[3.5.8 Pilots and testing activities 58](#_Toc204700985)

[3.6 T7.6 Hydrological model data processing 58](#_Toc204700986)

[3.6.1 FloodAdapt 58](#_Toc204700987)

[3.6.2 HydroMT-SFINCS 59](#_Toc204700988)

[3.6.3 HydroMT-FIAT 60](#_Toc204700989)

[3.6.4 SFINCS 61](#_Toc204700990)

[3.6.5 Delft-FIAT 62](#_Toc204700991)

[3.6.6 WFLOW.jl 63](#_Toc204700992)

[3.6.7 HydroMT-WFLOW 65](#_Toc204700993)

[3.6.8 RA2CE 66](#_Toc204700994)

[3.6.9 Hython Wflow\_SBM Hydrological Model 67](#_Toc204700995)

[3.6.10 Functionalities developed since the last release 69](#_Toc204700996)

[3.6.11 Integrations with other DTE components 69](#_Toc204700997)

[3.6.12 Pilots and testing activities 70](#_Toc204700998)

[3.7 T7.7 Fast particle detector simulation 70](#_Toc204700999)

[3.7.1 3DGAN and CaloINN 70](#_Toc204701000)

[3.7.2 Functionalities developed since the last release 71](#_Toc204701001)

[3.7.3 Integrations with other DTE components 72](#_Toc204701002)

[3.7.4 Integrations with DT Applications 72](#_Toc204701003)

[3.7.5 Pilots and testing activities 72](#_Toc204701004)

[4 Conclusions 73](#_Toc204701005)

[5 References 75](#_Toc204701006)

**List of Figures**

[Figure 1 Graphical representation of the Normalising Flows method 13](#_Toc204701007)

[Figure 2 Module Integration Diagram for the Lattice QCD use case 14](#_Toc204701008)

[Figure 3 Schema depicting the flow of control when accessing the Lattice Datalake. 16](#_Toc204701009)

[Figure 4 General outline of the DT structure. 19](#_Toc204701010)

[Figure 5 Operation Diagram of the ML-PPA 20](#_Toc204701011)

[Figure 6 High-level architecture of the DT 21](#_Toc204701012)

[Figure 7 System Context diagram of the Virgo Interferometer DT 22](#_Toc204701013)

[Figure 8 High-level architecture of the Alpine Drought Early Warning DT 25](#_Toc204701014)

[Figure 9 High-level architecture of the flood risk DTs for post-flood analysis (top) and climate impact assessment (bottom) 28](#_Toc204701015)

[Figure 10 System Context diagram of the Fast Detector Simulation DT 31](#_Toc204701016)

**List of Tables**

[Table 1 Summary of supported Digital Twins 33](#_Toc204701017)

# Executive summary

Work Package 7 (WP7) of the interTwin project is dedicated to the design, development, and integration of thematic modules that provide domain-specific functionalities to the Digital Twin Engine (DTE). These modules serve as reusable software components supporting the implementation of Digital Twin (DT) applications in both the environmental and physics domains, as defined in WP4.

This final report is a collective document written by the scientists who developed the thematic modules and consolidates the progress made in the development and integration of these modules throughout the project. It builds upon previous deliverables (D7.1 [[**R20**](#_References)], D7.2 [[**R2**](#_References)], D7.3 [[**R1**](#_References)], D7.4 [[**R3**](#_References)], D7.5 [[**R21**](#_References)], D7.6 [[**R4**](#_References)], D7.7 [[**R22**](#_References)] and D7.8 [[**R16**](#_References)]), offering a comprehensive overview of the final version of the modules, including newly added components developed to meet evolving requirements. The report captures the technical maturity of the modules, their alignment with scientific workflows, and their integration into the DTE platform in collaboration with WP5 (infrastructure) and WP6 (core services).

In the environmental domain, 20 thematic modules have been developed within the tasks Climate Analytics and Data Processing (T7.4), Earth Observation Modelling and Processing (T7.5), and Hydrological Model Data Processing (T7.6) to support the six environmental DT applications. These modules address the data ingestion, transformation, analysis, and model coupling and other needs of these applications. All modules are released as open-source software and are accompanied by documentation, metadata, and release notes to ensure transparency, usability, and reproducibility.

In the physics domain, nine thematic modules have been implemented within the tasks Lattice QCD simulations and data management (T7.1), Noise simulation for radio astronomy (T7.2), GenNN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for gravitational waves (T7.3) and Fast particle detector simulation (T7.7) to support the four DT applications from the physics domain. Each module’s functionality is aligned with the needs of its respective application, with a focus on efficient data handling, domain-specific processing, and scalability. The report includes technical summaries, licensing information, and details on software readiness.

The report also provides a summary of completed - and in a few cases only ongoing - integration efforts, highlighting interactions with the DTE core and infrastructure components.

# Introduction

## Scope

The deliverable provides the final report on the development and integration status of the thematic modules designed to support the Digital Twin applications within the interTwin project. It builds upon and consolidates all previous WP7 deliverables —particularly D7.1, D7.2, D7.3, D7.4, D7.5, D7.6, D7.7 and D7.8 — offering an updated, comprehensive view across both scientific domains covered by the interTwin DT applications, the Physics and the Environment domains.

A total of 29 thematic modules have been developed and integrated into the DTE framework, 20 of which are within the tasks supporting the environmental DT applications and 9 of which are within the tasks supporting the Physics ones. These modules address a broad range of functionalities required by Digital Twin (DT) applications and span several areas, including:

* High Energy Physics
* Radio Astronomy
* Gravitational waves
* Climate Research
* Environmental Monitoring

Since the previous releases, a number of modules have been further developed and two new modules have been added to meet the evolving needs of the DT applications. Specifically, the newly added modules are the ANNALISA module, by INFN, and the Dask Flood Mapper, by TU Wien. All thematic components have been developed in alignment with the overarching DTE architecture and the co-design process led by WP4.

The thematic modules in this deliverable are described with a focus on their development maturity, current features, integration status, licensing and documentation. It should be noted that a more elaborate description of the integration activities with the respective DT applications can be found in the final WP4 deliverables, D4.7[[1]](#footnote-1) and D4.8[[2]](#footnote-2) that will become publicly available at the end of August 2025.

## Document Structure

The document is organised into three main sections:

* [**Section 2**](#_Overview_of_thematic) provides an overview of the thematic modules, structured by task (T7.1–T7.7). Each subsection includes the related use case context, architectural components, and high-level workflows of the modules. This section also highlights the connection between the modules and their corresponding DT applications.
* [**Section 3**](#_Thematic_modules_and) gives a detailed account of the final developments and integration activities for each thematic module, including the:  
  + functionalities developed since the last release
  + integration with other DTE components and with DT applications
  + pilot implementations and testing activities
* Finally, [**Section 4**](#_Conclusions) presents the main conclusions and lessons learned across the domain-specific efforts in WP7, with a particular emphasis on integration outcomes and readiness for exploitation within DT applications.

# Overview of thematic modules

## T7.1 Lattice QCD simulations and data management

Lattice QCD involves the study of the properties of Quantum Chromodynamics in the low energy limit, where perturbation theory breaks down and numerical approaches are required. Within interTwin two parallel and complementary tracks have been implemented that address the practical and theoretical challenges of Lattice QCD simulations. These are the practical challenges of storing and moving the ever-increasing amounts of data associated with traditional large-scale HPC simulations, and the theoretical challenge of exploring, at the proof-of-concept level, the extent to which contemporary Machine-Learning techniques can make lattice simulations more efficient.

### Machine Learning for Lattice Simulations with normflow[[3]](#footnote-3)

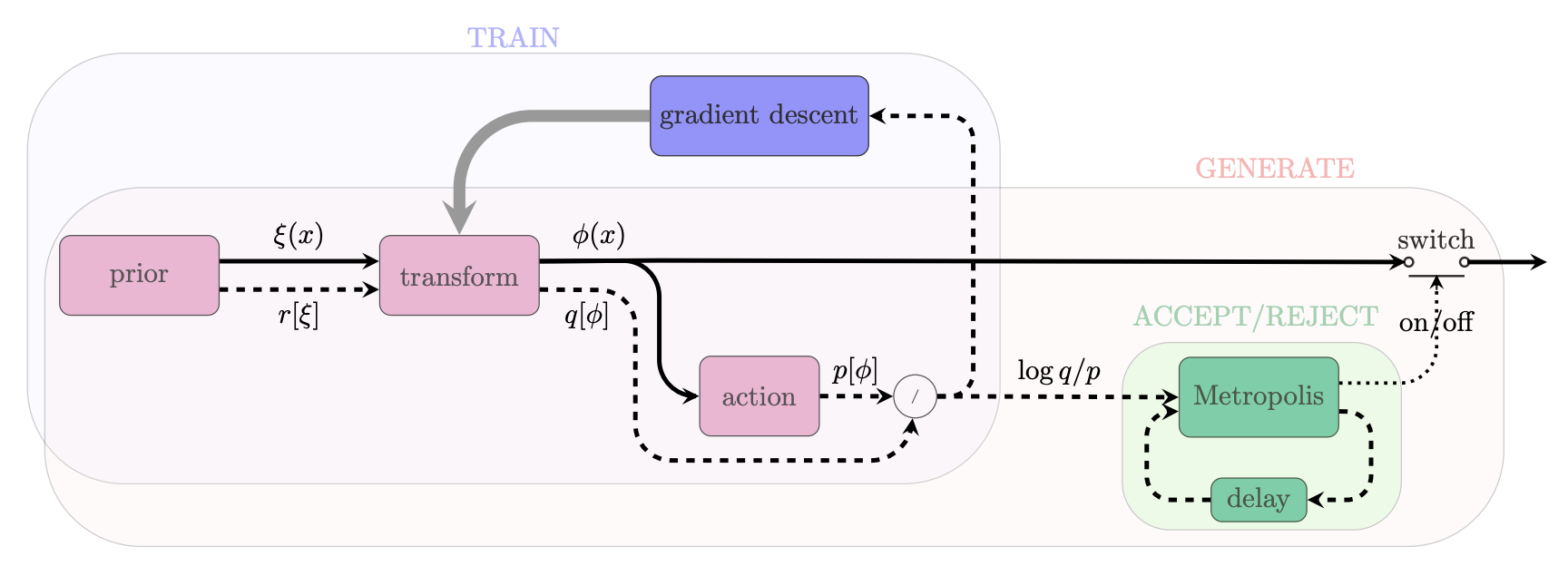
The efficiency of general-purpose Monte Carlo algorithms decreases dramatically when simulations take place near critical points due to critical slowing down. This is a general phenomenon in physics simulations related to phase transitions. This loss of efficiency also occurs in the high-resolution Lattice simulations needed for continuum extrapolations. Researchers want to be able to carry out simulations in regions of parameter space where topology freezing and long autocorrelations currently stymie progress.

Figure Graphical representation of the Normalising Flows method including a correcting accept/reject step to account for the fact that the model cannot be perfectly trained.

Through the development of the pytorch-based python package normflow, we have shown that Machine Learning can be used for field configuration generation with scalar theories and certain gauge theories on small lattices of dimensionality up to and including four [[**R9**](#_References)]. Moreover, it is being developed to handle the more complicated family of SU(3) gauge theories, this being an important step towards a ML lattice simulation of QCD [[**R10**](#_References)]. **Figure 2** outlines schematically the typical workflows of a developer and a user of normflow.

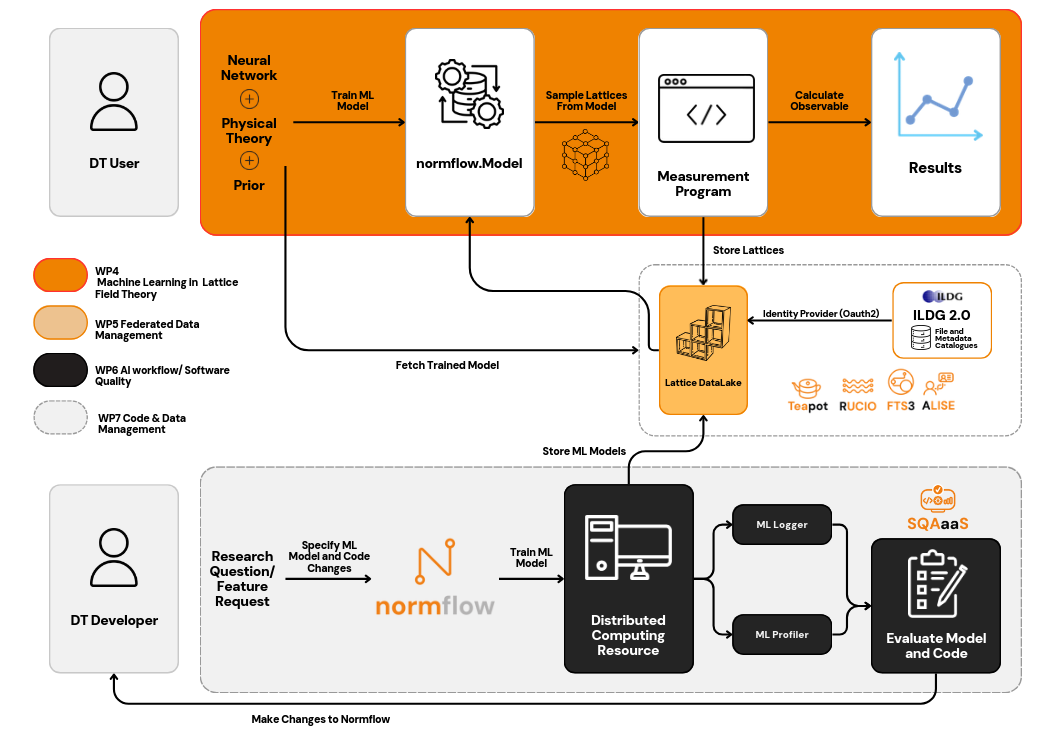


Figure Module Integration Diagram for the Lattice QCD use case

Improvements to the normflow software development workflow have been made by the integration of the SQAaaS module developed by WP6.2, with the progress of this integration being tracked in the corresponding WP4/7 deliverables. Software quality assurance in this context means making sure software packages developed for scientific research, like normflow, adhere to research software best practices, such as being licensed with an Open Source Initiative-approved licence. Currently the public version of normflow is credited with the Silver SQAaaS badge and this is displayed prominently on normflow’s public repository webpage. Since the previous deliverable, we have implemented the first automated tests of normflow, using the pytest package and the bash testing framework. Automated testing was demoed by WP6.2 at the 2024 IBERGRID conference[[4]](#footnote-4). We continue to add tests and are looking at an alternative, more flexible, way of automating the testing with SQAaaS.

### The Lattice Data Lake

Lattice QCD simulations are executed at scale on HPC systems that are controlled by a batch scheduler (such as SLURM[[5]](#footnote-5)). A typical workflow involves the generation of lattice field configurations, the measurement of an observable of interest over those configurations, and the statistical analysis of those measurements. All these steps, but especially the generation of configurations, can be highly computationally intensive and often require lots of disk space.

The openQxD simulation software is a highly optimised C code designed to simulate QCD and QCD+QED theories on a lattice. It is similar to many other traditional Markov Chain Monte Carlo (MCMC) lattice codes used across the field to conduct particle physics research. It is available on Gitlab[[6]](#footnote-6) and is described at length in the literature [[[**R**](https://docs.google.com/document/d/17cu4vImdJHxk-sw41nlC1NtW2UUv7Qk1/edit#heading=h.2dlolyb)**5**](#_References)]. A concise description of the software is given in section 3.1 of D7.2 [[**R2**](#_References)] along with links to further technical documentation. It is under active development though this work is not being done as part of the interTwin project.

In previous deliverables we described some of the issues encountered by lattice researchers when trying to store and access their data [[**R3**](#_References)]. We argued that lattice configurations should be made more easily available to the members of a collaboration. It was realised early on that the use of federated identities and group-based access control would be crucial to achieving this goal of easier access in a controlled way. The Data Lake framework proposed and developed by WP5 followed naturally. In this framework, the members of a collaboration would have group-access enabled read permission for their data, while a subset of the collaboration, those in charge of generating configurations, would also have write permission. In D7.4 we described our efforts relating to testing and benchmarking the DataLake prototype with real and toy lattice data [[**R3**](#_References)]. After providing feedback to the DataLake developers it was decided that the Lattice group should get its own Lattice Data Lake in order to satisfy its particular read/write permission specifications.

It was also decided that a phased approach to the Lattice Data Lake rollout would be preferable. The first phase involved opening an FTS connection and transferring data between the DESY-Zeuthen storage endpoint and an endpoint at CESGA. The required FTS server has been updated and is now able to accept ILDG tokens. This is important as the Lattice Data Lake will use the ILDG as its identity and access manager. The ILDG can support multiple locations, recorded as URLs, for each piece of data. We outlined a possible schema for the Data Lake URLs that would be recorded in the ILDG in D7.6 [[**R4**](#_References)].



Figure Schema depicting the flow of control when accessing the Lattice Datalake. In this example the institutional Identity Provider (IdP) is CSIC, but it could be any other IdP trusted by the ILDG.

Thanks to the efforts of WP5 and the ILDG, there is now a functional prototype of the “Lattice Datalake” with three storage endpoints (SEs); two at DESY and one at CESGA. **Figure 3** illustrates the flow of control when accessing the datalake, and highlights how a user, once authenticated, could also access the ILDG’s File (FC) and Metadata (MDC) Catalogues.

Lattice Data Lake access works as follows [[**R17**](#_References)]:

* A client queries the ILDG for the location of some files it wishes to access.
* ILDG authorises the request, verifying the user is allowed to read the files.
* ILDG returns the location of the files along with the token(s) needed for their access.
* For each file the client contacts the storage to request access and supplies the corresponding token.
* The storage verifies the token and provides the requested file.

In D7.8 we described our progress towards our goal of extending the ILDG catalogue to support a Lattice Data Lake as a possible source of data [[**R16**](#_References)]. The phased rollout of the Data Lake continues and the focus now is on a new endpoint at CESGA which will replace the current endpoint. The WP5 group and the CESGA administrators are working on getting teapot[[7]](#footnote-7) access to CESGA’s HPC storage. This represents a major potential improvement on the current situation because this SE will be directly accessible from the compute nodes. This will enable direct integration with normflow and itwinai by opening up the possibility of immediately writing trained models and other data directly to the Lattice Data Lake once normflow finishes training. This activity is expected to be completed by the end of the project.

## T7.2 Noise simulation for radio astronomy

### Use-case description

As outlined in the previous reports [[**R2**, **R4**, **R16**](#_References)], this task is designed to be instrumental in solving a big problem that is about to arise in modern observational astronomy in general and radio astronomy in particular, and to become one of the largest issues in the whole field: the problem of data overflow. Previous generations of telescopes typically produced no more than a few petabytes of data per year; thus, the raw data was generally kept either indefinitely or long enough for the science team to reduce and analyse it, and then approve the deletion, which meant several months or even years. With the arrival of the new so-called Square Kilometre Array[[8]](#footnote-8) "pathfinders", such as South African MeerKAT[[9]](#footnote-9) or Australian ASKAP[[10]](#footnote-10), the data acquisition rate increases enormously, these tools can easily produce several petabytes of raw data per week[[11]](#footnote-11). No current astronomical institution can handle keeping such volumes of data even for a month or employ a team of experts large enough to quickly process it or sort through it manually. Thus, it is crucial to develop automated decision-making systems that can sort through the raw data in real or near-real time (since telescopes usually have downtime due to maintenance or source availability, the data can be pooled for short periods of order of days) and separate the data flow into the scientifically important data that must be kept while the rest that can be safely deleted.

Another reason to be able to automatically sort through the incoming data is that modern radio astronomy is increasingly interested in transient sources. Previously sources had to be observed for long periods of time to be able to achieve the necessary signal to noise ratio, thus it was possible to observe reliably or even discover at all only permanent or fast periodic[[12]](#footnote-12) sources like pulsars. Since the new telescopes are much more sensitive, they can systematically probe the transient radio sky, which currently is generally unknown. Such studies are very important, since it is believed that the transients[[13]](#footnote-13) result from very far and enormously energetic exotic events (like a collapsing supermassive star) that may provide essential clues for the areas of physics that cannot be studied experimentally in any other way, e.g., quantum gravity. An automated expert system can help with this: if something like a transient source (or unusual in general) signature is found in the data flow, it can immediately trigger the "target of opportunity" mode of observation for the detected anomaly, and alert the scientists on duty, who would decide the best course of further action. This will also allow us to easily organise concerted efforts of observing rare important sources by several instruments, covering a range of wavelengths, e.g., combining Earth-based radio observations with space-based optical and X-ray observations — it is already done today, but with typical response times very far from ideal[[14]](#footnote-14).

Pulsars are ideal test subjects for this task since they reliably produce periodic bursts of scientifically significant data with certain variability in signal strength and other parameters. However, because of their nature “silent” most of the time, a telescope observing a pulsar mostly records either an "empty" data stream, i.e., only the noise, or some sort of RFI due to artificial or natural electro-magnetic phenomena unrelated to space.

The third reason for this task is that current common radio astronomy software tools are inadequate, they are computationally slow and handle parallelization poorly. For the tasks at hand, we are building tools that can be efficiently run on modern HPC clusters, with scalability to at least hundreds of cores. It is connected to the main task of the ML data classification system in a way that, although the classification system itself will be run on ordinary observatory computers embedded in a telescope’s data acquisition system, the training of new models before each new type of observation, which is the most computationally intensive task, will have to be performed on supercomputers.

To be able to detect special and important events in the data, one must first understand well the regular and mundane features of the data stream that in radio astronomy translates into noise and radio-frequency interference (RFI).

### Machine Learning-based Pipeline for Pulsar Analysis (ML-PPA)

As previously reported [[**R3**, **R16**](#_References)], motivated by these points we developed a framework for extracting pulsar signals from radio-astronomical observatory data streams, under the designation of ML-PPA (Machine Learning-based Pipeline for Pulsar Analysis): a ML-based data-labelling system that reads the data flow coming from a real telescope observing a pulsar. An important separate component is a DT of an astronomical source-telescope system, able to generate synthetic output signals identical to the data recorded by a real telescope. The resulting DT-generated data is to be used to train the ML data-classification tool. The DT is physics-based: a set of control parameters will allow adjustment of the output to different sources, detection instruments, and observing conditions.

Four modules are being developed under the umbrella designation of ML-PPA:

* [**PulsarDT**](https://gitlab.com/ml-ppa/pulsardt)
* [**PulsarDT++**](https://gitlab.com/ml-ppa/pulsardtpp)
* [**PulsarRFI\_Gen**](https://gitlab.com/ml-ppa/pulsarrfi_gen)
* [**PulsarRFI\_NN**](https://gitlab.com/ml-ppa/pulsarrfi_nn)

**PulsarDT**: physics-based DT, simulation of the propagation of pulsar signals from the source to antennas (**Figure 4**) and generation of synthetic data – written in Python, to test algorithmic strategies for physical models of pulsars, interstellar medium, telescopes, interference, and noise.

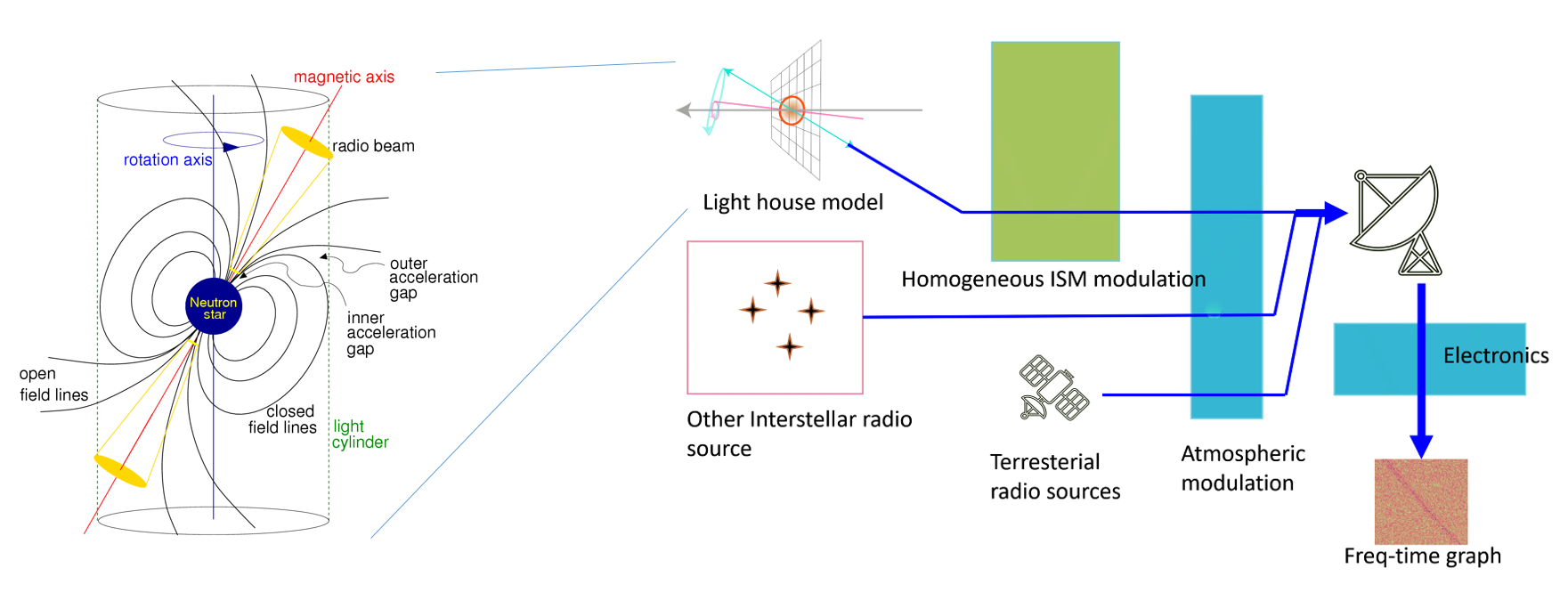


Figure General outline of the DT structure: modelling the astrophysical source (pulsar), transmission of the signal through the interstellar matter, receiving and processing by a radio telescope, adding sources of both natural and artificial interference and noise.

**PulsarDT++**: PulsarDT is implemented in C++ in order to improve its speed and allow for parallelization, easily deployable in a singularity container.

**PulsarRFI\_Gen**: empirical DT, generating “timeframes”, 2D images (time-frequency) of all possible types of telescope output observing a pulsar: pulses (scientifically relevant data), two different types (“narrow” and “broad”) of RFI signals, and “empty” frames, containing only noise. It creates these timeframes by mimicking available real data (based on the geometry of images, noise characteristics etc.) rather than generating them from the physical first principles as PulsarDT does. By using this alternative method it provides comparison for PulsarDT/DT++ and substitutes training data for the ML classifier.

**PulsarRFI\_NN**: the ML classifier. It is a CNN-based tool for the identification of various types of pulsar and RFI signals in the “timeframes”, 2D images (time-frequency).

The general diagram of the intended operation of the ML-PPA and its interaction with other interTwin components is shown in **Figure 5**.

ML-PPA has been tested with real data collected by observing various pulsars with two telescopes: the Effelsberg 100m radio telescope[[15]](#footnote-15) and the above-mentioned MeerKAT array.

A more detailed overview of the state of the project and its full theoretical background can be found in [[**R19**](#_References)].

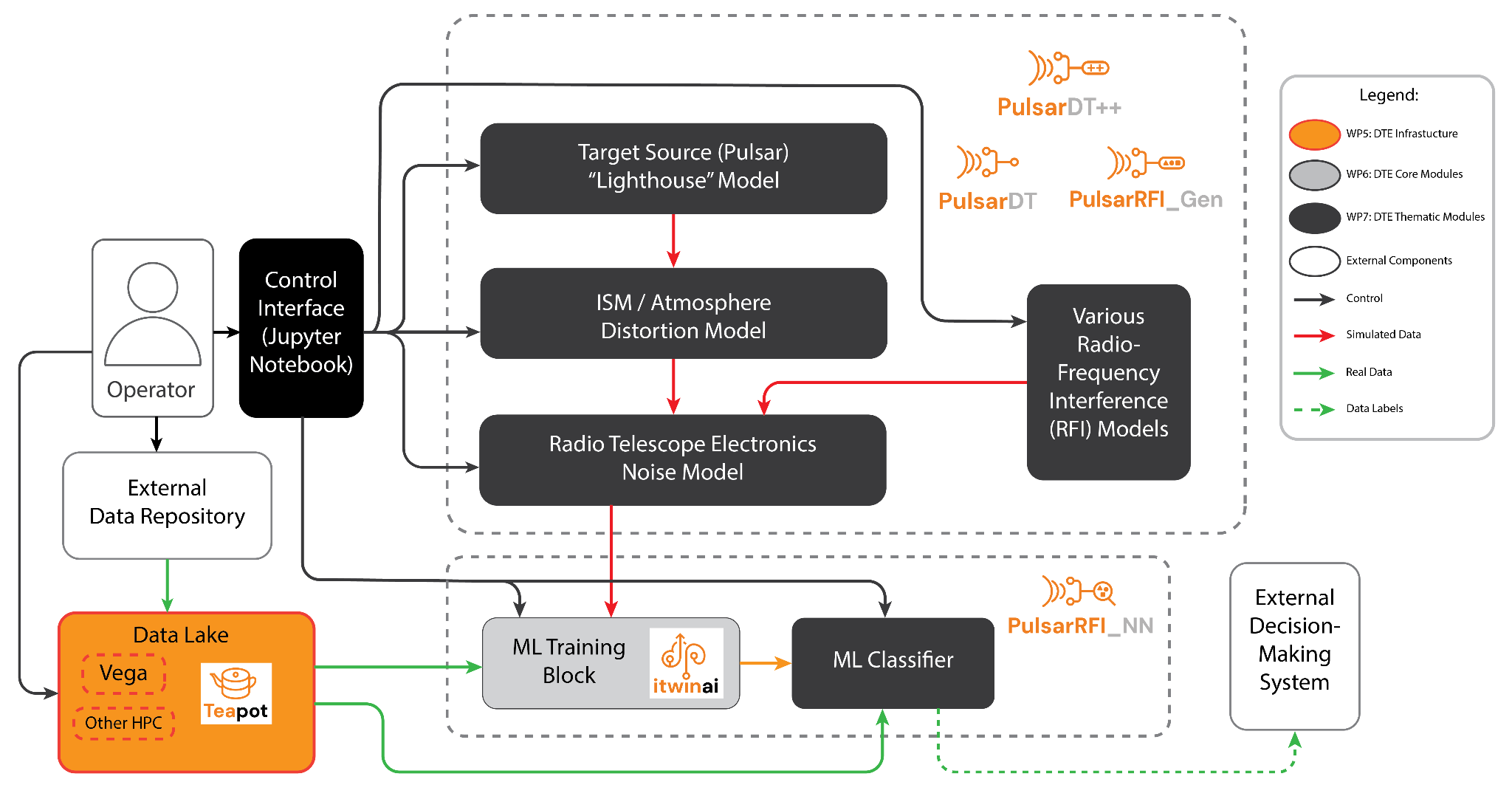


Figure Operation Diagram of the ML-PPA

## T7.3 GenNN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for gravitational waves

### Use-case description

The sensitivity of Gravitational Wave (GW) interferometers is limited by noise. We have been using Generative Neural Networks (GenNNs) to produce a Digital Twin (DT) of the Virgo interferometer to realistically simulate transient noise in the detector. We have used the GenNN-based DT to generate synthetic strain data (a channel that measures the deformation induced by the passage of a gravitational wave). Furthermore, the detector is equipped with sensors that monitor the status of the detector’s subsystems as well as the environmental conditions (wind, temperature, seismic motions) and whose output is saved in the so-called auxiliary channels. Therefore, in a second phase, also from the perspective of the Einstein Telescope, we will use the trained model to characterise the noise and optimise the use of auxiliary channels in vetoing and denoising the signal in low-latency searches, i.e., those data analysis pipelines that search for transient astrophysical signals in almost real time. This will allow the low-latency searches (not part of the DT) to send out more reliable triggers to observatories for multi-messenger astronomy.

**Figure 6** shows the high-level architecture of the DT. Data streams from auxiliary channels are used to find the transfer function of the system producing non-linear noise in the detector output. The output function compares the simulated and the real signals in order to issue a veto decision (to further process incoming data in low-latency searches) or to remove the noise contribution from the real signal (denoising).

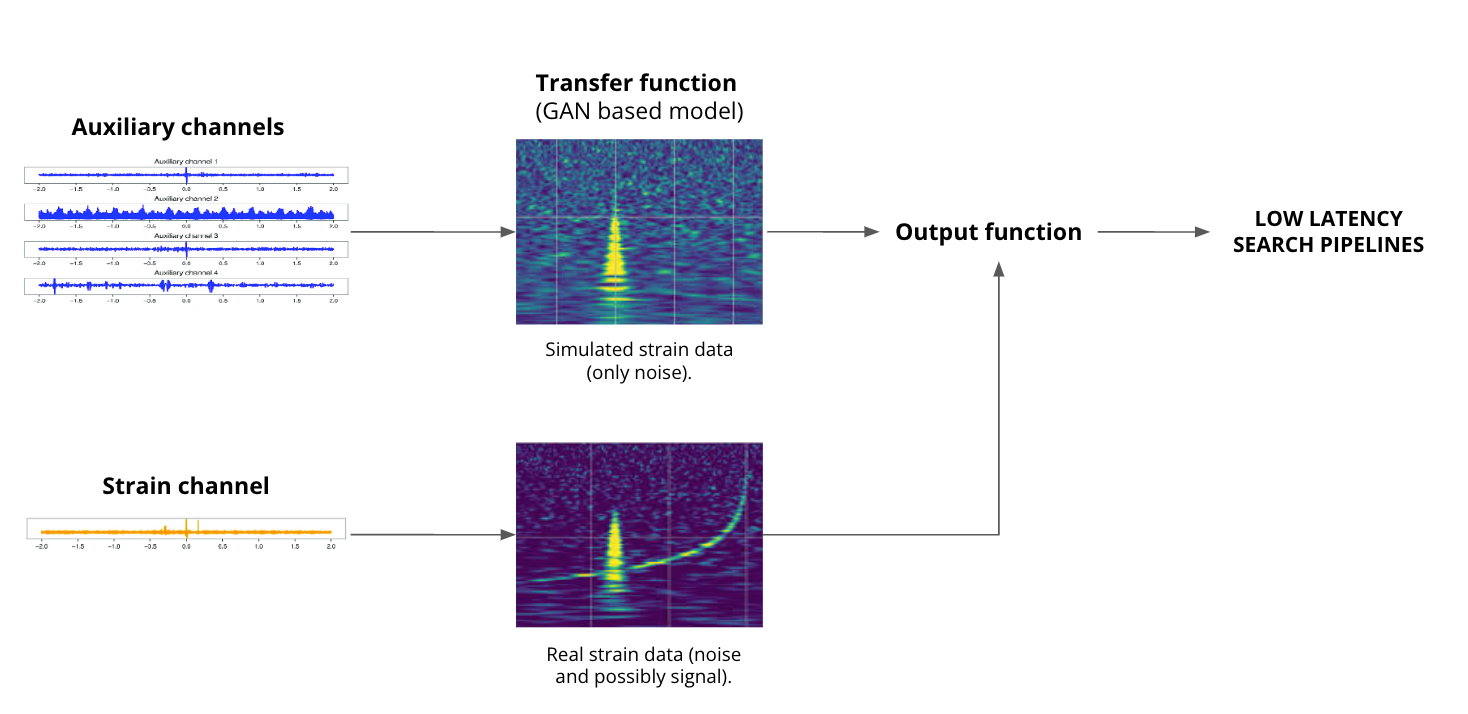


Figure High-level architecture of the DT

### High-level architecture of the DT implementation

**Figure 7** shows the System Context diagram of the DT for the veto and denoising pipeline.   
Two main subsystems characterise the DT architecture: the *Training DT subsystem* and the *Inference DT subsystem.* The Training DT subsystem is responsible for the periodical re-training of the DT model on a buffered subsample of the most recent Virgo data. The DT model needs to be updated to reflect the current status of the interferometer, so continuous retraining of the GenNN needs to be carried out. The Inference DT subsystem is responsible for the low latency vetoing and denoising of the detector’s data stream.  
All modules within both subsystems are implemented as **itwinai**[[16]](#footnote-16)plugins. Itwinai offers several key features that are beneficial to the DT, including distributed training capabilities, a robust logging and model catalogue system, enhanced code reusability, and a user-friendly configuration interface for pipelines.

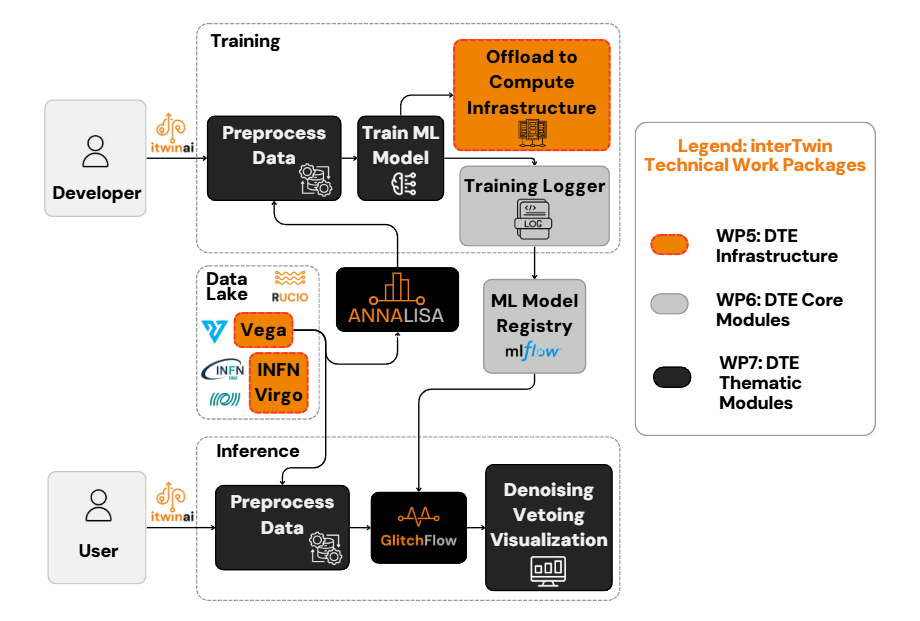


Figure System Context diagram of the Virgo Interferometer DT

### The Training DT subsystem

The Training Subsystem is activated by operators. Its workflow begins with an initial, one-time step: the selection of pertinent channels for network training. This preliminary action is carried out by the **ANNALISA** (Advanced Nonlinear transient-Noise Analyser of Laser Interferometer Sensor Array) module, which makes use of time-frequency domain representation of the data, namely the QTransform, to evaluate correlations among the main and auxiliary channels as a measure of temporally coincident spikes in the energetic content of the signals above a critical threshold.

Following this, the operators preprocess data retrieved from the Virgo Data Lake.

Data preprocessing steps, all available from the ANNALISA module, consist of data resampling, whitening, spectrogram generation, image cropping, and loading into a custom PyTorch dataloader.  
The data loader subsequently feeds a GenNN during training. The final NN architecture chosen for the DT is a Convolutional U-net, featuring residual blocks and attention gates with enhanced skip connections. This design allows for a better capture of data complexity and interdependence. Both the model definition and training are handled by the **GlitchFlow** module.

As the model undergoes training, its learned weights and various performance metrics are systematically logged into a dedicated model registry on MLFlow[[17]](#footnote-17). This ensures that the trained model is not only preserved but also readily accessible for use by the Inference Subsystem. Both the logging process and the offloading to computing infrastructure during training are facilitated by itwinai.

### The Inference DT subsystem

The Inference Subsystem is activated by users, preferably GW detector characterization or data analysis experts. They start by selecting the data for analysis, which then undergoes the same preprocessing steps as those applied during the training phase. Subsequently, a trained model is loaded from the model catalogue and utilized to perform inference on the chosen data. The output of this process comprises "clean" data, ideally free of glitches, and metadata containing veto flagging information, which identifies glitch instances. Both the cleaned data and metadata are logged, offering a complete record of the denoising and vetoing operations.  
The logged details, including images of the real, generated, and cleaned data, are accessible on TensorBoard[[18]](#footnote-18). Metadata containing veto flag information, organized by the GPS time of the analysed data, is also logged. Furthermore, metadata for any data that failed to be cleaned is recorded, including the area and Signal-to-Noise Ratio (SNR) of glitches still visible after cleaning. To access this information, users can launch TensorBoard and navigate through the logged events, which are categorized by run and timestamp, allowing for detailed visualization and analysis of the inference results. The entire pipeline, encompassing data selection, inference, and logging, is configurable via a YAML file, enabling users to specify modules to execute, preprocessing parameters, dataset specifics, network architecture, and paths for saving results.

### The Virgo Data Lake

The transient noise data is being stored in the interTwin Data Lake, which we are managing in synergy with the developers of task 5.1. The Data Lake is managed by the Rucio software, which ensures scalable and efficient data transfer and storage. Specifically, we registered two Rucio Storage Elements (RSEs): one at INFN, where the data is originally stored on tape, and one at the Vega EuroHPC[[19]](#footnote-19) . The RSEs are part of a private Virgo Virtual Organisation (virgo.intertwin.eu), created to restrict data access to only authorised people who are part of the Virgo community. The data is transferred from the former RSE to the latter via a Transfer File System mediated by Rucio. It is then possible to use the data to develop and deploy the different modules directly on Vega, making full use of the computational resources made available by the collaboration.

## T7.4 Climate analytics and data processing

The goal of task 7.4 is to provide a set of thematic software modules for supporting climate/weather-related data processing and data-driven models for climate-based DTs.

* + 1. Use cases

Thematic modules developed in T7.4 have been used to support multiple DT applications from WP4, in particular:

* Tropical Cyclones detection and tracking due to climate change (T4.5)
* Wildfires prediction due to climate change (T4.5)
* Eddies prediction (T4.5)
* Alpine Droughts Early Warning use-case (T4.6)
* Extreme rainfall, temperature and wind - weather extremes as a result of climate change (T4.7)
  + 1. Workflows

Details about the workflows for the different DT applications will be reported in D4.7.

* + 1. Thematic modules

The final set of thematic modules from T7.4 was described in D7.7 and includes the following:

1. **ML TC detection:**Python modules for tropical cyclones-related data analysis and events detection*.*
2. **ML4Fires***:* Python modules for wildfires-related data analysis and events prediction*.*
3. **eddiesML***:* Python modules for oceanic mesoscale eddies data analysis*.*
4. **xtclim***:* Python module forgeneric detection and characterization of climate extreme changes and impacts in the future climate projections*.*
5. **downscaleML***:* Python package for downscaling climate data.
6. **emergence.compound**: Library for detection of time or periods of emergence for compound events.
7. **Esgpull\_rucio***:* toolkit for gathering CMIP6 data from ESGF, through the esgpull tool, and uploading it to the RUCIO data lake.

Details about each module are presented in section 3.4.

## T7.5 Earth Observation Modelling and Processing

* + 1. Alpine Droughts Early Warning use-case description

The Alpine Drought Early Warning DT objective is to support users from local and regional water management agencies, civil protection and research institutes to run complex workflows and integrate heterogeneous data inputs to finally produce hydrological forecasts from seasonal climate forecasts. The forecasted variables are surface soil moisture and evapotranspiration.

Both DT developers and users interact with the DT functionalities through the openEO API, connecting openEO processes to generate a configurable process graph. Each openEO process wraps an application component of the workflow.

The DT developer can set up and run the physical-based hydrological model, using the *HydroMT* and *Wflow\_sbm* thematic modules. The outputs are used to train the hydrological emulator leveraging the core module *ItwinAI* and the *Hython\_sbm* thematic module, integrated in the *hython-itwinai* plugin. The emulator is then calibrated by using parameter learning, leveraging remote sensing surface soil moisture. The plugin also supports emulator model logging, hyper parameter optimization and uncertainty quantification. The final model is archived on the *Mlflow* server model registry and is accessible to the DT user.

The DT user can downscale the climate seasonal forecast using *downscaleML* thematic module. The trained *hython\_sbm* model weights are uploaded from the *Mlflow* model registry, then the downscaled seasonal forecasts are used to force the *Hython\_sbm* emulator and produce hydrological seasonal forecasts.

### High-level Architecture of the DT Implementation

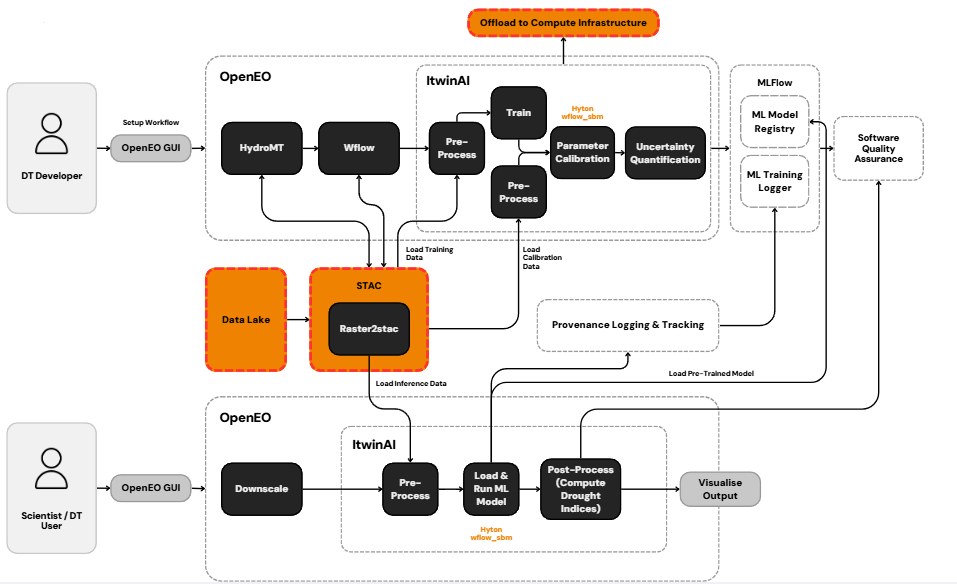


Figure High-level architecture of the Alpine Drought Early Warning DT

* + 1. Workflow description

A digital twin developer interfaces with openEO through one of its clients (Python, R, Web Editor). HydroMT and Wflow serve as model-building components that have been containerized for use in OSCAR. Initially, HydroMT is executed on the input data to create a configuration file for the Wflow model builder. The outcomes are registered as STAC collections and stored in the interTwin data lake utilizing the raster2stac component. These files are subsequently employed as input for the Wflow step. The Wflow model builder generates a NetCDF file, which is again registered in STAC and uploaded to the object store. Ultimately, the Wflow output is specified as input in the Itwinai configuration file, along with additional configurable parameters for the training phase. The resulting model is then uploaded and registered in MLflow.

A digital twin user connects to openEO through one of its clients, similar to the digital twin developer. The user specifies a spatial and temporal extent during the downscaling phase. The preprocessing phase occurs in openEO, with the data saved to STAC, allowing it to be loaded by the ML model. During the post-processing phase, drought indices are calculated, and the output can be visualized.

### Thematic Modules Integration

**openeo-processes-dask:**

This thematic module contains the Python implementation of the openEO processes, implemented using Xarray and Dask. The recent development allowed us to interact with the interTwin object storage, containing the dataset stored in different formats, like COG (Cloud Optimized Geotiff), netCDF and Zarr. Specifically, the entry point for the users will be the load\_stac process, which will query the interTwin STAC API (SpatioTemporal Asset Catalog) to get the required data. Specific work has been carried out to allow precise merge of data from different sources (satellite, climate).

**downscaleML :**

The downscaleML thematic module enables automated machine learning–based downscaling of seasonal climate forecasts (temperature, precipitation and surface solar radiation downwards), with a focus on climate extremes. It integrates with openeo-processes-dask, raster2stac, and STAC to form an end-to-end, reproducible pipeline. Both DT developers and users can apply it to the domain and target variable of their choice. In the Alpine Drought Early Warning use case, it is used at the inference stage to generate downscaled forecasts that feed into the early warning system.

**hython\_sbm:**

This thematic module consists of a deep learning LSTM surrogate, tailored to emulate any grid based hydrological or land surface physical-based model. The module also supports other deep learning architectures, such as Conv-LSTM and Transformer. Finally, the module provides functionality as well to run uncertainty estimation.

**raster2stac:**

This thematic module is crucial for mapping the data output operations of various thematic modules, as it facilitates the creation of valid STAC Collections from a wide array of file formats or in-memory data represented as Xarray objects. It adheres to STAC best practices to ensure optimal interoperability and is equipped to manage both geospatial and climate data. The outcome is a collection of JSON documents that include the necessary STAC documents for generating STAC Collections and STAC Items through HTTP requests at the intertwin STAC API. The resulting files are automatically saved to the interTwin object storage.

**OSCAR:**

OSCAR enables seamless serverless execution of containers in the cloud. It provides functionality to register user-defined containers and associated workflows, which can be triggered to initiate processing tasks. To facilitate integration between OSCAR and the openEO ecosystem, a dedicated openEO process called *run\_oscar* has been defined. The implementation of the run\_oscar process is currently in progress and is expected to be completed by the end of the project. The goal is to empower users to submit and execute their own containers directly through an openEO process graph. Thanks to openEO’s broad range of clients and libraries, this integration will support flexible and user-friendly interaction methods.

## T7.6 Hydrological model data processing

### Use-case Description

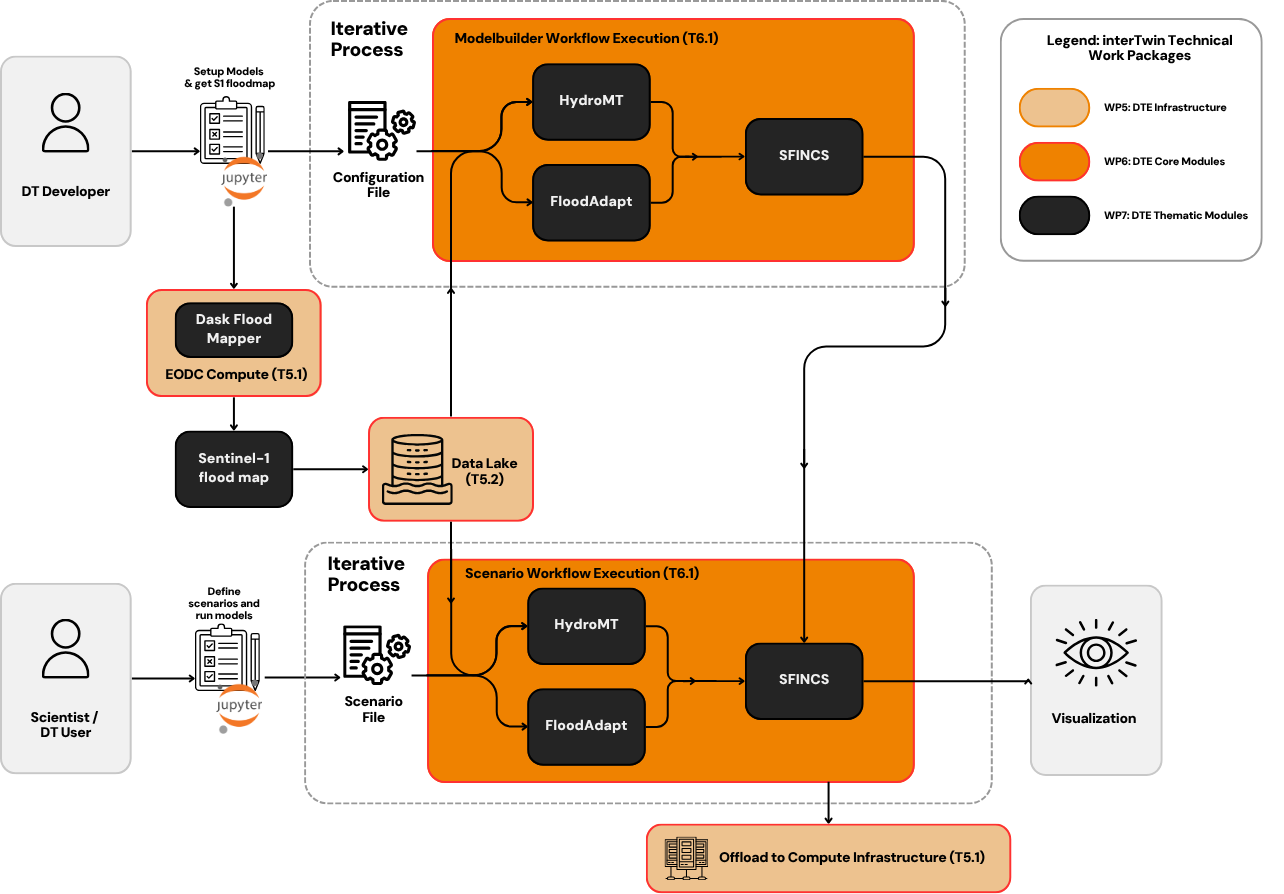
Hydrological model data processing in T7.6 supports the implementation of two Digital Twins (DTs) for flood risk management: post-flood analysis and climate impact assessment. Both use cases aim to help decision-makers understand, mitigate, and adapt to flood risks under present and future climate conditions.

The post-flood analysis DT enables near-real-time evaluation of observed flood events, integrating Earth Observation (EO) data and fast flood modelling to identify inundated areas and quantify their impacts. The climate impact DT focuses on forward-looking scenario simulations to assess long-term changes in flood exposure and vulnerability driven by changing metocean and hydrological conditions.

To support these use cases, we developed a suite of hydrological, flood and impact modelling modules that can be flexibly configured and deployed. The system architecture integrates both data-driven and physics-based models into orchestrated, near-automated workflows, enabling high-resolution flood simulation, impact quantification, and infrastructure vulnerability analysis. The outputs are designed to inform flood resilience planning, infrastructure protection, and climate adaptation policies.

### High-level Architecture of the DT Implementation

**Figure 9** shows the high-level architecture of the DT workflows. It highlights the iterative nature of model building and scenario execution, where users configure flood models through Jupyter notebooks. Configuration files are provided for the execution of modular workflows, consisting of hydrological, inundation, and impact models. The system interfaces with the interTwin Data Lake (T5.2) and leverages container orchestration on HPC and Kubernetes environments through OSCAR (WP6) for scalable execution. Interactive visualisation is provided in the Jupyter Notebooks to explore simulated flood maps, damage assessments, and network disruptions.



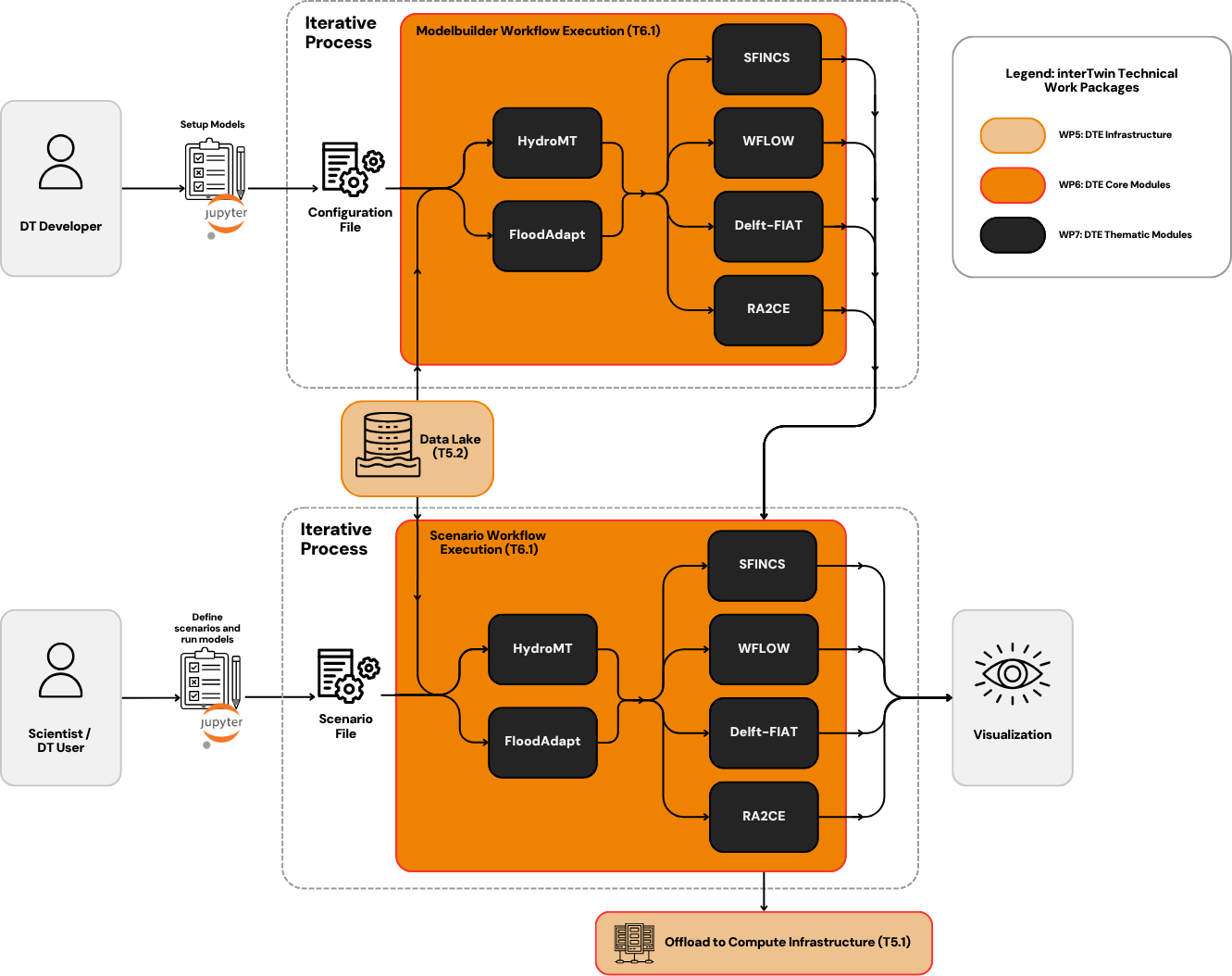


Figure High-level architecture of the flood risk DTs for post-flood analysis (top) and climate impact assessment (bottom)

### Modelbuilder Workflow

The Modelbuilder workflow is designed for DT developers to set up hydrological models for a specific region of interest. It is an iterative process where users configure and run models, calibrate results, and prepare baseline conditions. The workflow integrates the following key thematic modules:

* HydroMT: A Python-based model-building framework used to ingest and pre-process geospatial data for hydrological simulations.
* FloodAdapt: A scenario configuration and adaptation planning tool that prepares inputs for evaluating risk and resilience measures.
* WFLOW: A spatially distributed hydrological model used for simulating rainfall-runoff processes.
* SFINCS: A reduced-complexity, high-performance inundation model capable of modelling flood propagation across urban and rural landscapes.
* Delft-FIAT: A fast flood impact assessment tool that combines hazard, exposure, and vulnerability to estimate economic damage.
* RA2CE: A network risk assessment model for quantifying flood-induced disruptions to road infrastructure and accessibility.

This workflow is executed through Jupyter notebooks and relies on default configuration files, which can be customized per location or hazard type. Input and output data for model building is stored in the interTwin Data Lake.

### Scenario Workflow

The Scenario Workflow Execution allows DT users to run “what-if” simulations using predefined or user-generated climate or hazard scenarios. The same modules from the modelbuilder phase are reused here, driven by scenario files that define variables such as rainfall intensity, sea level rise, land-use changes, and flood defence breach conditions.

Execution is automated and offloaded to interTwin computing infrastructure (T5.1). Output maps and statistics include:

* Flood extent and depth
* Building-level economic damages
* Network accessibility losses

Currently, these outputs are being linked to visualisation tools in the Jupyter Notebooks, allowing interactive exploration of simulated scenarios for resilience planning and climate adaptation.

### Data Integration and Infrastructure

All input data are currently managed through the interTwin Data Lake (T5.2). The capability to similarly manage intermediate data is in progress, intending to test the ability to trigger OSCAR services directly from the Data Lake. Integration with the Data Lake is underway, with workflows already configured to store intermediary and final results in standard formats (e.g., NetCDF, GeoTIFF). EO-derived flood extent maps (e.g., from Sentinel-1) and climate model outputs (e.g., CMIP6) are preprocessed and used for model calibration and scenario generation.

The entire pipeline is deployable on interTwin’s federated computing infrastructure, supporting both containerized workloads and HPC-based processing. Workflow execution is managed using the OSCAR core service (WP6), ensuring consistency, reproducibility, and performance across deployments.

### Summary

T7.6 delivers a framework for flood risk Digital Twins. Through modular open-source tools, and integration with the interTwin ecosystem, it enables both reactive (post-event) and proactive (climate adaptation) use cases. The combination of hydrological, inundation, and impact models allows for analysis of flood hazards, economic losses, and infrastructure vulnerabilities. The final developments are focused on enhancing integration with the Data Lake and finalising the visualisation components, further increasing the DTs’ accessibility and policy relevance.

## T7.7 Fast particle detector simulation

### Use-case description

Task 7.7's goal is to develop the thematic module for the fast detector simulation using generative deep learning models.

Simulations in particle physics are needed to compare theoretical model predictions with experimental data. As the amount of experimental data increases, more simulated data needs to be produced. In experiments like those at the Large Hadron Collider, where large amounts of data are collected, optimizing computational resources for simulations is important. The most computationally expensive step is modelling particle interactions with detector materials, especially in calorimeter detectors.

Machine learning, in particular deep generative models, has been explored as an alternative to traditional simulation methods. This thematic module provides two generative models: Generative Adversarial Networks (GANs) [[**R15**](#_References)] and Normalising flows networks [[**R13**](#_References)], for fast simulation of calorimeter response on a particle passage.

### High-level Architecture of the DT Implementation

The DT consists of two main workflows, the training workflow, and the inference workflow, as illustrated in **Figure 10**. For the 3DGAN model, the whole pipeline is implemented as itwinai plugins. For CaloINN, the training workflow is integrated with itwinai, and the inference workflow implementation will be finalized before the end of the project. Below, the application functionalities and their specifications included in each workflow are described.

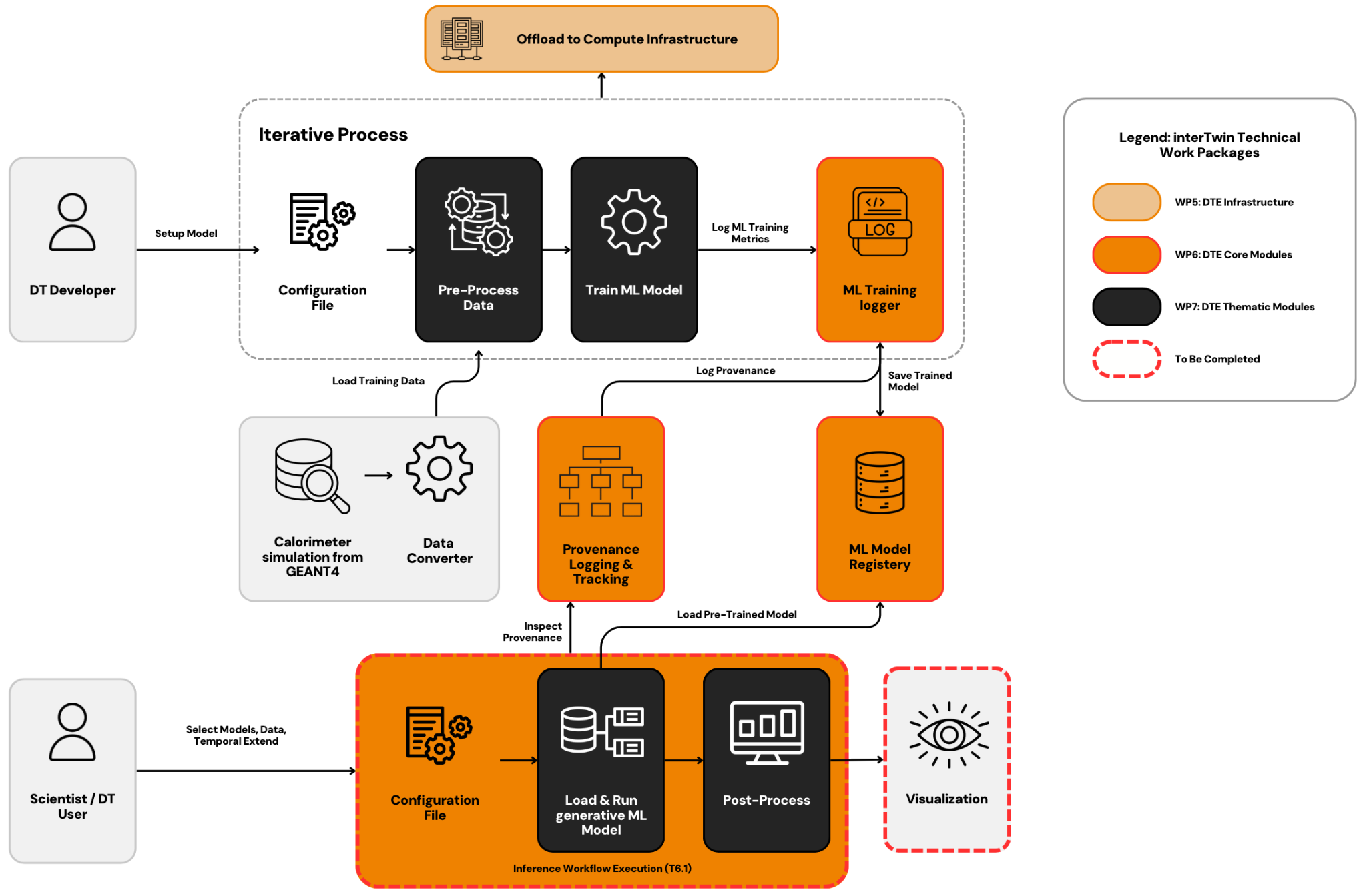


Figure System Context diagram of the Fast Detector Simulation DT

First, we use Monte Carlo based simulation framework, Geant4[[20]](#footnote-20) , to produce the training data. Two deep learning (3D Generative Adversarial Network - 3DGAN [[**R11**](#_References)] and Normalising flows network - CaloINN [[**R14**](#_References)]) components, developed for a specified particle detector set-up, are trained on Geant4 Monte Carlo simulations to generate detector response.

More specifically, T7.7 developed capabilities for T4.2 defined DT application that enable the specific DT operator to:

* pre-process the data, that a Geant4 application produced and simulate particles passing through a specific detector setup
* train a 3DGAN or CaloINN model on the pre-processed simulated data, with specified model input conditions (e.g. particle’s entrance angle, initial energy, and type)
* use the trained model during the inference step to replicate the detector’s response (fast simulation).

The Geant4 simulation toolkit performs particle physics simulations based on Monte Carlo (MC) techniques. The training workflow design includes the following functionalities, which will run on HPC systems. Geant4 simulates particle interactions, outside of the DT, producing data based on a detector-specific configuration. The produced data consists of the energy measured by the detector sensors, the properties of the initial particle, such as its type, energy, and trajectory angle with respect to the detector volume, and other metadata. The produced data, in ROOT format, are stored at data centres provided by project partners, with CERN currently serving as the primary storage site.

The data produced from the traditional Geant4 simulation in ROOT format requires conversion into the HDF5 format for further preprocessing before being input into any of the implemented models. The conversion is performed using Python scripts. The converted data is then stored at the data centres. Following the ROOT to HDF5 format conversion, the HDF5 data is further pre-processed and transformed into numpy arrays, a process incorporated within the model training scripts.

A 3DGAN and CaloINN are trained on the pre-processed data, conditioned on specific input describing the properties of the particles. The data is retrieved from the storage space where they reside. Hyperparameter optimization (HPO) is also employed to improve model performance. During the validation step, the model-generated data and the Geant4 simulated data distributions are both visualised for simpler comparison.

At the end, the training workflow stores the optimised models, selected based on validation results. The model registry where the 3DGAN models are stored is managed by Task 6.5. At the time of writing this report, the model registry is not used for the CaloINN model, but this functionality will be implemented.

During the inference workflow process, simulations of the specified detector’s response are produced. For 3DGAN the detector response is represented as 3D images consisting of the secondary particles’ positions (x, y, z coordinates) in the detector and their corresponding energy measurements. CaloINN represents calorimeter response as energy deposits in detector volume segmented into layers aligned with the direction of the incoming particle, with each layer further divided into radial and angular bins in polar coordinates (voxels). A cluster of detector signals is characterised by the incident energy of the incoming particle and the energy depositions in each voxel.

The inference step includes data visualisation as well for validating the efficacy and accuracy of the AI generated data. The data distribution comparisons are drawn between the 3DGAN-generated or CaloINN-generated data and real data (either derived from a traditional Geant4 simulation or data derived from accelerator test beams).

Finally, based on the results visualised, two possible workflows are proposed for simulation tuning. The model can either be re-inferred with different model input parameter values, provided these parameter values have been accounted for during model training. Alternatively, if a different value range of the conditional parameters is needed, the training workflow must be re-run from the beginning. These two possible workflows allow for greater flexibility and adaptability in tuning the detector's responses to various particle interactions.

## Summary of supported Digital Twins

The thematic modules described in the previous sections were developed to support several diverse Digital Twin applications from WP4. The following table provides an overview of these DT applications currently supported by the different thematic modules.

Table Summary of supported Digital Twins

|  |  |
| --- | --- |
| **Thematic modules classification** | **Digital Twins applications** |
| T7.1 Lattice QCD simulations and data management | [**Digital Twin for Lattice QCD simulation**](https://www.intertwin.eu/intertwin-use-case-lattice-qcd-simulation) |
| T7.2 Noise simulation for radio astronomy | [**Digital Twin to simulate 'noise' in Radio Astronomy**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-to-simulate-noise-in-radio-astronomy) |
| T7.3 GAN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for Gravitational Waves | [**Digital Twin to simulate 'noise' in the Virgo Gravitational Wave interferometer**](https://www.intertwin.eu/intertwin-use-case-virgo) |
| T7.4 Climate analytics and data processing | * [**Digital Twin for projecting wildfire danger due to climate change**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-for-projecting-wildfire-danger-due-to-climate-change) * [**Digital Twin for projecting the occurrence of tropical cyclones due to climate change**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-for-projecting-the-occurrence-of-tropical-cyclones-due-to-climate-change) * Eddies prediction on unstructured meshes * [**Digital Twin for Drought Early Warning in the Alps**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-for-drought-early-warning-in-the-alps) * [**Extreme rainfall, temperature and wind - weather extremes as a result of climate change**](https://www.intertwin.eu/intertwin-use-case-extreme-rainfall-temperature-and-wind-weather-extremes-as-a-result-of-climate-change) |
| T7.5 Earth observation modelling and processing | * [**Digital Twin for Drought Early Warning in the Alps**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-for-drought-early-warning-in-the-alps) * [**Digital Twin for post-flood analysis in coastal regions**](https://www.intertwin.eu/intertwin-use-case-flood-early-warning-in-coastal-and-inland-regions) |
| T7.6 Hydrological model data processing | * [**Digital Twin for Drought Early Warning in the Alps**](https://www.intertwin.eu/intertwin-use-case-a-digital-twin-for-drought-early-warning-in-the-alps) * [**Digital Twin for post-flood analysis in coastal regions**](https://www.intertwin.eu/intertwin-use-case-flood-early-warning-in-coastal-and-inland-regions) * [**Deploying FloodAdapt, a digital twin for flood impact modelling, anywhere on Earth**](https://www.intertwin.eu/intertwin-use-case-deploying-floodadapt-a-digital-twin-for-flood-impact-modelling-anywhere-on-earth) |
| T7.7 Fast simulation with GAN | [**Particle detector data-driven Digital Twin**](https://www.intertwin.eu/intertwin-use-case-a-particle-detector-data-driven-digital-twin-for-high-energy-physics)  [**for High-Energy Physics**](https://www.intertwin.eu/intertwin-use-case-a-particle-detector-data-driven-digital-twin-for-high-energy-physics) |

# Thematic modules and final development and integration activities

## T7.1 Lattice QCD simulations and data management

### openQxD

|  |  |
| --- | --- |
| Component name and logo | openQxD |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-openqxd**](https://www.intertwin.eu/article/thematic-module-openqxd) |
| Description | Flexible code that implements advanced lattice simulation techniques on HPC systems. |
| Value proposition | The base software component necessary to simulate quantum field theories with C\* boundary conditions. |
| Users of the Component | Expert users and Developers |
| User Documentation | [**https://gitlab.com/rcstar/openQxD/-/tree/master/doc**](https://gitlab.com/rcstar/openQxD/-/tree/master/doc?ref_type=heads) |
| Technical Documentation | [**https://gitlab.com/rcstar/openQxD/-/tree/master/doc**](https://gitlab.com/rcstar/openQxD/-/tree/master/doc?ref_type=heads) |
| Responsible | **RC\* Collaboration** |
| Licence | GPLv2 |
| Source code | [**https://gitlab.com/rcstar/openQxD**](https://gitlab.com/rcstar/openQxD) |
| Language | C |

### normflow

|  |  |
| --- | --- |
| Component name and logo | normflow |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-normflow**](https://www.intertwin.eu/article/thematic-module-normflow) |
| Description | For applying the method of normalising flows as a generative model for lattice simulations. |
| Value proposition | This package contains utilities for the implementation of normalising flows as a generative model using Pytorch. |
| Users of the Component | Expert users and Developers |
| User Documentation | [**https://github.com/interTwin-eu/normflow-plugin**](https://github.com/interTwin-eu/normflow-plugin) |
| Technical Documentation | [**https://github.com/interTwin-eu/normflow-plugin**](https://github.com/interTwin-eu/normflow-plugin) |
| Responsible | **ETHZ** |
| Licence | MIT |
| Source code | [**https://github.com/interTwin-eu/normflow-plugin**](https://github.com/interTwin-eu/normflow-plugin) |
| Language | Python |

### Functionalities developed since the last release

Building blocks have been added for gauge theories. It is now possible to assemble models appropriate for gauge theories and train them.

### Integrations with other DTE components

The component is based on itwinai (WP6). The central Trainer of the package is a subclass of itwinai TorchTrainer, which allows the use of a more standard way of saving and loading the models. It also allows the distribution of ML training over multiple workers (GPUs) and has been tested for multi-node configuration. itwinai enables distributed training on HPC by giving the option to the use case developers to switch between different distributed strategies, depending on which one is more suitable for the problem. During the development of this use case, Horovod, DeepSpeed, and torch DistributedDataParallel strategies were benchmarked. The module also has been implemented as an itwinai plugin, allowing the use-case to continue their developments independently. Moreover, thanks to the profiling functionality provided by itwinai, we were able to identify a major bottleneck in the QR decomposition and reduce training time by 70% on GPUs.

### Integrations with DT Applications

The components developed are part of the DT Application on LatticeQCD Simulation from T4.1

### Pilots and testing activities

### Both components have been piloted on the VEGA EuroHPC GPU partition resources dedicated to the project.

## T7.2 Noise simulation for radio astronomy

### PulsarDT

|  |  |
| --- | --- |
| Component name and logo | PulsarDT |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-pulsardt**](https://www.intertwin.eu/article/thematic-module-pulsardt) |
| Description | Physics-based DT, simulation of the propagation of pulsar signals from the source to antennas and generation of synthetic data – written in Python. |
| Value proposition | The physics-based DT, to be used to generate synthetic data to train the ML classifier. This particular component is written in Python as a model of how the different aspects of the physics-based DT can be implemented, while its counterpart, PulsarDT++ implements what has already been well-established in the C++ production version. |
| Users of the Component | Expert Users and Developers |
| User Documentation | [**https://gitlab.com/ml-ppa/pulsardt**](https://gitlab.com/ml-ppa/pulsardt) |
| Technical Documentation | [**https://gitlab.com/ml-ppa/pulsardt**](https://gitlab.com/ml-ppa/pulsardt) |
| Responsible | **ML-PPA collaboration** |
| Licence | GNU AGPLv3 |
| Source code | [**https://gitlab.com/ml-ppa/pulsardt**](https://gitlab.com/ml-ppa/pulsardt) |
| Language | Python |

### PulsarDT++

|  |  |
| --- | --- |
| Component name and logo | PulsarDT++ |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-pulsardt-2**](https://www.intertwin.eu/article/thematic-module-pulsardt-2) |
| Description | Physics-based DT, simulation of the propagation of pulsar signals from the source to antennas and generation of synthetic data – written in C++. |
| Value proposition | PulsarDT implemented in C++ in order to improve its speed and allow for parallelization, easily deployable in a singularity container. |
| Users of the Component | Expert Users and Developers |
| User Documentation | [**https://gitlab.com/ml-ppa/pulsardtpp**](https://gitlab.com/ml-ppa/pulsardtpp) |
| Technical Documentation | [**https://gitlab.com/ml-ppa/pulsardtpp**](https://gitlab.com/ml-ppa/pulsardtpp) |
| Responsible | **ML-PPA collaboration** |
| Licence | GNU AGPLv3 |
| Source code | [**https://gitlab.com/ml-ppa/pulsardtpp**](https://gitlab.com/ml-ppa/pulsardtpp) |
| Language | C++, Python |

### PulsarRFI\_Gen

|  |  |
| --- | --- |
| Component name and logo | PulsarRFI\_Gen |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-pulsardt-3**](https://www.intertwin.eu/article/thematic-module-pulsardt-3) |
| Description | Empirical DT, generating “timeframes”, 2D images (time-frequency) with various classes of pulsar and RFI signals. This DT creates various types of telescope signals by mimicking available real data rather than generating them from the physical first principles as PulsarDT does. |
| Value proposition | By using an alternative and fundamentally different method of DT creation this tool provides comparison for PulsarDT/DT++ and substitute training data for the ML classifier. |
| Users of the Component | Expert Users and Developers |
| User Documentation | [**https://gitlab.com/ml-ppa/pulsarrfi\_gen**](https://gitlab.com/ml-ppa/pulsarrfi_gen) |
| Technical Documentation | [**https://gitlab.com/ml-ppa/pulsarrfi\_gen**](https://gitlab.com/ml-ppa/pulsarrfi_gen) |
| Responsible | **ML-PPA collaboration** |
| Licence | GNU AGPLv3 |
| Source code | [**https://gitlab.com/ml-ppa/pulsarrfi\_gen**](https://gitlab.com/ml-ppa/pulsarrfi_gen) |
| Language | Python |

### PulsarRFI\_NN

|  |  |
| --- | --- |
| Component name and logo | PulsarRFI\_NN |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-pulsarrfi\_nn**](https://www.intertwin.eu/article/thematic-module-pulsarrfi_nn) |
| Description | The ML classifier. It is a CNN-based tool for the identification of various types of pulsar and RFI signals in the “timeframes”, 2D images (time-frequency). |
| Value proposition | The main tool of the framework, it plays a key role in the ML-PPA. |
| Users of the Component | Expert Users and Developers |
| User Documentation | [**https://gitlab.com/ml-ppa/pulsarrfi\_nn**](https://gitlab.com/ml-ppa/pulsarrfi_nn) |
| Technical Documentation | [**https://gitlab.com/ml-ppa/pulsarrfi\_nn**](https://gitlab.com/ml-ppa/pulsarrfi_nn) |
| Responsible | **ML-PPA collaboration** |
| Licence | GNU AGPLv3 |
| Source code | [**https://gitlab.com/ml-ppa/pulsarrfi\_nn**](https://gitlab.com/ml-ppa/pulsarrfi_nn) |
| Language | Python |

### Functionalities developed since the last release

* Spark Pattern Generator: A new module has been added to create custom spark patterns.
* Optimised Data Generation Pipeline**:** The data generation pipeline has been modularised and optimised using Ray, with added functionality to save outputs at various stages for enhanced data reproducibility.
* Advanced Visualisation Tool**:** An upgraded visualisation tool has been integrated to plot pulsar states using PyVista, providing more detailed and interactive representations.
* Automated Testing: Ensuring quality control for further iterations.
* Installable via pip: This tool can easily be installed through the package provided in the registry.

### Integrations with other DTE components

ML training is implemented with itwinai from WP6, and the modules are integrated with Data Lake using Teapot, which is used to access input data from the repositories at DZA.

### Integrations with DT Applications

The components developed are the main components of the DT Application on Radio Astronomy from T4.2

### Pilots and testing activities

The integration with itwinai of the components has been tested in the resources offered by Julich Supercomputer.

## T7.3 GenNN-based thematic modules to manage noise simulation, low-latency de-noising and veto generation for gravitational waves

### GlitchFlow

|  |  |
| --- | --- |
| Component name and logo | GlitchFlow |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-glitchflow**](https://www.intertwin.eu/article/thematic-module-glitchflow) |
| Description | Provides a set of Python modules to set up a GenNN architecture, train it and use it to clean glitches in GW data |
| Value proposition | Main component of the Virgo DT. It performs vetoing and denoising on GW interferometer data leveraging GenNN, specifically a Convolutional U-net with residual blocks and attention gates enhanced skipped connections, to map carefully selected auxiliary channels (sensitive to the noise but not to GW signals) into the main channel of the interferometer. The generated output is then subtracted from the interferometer main channel data. |
| Users of the Component | * GW detector characterization and data analysis experts * Developers |
| User Documentation | [**https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final\_Release/README.md**](https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final_Release/README.md) |
| Technical Documentation | [**https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final\_Release/README.md**](https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final_Release/README.md) |
| Responsible | **INFN** |
| Licence | MIT |
| Source code | [**https://github.com/interTwin-eu/DT-Virgo-dags/tree/main/Final\_Release/Glitchflow**](https://github.com/interTwin-eu/DT-Virgo-dags/tree/main/Final_Release/Glitchflow) |
| Language | Python |

### ANNALISA

|  |  |
| --- | --- |
| Component name and logo | ANNALISA |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-annalisa/**](https://www.intertwin.eu/article/thematic-module-annalisa/) |
| Description | Provides a set of Python modules for supporting processing and Channel selection for GW data |
| Value proposition | ANNALISA’s channel selection algorithm makes use of time-frequency domain representation of the data, namely the QTransform, to evaluate correlations among the main and auxiliary channels as a measure of temporally coincident spikes in the energetic content of the signals above a critical threshold.  The data preprocessing consists of data resampling, whitening, spectrogram generation, image cropping, and loading into a custom PyTorch dataloader. |
| Users of the Component | * GW detector characterization and data analysis experts * Developers |
| User Documentation | [**https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final\_Release/README.md**](https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final_Release/README.md) |
| Technical Documentation | [**https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final\_Release/README.md**](https://github.com/interTwin-eu/DT-Virgo-dags/blob/main/Final_Release/README.md) |
| Responsible | **INFN** |
| Licence | MIT |
| Source code | [**https://github.com/interTwin-eu/DT-Virgo-dags/tree/main/Final\_Release/Annalisa**](https://github.com/interTwin-eu/DT-Virgo-dags/tree/main/Final_Release/Annalisa) |
| Language | Python |

### Functionalities developed since the last release

The entire GlitchFlow pipeline has been migrated from Airflow DAGs to itwinaI plugins.

Metric and accuracy logging during training, along with model uploads, have been implemented on MLFlow via itwinaI. The inference subsystem has been integrated, and the logging of denoised data, veto, and denoising metadata on Tensorboard has been added.

While the Annalisa module has transitioned from a pip-installable Python package to an itwinai plugin. The Qtransform algorithm has undergone updates to address minor border effects and now offers the capability to return uninterpolated Q-tiles, energy or amplitude Q-tiles, and phase Q-tiles. The final version of the model was trained using amplitude and phase Q-tiles together, as they provide more informative and neural-network-friendly data compared to the previously used energy Q-tiles. A new whitening algorithm has been developed. This algorithm aims to mitigate edge artifacts that arise when whitening short time series, and it also improves the overall signal-to-noise ratio of glitches in both main and auxiliary channels.

A new custom data class named TFrame has been developed. Built upon PyTorch's Tensor, TFrame allows for the inclusion of metadata and eliminates the dependency on gwpy's data structure. The entire pipeline is now built on PyTorch, incorporating newly added custom resampling and pass-band filters.

### Integrations with other DTE components

The GlitchFlow module has been implemented as a itwinai plugin, which allows for offloading to computing infrastructure and distributed training, user friendly configuration of the pipeline and integration with metadata logging and weights uploading to model catalog on MLflow.

The user can set via .yaml configuration file:

* Steps and order (scan, preprocess, training, inference, visualization)
* Training parameters (loss function, learning rate, number of epochs, accuracy function, batch size, data normalization)
* NN architecture and weights form model registry
* Path for saving and loading weights, data, results

The Annalisa module has been also implemented as a itwinai plugin, which allows for user friendly full configuration of the pipeline via .yaml file, including:

* Preprocess parameters (sampling rate, bandpass filter parameters, whitening parameters, Qtransform parameters)
* Dataset path

### Integrations with DT Applications

Both GlitchFlow and the Annalisa modules are the main components of the Virgo DT (T4.4). While the former is responsible for vetoing and denoising on GW interferometer data, the latter is used for channel selection and data preprocessing in the Virgo DT.

### Pilots and testing activities

Both modules were developed and tested on the INFN-Turin computing infrastructure where the training of the GenNN was conducted on a Nvidia Grace Hopper GH200. The components have been also tested on the VEGA EuroHPC resources part of the project.

## T7.4 Climate analytics and data processing

### ML TC detection

|  |  |
| --- | --- |
| Component name and logo | **Thematic modules for tropical cyclones (TCs)** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-ml-tc-detection**](https://www.intertwin.eu/article/thematic-module-ml-tc-detection) |
| Description | Provides a set of Python modules for supporting processing and analysis of TC-related data and data-driven models |
| Value proposition | Address tropical cyclones analysis by providing the tools for gathering and pre-processing data, training different ML models, and post-processing the results. Furthermore, it provides functions for training ML models and running ensembles of multiple ML models. Different types of ML models are supported, in particular CNN, Transformers and GNN. Both deterministic and data-driven trackers are supported. |
| Users of the Component | * Developers of DTs * Expert scientists |
| User Documentation | [**https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/blob/main/README.md**](https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/blob/main/README.md) |
| Technical Documentation | [**https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/blob/main/README.md**](https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection/blob/main/README.md) |
| Responsible | **CMCC** and **UNITN** |
| Licence | GPLv3 |
| Source code | [**https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection**](https://github.com/CMCC-Foundation/ml-tropical-cyclones-detection) |
| Language | Python, PyTorch |

### ML4Fires

|  |  |
| --- | --- |
| Component name and logo | **ML4Fires** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-ml4fires**](https://www.intertwin.eu/article/thematic-module-ml4fires) |
| Description | Provides a set of Python modules for supporting the processing and analysis of wildfires-related data and data-driven models |
| Value proposition | Address wildfires analysis and prediction (e.g., producing burned areas maps) providing tools that allow users to pre-process data, choose ML model architecture, train the model, post-process and visualize the results. Furthermore, it integrates functionalities to track ML model metrics and provenance during the training phase. The set of python modules also provides tools to use the CMIP6 data to predict and analyze the burned area utilizing data-driven techniques. |
| Users of the Component | * Developers of DTs * Expert scientists |
| User Documentation | [**https://github.com/CMCC-Foundation/ML4Fires/blob/main/README.md**](https://github.com/CMCC-Foundation/ML4Fires/blob/main/README.md) |
| Technical Documentation | [**https://github.com/CMCC-Foundation/ML4Fires/blob/main/README.md**](https://github.com/CMCC-Foundation/ML4Fires/blob/main/README.md) |
| Responsible | **CMCC** |
| Licence | Apache v2.0 |
| Source code | [**https://github.com/CMCC-Foundation/ML4Fires**](https://github.com/CMCC-Foundation/ML4Fires) |
| Language | Python, PyTorch |

### eddiesML

|  |  |
| --- | --- |
| Component name and logo | **eddiesML** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-eddiesgnn**](https://www.intertwin.eu/article/thematic-module-eddiesgnn) |
| Description | Provides a set of Python modules for supporting processing and analysis of eddy-related data |
| Value proposition | Address oceanic mesoscale eddies analysis by providing the tools for pre-processing of FESOM2 data and training CNN models. It can be considered an example of “exploitation” as it applies interTwin technologies to an extra (external) application, showing the potential of such integration. |
| Users of the Component | * Developers of DTs * Expert scientists |
| User Documentation | [**https://github.com/LegoCreation/CNN\_eddy\_detection/blob/unitn\_work/readme.md**](https://github.com/LegoCreation/CNN_eddy_detection/blob/unitn_work/readme.md) |
| Technical Documentation | [**https://github.com/LegoCreation/CNN\_eddy\_detection/blob/unitn\_work/readme.md**](https://github.com/LegoCreation/CNN_eddy_detection/blob/unitn_work/readme.md) |
| Responsible | **UNITN** |
| Licence | GPLv3 |
| Source code | [**https://github.com/LegoCreation/CNN\_eddy\_detection/tree/unitn\_work**](https://github.com/LegoCreation/CNN_eddy_detection/tree/unitn_work) |
| Language | Python, Tensorflow |

### xtclim

|  |  |
| --- | --- |
| Component name and logo | **A logo of a planet with lightning and flames**  **xtclim** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-xtclim**](https://www.intertwin.eu/article/thematic-module-xtclim) |
| Description | xtclim is a Python package implementing an unsupervised Deep Learning method, a CVAE that can characterise generic climate extreme events |
| Value proposition | Base methods and functions to provide the extraction of generic characteristics of climate extremes, using an AI anomaly detection method. It enables users to explore the impacts of extreme events on specific users’ applications in the context of selected climate simulations. |
| Users of the Component | * Developers of DTs * Expert scientists |
| User Documentation | [**https://github.com/interTwin-eu/xtclim/notebooks/presentation\_notebook.ipynb**](https://github.com/interTwin-eu/xtclim/notebooks/presentation_notebook.ipynb)  [**https://github.com/interTwin-eu/xtclim/README.md**](https://github.com/interTwin-eu/xtclim/README.md) |
| Technical Documentation | [**https://github.com/interTwin-eu/xtclim/README.md**](https://github.com/interTwin-eu/xtclim/README.md) |
| Responsible | **CERFACS** |
| Licence | Apache 2 |
| Source code | [**https://github.com/interTwin-eu/xtclim**](https://github.com/interTwin-eu/xtclim) |
| Language | Python, PyTorch |

### downscaleML

|  |  |
| --- | --- |
| Component name and logo | **downscaleML: Downscaling Climate Data** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-downscaleml**](https://www.intertwin.eu/article/thematic-module-downscaleml) |
| Description | downScaleML is an open-source Python package, designed to streamline the process of climate data downscaling using machine learning techniques. It offers an automated workflow tailored for downscaling ECMWF’s ERA5 and SEAS5 seasonal forecast climate variables, specifically temperature, precipitation and downward surface solar radiation, with a particular emphasis on addressing climate extremes. |
| Value proposition | It eases forecast data preprocessing, statistical downscaling, through a selection of machine learning techniques, and result validation. It provides a flexible module for any modelling scheme requiring tailored climate inputs, and enables scalability and applicability to other domains, resolutions, and datasets. |
| Users of the Component | * Developers of DTs * Expert scientists |
| User Documentation | [**https://github.com/interTwin-eu/downScaleML#readme**](https://github.com/interTwin-eu/downScaleML#readme) |
| Technical Documentation | [**https://github.com/interTwin-eu/downScaleML#readme**](https://github.com/interTwin-eu/downScaleML#readme) |
| Responsible | **EURAC** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/interTwin-eu/downScaleML**](https://github.com/interTwin-eu/downScaleML) |
| Language | Python, PyTorch |

### emergence.compound

|  |  |
| --- | --- |
| Component name and logo | **emergence.compound** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-compevpoetoe**](https://www.intertwin.eu/article/thematic-module-compevpoetoe) |
| Description | Provides a set of R functions for determining if periods of emergence (PoE) and/or time of emergence (ToE) of compound events probabilities have emerged in data.  Publication where the module has been involved: https://doi.org/10.5194/egusphere-2025-461 |
| Value proposition | This module allows to statistically model if and how compound events have significantly evolved through time, based on reanalysis or simulated data. The definition of the compound (i.e., involved variables) is made by the user. |
| Users of the Component | * Expert scientists |
| User Documentation | [**https://github.com/josephine400/emergence.compound/blob/main/README.md**](https://github.com/josephine400/emergence.compound/blob/main/README.md) |
| Technical Documentation | [**https://github.com/josephine400/emergence.compound/blob/main/README.md**](https://github.com/josephine400/emergence.compound/blob/main/README.md) |
| Responsible | **CNRS** |
| Licence | CeCill-C |
| Source code | [**https://github.com/josephine400/emergence.compound**](https://github.com/josephine400/emergence.compound) |
| Language | R |

### Esgpull\_rucio

|  |  |
| --- | --- |
| Component name and logo | **esgpull\_rucio** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-esgpull\_rucio**](https://www.intertwin.eu/article/thematic-module-esgpull_rucio) |
| Description | A toolkit designed to interface with the esgpull (ESGF download tool) database to maintain an up-to-date Intake catalog and manage RUCIO datasets. It automatically detects completed file downloads from the database, registers each completed dataset, and uses the RUCIO Python API to create any missing data identifiers (DIDs). The tool also uploads the corresponding files and attaches them to the appropriate RUCIO datasets. |
| Value proposition | The tool seamlessly integrates with the ESGPull DB API, streamlining the workflow for downloading, cataloging, and appending ESGF and CMIP data to RUCIO |
| Users of the Component | * Developers of DTs |
| User Documentation | [**Github**](https://github.com/AtefBN/esgpullUtilties/blob/main/README.md) |
| Technical Documentation | [**Github**](https://github.com/AtefBN/esgpullUtilties/blob/main/README.md) |
| Responsible | **CNRS** |
| Licence | CeCill-C |
| Source code | [**https://github.com/AtefBN/esgpullUtilties**](https://github.com/AtefBN/esgpullUtilties) |
| Language | Python/SQL |

### Functionalities developed since the last release

The ML TC Detection module has been extended since the previous release as follows:

* enhancing the TC tracking inference pipeline by refining the integration between detection outputs and the tracking algorithm, improving temporal continuity and trajectory reliability;
* additional validation metrics (e.g., track duration and spatial distribution) have been integrated into the module to provide a more robust evaluation of the results and related visualization tools have been extended to support comparative and exploratory analysis;
* dedicated data preprocessing functions have been implemented to facilitate the preparation of weather and climate data (i.e., CMIP6) inputs for the detection and tracking pipelines, ensuring compatibility and scalability across datasets.

The current version of the ML4Fires thematic module provides an updated and improved documentation to describe the different capabilities of the module. In terms of developments and extensions, this version includes:

* Logging of training and validation metric was enabled and allows for logging of metric from any python library, such as sci-kit or torchmetric, as well as from any python module locally defined.
* Extended the inference pipelines for supporting results processing and visualisation;
* Improved computation of metric during the training, validation and test phase of the ML model.
* Finalization of pipeline to read, collect and aggregate CMIP6 datasets to prepare them for inference. To this end, several tools have been provided for the CMIP6 inference notebook to process and visualize the inference and the CMIP6 dataset.

The eddiesML module has been finalized. In this version, the application from the Alfred Wegener Institut (AWI) has been finalized, tested and validated on VEGA, by using data from FESOM2 model simulations.

Future plans (beyond the project lifetime) in collaboration with AWI relates to evaluating the potential benefits in migrating from CNN to GNN considering the unstructured nature of the FESOM2 mesh. The eddiesML has represented a successful attempt to exploit interTwin technologies within external DTs, thus showing the benefits coming from interTwin capabilities developed during the project. As an example, provenance tracking has been integrated via yProv4ML API, by simply adding a few extra lines of code, to generate a complete provenance graph documenting the eddiesML training process.

The xtclim module has been further developed and validated during the last period. The current version of xtclim thematic module is publicly available as open-source code on github. Documentation has been improved: user documentation using a Jupyter notebook, and technical documentation in the form of a technical report. The updated version includes the following:

* Update: Complete refactoring of the code. Cleanup of repositories and removal of old versions in separate repositories.
* Update: Improve model output and behavior: removal of topography as an input parameter of the CVAE method.
* Update: Optimization of the hyper-parameters.
* Update: User documentation much improved through the use of a complete Jupyter Notebook. More end-user analysis products.
* Update: sample NetCDF (CMIP6) dataset provided as an example.
* New: technical documentation (technical report).
* New: Unit tests (github CI) and SQAAS Gold Medal.

The current version of the downscaleML module introduces several important advancements aimed at improving reproducibility, integration, and automation. Key developments include:

* Transitioned from a standalone setup to an integrated workflow using OpenEO Dask-based preprocessing with Dockerized deployment of the downscaleML model. The codebase was refactored to support this architecture.
* Developed a Jupyter Notebook pipeline to execute the full OpenEO processing chain and trigger the downscaleML inference via Docker, enabling seamless end-to-end execution.
* Improved support for STAC-formatted datasets across the training, validation, and testing phases, with tighter coupling to the ML pipeline.
* Unit tests implemented and updated for all preprocessing components using OpenEO Dask-based workflows, ensuring reliability and code robustness.
* Integrated with the raster2stac package to direct OpenEO process outputs to a STAC catalog and store results in the interTwin S3 bucket, facilitating standard-compliant data sharing and discoverability.

### Integrations with other DTE components

The *ML TC Detection* and *ML4Fires* module both integrate the *itwinai* (WP6) tool for logging the ML model skills during training, the selected hyper-parameters of an experiment and the resulting model on MLflow. Through this integration also the provenance of the training process is tracked via *yProv4ML* (WP6). In order to provide seamless access to the data needed by the DT applications, both modules can interact with *RUCIO* (WP5) for discovering the needed data. Moreover, workflows based on *Ophidia* (WP6) are integrated for supporting pre-processing or post-processing pipelines on CMIP6 climate projection data (HighResMIP or ScenarioMIP).

In both cases, Docker images, based on the itwinai one, integrating the different components and tools (e.g., PyTorch, Ophidia) needed for running the different stages of the pipelines are available (from WP6). Such images can be converted into Singularity images and transparently deployed on the infrastructure using *interLink* (WP5).

CMIP6 data can be gathered from the ESGF nodes and uploaded on the *RUCIO* data lake (WP5) using the *esgpull\_rucio* thematic tool.

The *xtclim* module is integrated as a plug-in in the *itwinai* (WP6). Like the ML TC Detection and ML4Fires modules, *xtclim* has been extended to interact with RUCIO (WP5) for discovering the needed data.

The *downscaleML* module integrates with the OpenEO Dask-based processes for preprocessing, utilizes *raster-to-stac* to publish STAC items to the interTwin STAC catalogue, and connects to interTwin storage for data storage and retrieval.

The latest version of the module *eddiesML* includes the successful integration of the provenance tracking via *yProv4ML* (WP6).

### Integrations with DT Applications

The *ML TC Detection* module is used directly to support the DT application on detection and tracking of TCs, while the *ML4Fires* module is used in the DT application for the prediction of wildfires due to climate change.

The *xtclim* module is used as a DT application for the detection and characterization of climate extremes.

The *downScaleML* module supports the drought early warning Digital Twin by downscaling seasonal forecasts to high-resolution inputs for hydrological drought prediction in Alpine river basins.

The *eddiesML* module supports the eddies DT for the processing and analysis of eddy-related data.

### Pilots and testing activities

The *ML TC Detection*, *ML4Fires* and *eddiesML* moduleshave been deployed and tested on the interTwin testbed running at Vega (WP5). The *xtclim* module has also been tested on the same testbed.

The emergence.compuond module has been tested for identification of times and periods of emergence of compounding “hot and dry” events over the European domain (Schmutz et al, in revision) [[**R23**](#_References)].

## T7.5 Earth Observation Modelling and Processing

### openeo-processes-dask

|  |  |
| --- | --- |
| Component name and logo | **openeo-processes-dask** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-openeo-processes-dask**](https://www.intertwin.eu/article/thematic-module-openeo-processes-dask) |
| Description | Python implementation of openEO processes. |
| Value proposition | Base component necessary to run openEO process graphs. All the processes are implemented using Dask **[**[**R7**](#_References)**]**, making them easily scalable and parallelizable. |
| Users of the Component | * Expert users * Flood and drought modellers |
| User Documentation | [**https://open-eo.github.io/openeo-python-client/cookbook/localprocessing.html**](https://open-eo.github.io/openeo-python-client/cookbook/localprocessing.html) |
| Technical Documentation | [**https://github.com/Open-EO/openeo-processes-dask**](https://github.com/Open-EO/openeo-processes-dask) |
| Responsible | **EODC** and **EURAC** |
| Licence | Apache 2.0 |
| Source code | [**https://github.com/Open-EO/openeo-processes-dask**](https://github.com/Open-EO/openeo-processes-dask) |
| Language | Python |

### openeo-pg-parser-networkx

|  |  |
| --- | --- |
| Component name and logo | **openeo-pg-parser-networkx** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-openeo-pg-parser-networkx/**](https://www.intertwin.eu/article/thematic-module-openeo-pg-parser-networkx/) |
| Description | Parse openEO process graphs from JSON to traversable Python objects. |
| Value proposition | Base component necessary to parse openEO process graphs, before calling openeo-processes-dask. |
| Users of the Component | * Expert users |
| User Documentation | [**https://github.com/Open-EO/openeo-pg-parser-networkx/blob/main/README.md**](https://github.com/Open-EO/openeo-pg-parser-networkx/blob/main/README.md) |
| Technical Documentation | [**https://github.com/Open-EO/openeo-pg-parser-networkx/blob/main/README.md**](https://github.com/Open-EO/openeo-pg-parser-networkx/blob/main/README.md) |
| Responsible | **EODC** and **EURAC** |
| Licence | Apache 2.0 |
| Source code | [**https://github.com/Open-EO/openeo-pg-parser-networkx**](https://github.com/Open-EO/openeo-pg-parser-networkx) |
| Language | Python |

### raster-to-stac

|  |  |
| --- | --- |
| Component name and logo | **raster-to-stac** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-raster-to-stac/**](https://www.intertwin.eu/article/thematic-module-raster-to-stac/) |
| Description | Create STAC metadata for raster datasets. |
| Value proposition | Makes a resulting dataset easily accessible, interoperable, and shareable. |
| Users of the Component | * Expert users |
| User Documentation | [**https://raster2stac.readthedocs.io/en/latest/**](https://raster2stac.readthedocs.io/en/latest/) |
| Technical Documentation | [**https://raster2stac.readthedocs.io/en/latest/**](https://raster2stac.readthedocs.io/en/latest/) |
| Responsible | **EURAC** |
| Licence | MIT |
| Source code | [**https://gitlab.inf.unibz.it/earth\_observation\_public/raster-to-stac/**](https://gitlab.inf.unibz.it/earth_observation_public/raster-to-stac/) |
| Language | Python |

### dask-flood-mapper

|  |  |
| --- | --- |
| Component name and logo | **Dask Flood Mapper** |
| Description | dask-flood-mapper is an open-source Python package that uses Sentinel-1 radar images to map floods, replicating the TU Wien Bayesian-based flood mapping algorithm. It employs [**dask**](https://www.dask.org/) for scalable processing and accesses data via [**STAC**](https://stacspec.org/en) with [**odc-stac**](https://odc-stac.readthedocs.io/en/latest/). The algorithm depends on three pre-processed input datasets—Sentinel-1 SIG0 Backscatter, harmonic parameters (HPAR), and Mean Projected Local Incidence Angle (PLIA)—stored and accessible via STAC at the Earth Observation Data Centre for Water Resources Monitoring (EODC).  The package provides functionality to dynamically calculate HPAR from SIG0 if the former is not available, enabling the use of different STAC catalogues in future implementations.  dask-flood-mapper allows remote processing of the data at the EODC with the aid of a [**Dask Gateway**](https://docs.eodc.eu/services/dask.html), avoiding large file transfers to the user’s workstation.  The output data consists of a Bayesian decision of flood presence or probability per pixel. |
| Value proposition | The STAC and Dask-based solution for flood mapping allows cloud computing close to the data thereby freeing the user of the burden of downloading Sentinel-1 radar images while using the scalable resources of the host for processing. In combination with the intuitive API, this solution is an independent and reusable module that can be easily integrated into existing workflows. Furthermore, the open-source Python package allows expert users to adapt the flood mapping workflow to their own insights and demands. This sets it apart from the current [**Global Flood Monitoring implementation**](https://dev.globalfloods.eu/), which provides only statically produced flood maps. |
| Users of the Component | Expert users, flood risk specialists |
| User Documentation | [**https://intertwin-eu.github.io/dask-flood-mapper/README.html**](https://intertwin-eu.github.io/dask-flood-mapper/README.html) |
| Technical Documentation | [**https://github.com/interTwin-eu/dask-flood-mapper/blob/main/CONTRIBUTING.md**](https://github.com/interTwin-eu/dask-flood-mapper/blob/main/CONTRIBUTING.md) |
| Responsible | **TU Wien** |
| Licence | MIT |
| Source code | [**https://github.com/interTwin-eu/dask-flood-mapper**](https://github.com/interTwin-eu/dask-flood-mapper) |
| Language | Python |

### Functionalities developed since the last release

*dask-flood-mapper* is an evolution of the *openeo-flood-mapper* workflow, described in D7.5 [[**R21**](#_References)], which provides Dask compatibility and an interface that easily integrates into the Jupyter-Notebook-based workflows used in FloodAdapt (see [**Section 3.6.1**](#_FloodAdapt)). The software has been analysed for software standards by the Software Quality Assurance as a Service and has received the Gold Badge.

### Integrations with other DTE components

The *openeo-processes-dask* and *openeo-pg-parser-networkx* components are integrated into the openEO framework from WP6 as additional components. The *raster-to-stac* component is integrated with storage resources compatible with S3 interfaces from WP5. While the *dask-flood-mapper* integrates the openEO framework and in particular the *openeo-processes-dask* component

### Integrations with DT Applications

*dask-flood-mapper* has been integrated into the post-flood analysis and flood impact monitoring use cases, informing the process-based models developed in T4.7. The *openeo-processes-dask, openeo-pg-parser-networkx* and *raster-to-stac* components are being used in the DT Application from T4.6

### Pilots and testing activities

dask-flood-mapper and its integration into DT Applications were tested in a pilot use-case analyzing flood impacts of the storm surge of Storm Babet (October 2023) on the Darss Peninsula in Germany.

The *openeo-processes-dask* has been deployed in the GRNET testing openEO cluster.

## T7.6 Hydrological model data processing

### FloodAdapt

|  |  |
| --- | --- |
| Component name and logo | **FloodAdapt** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-floodadapt/**](https://www.intertwin.eu/article/thematic-module-floodadapt/) |
| Description | A thematic module which can be used to assess the benefits and costs of Flood Resilience measures in a community. It uses SFINCS, WFLOW, FIAT-Objects, and RA2CE in the background. |
| Value proposition | FloodAdapt is a decision-support tool and API that seeks to advance and accelerate flooding-related adaptation planning. It brings rapid, physics-based compound flood modelling and detailed impact modelling into an easy-to-use system, allowing non-expert end-users to evaluate a wide variety of compound events, future conditions, and adaptation options in minutes. FloodAdapt serves as a connector between scientific advances and practitioner needs, improving and increasing the uptake and impact of adaptation research and development. |
| Users of the Component | * Non-expert end-users * Decision makers * Planners |
| User Documentation | [**https://www.deltares.nl/en/software-and-data/products/floodadapt**](https://www.deltares.nl/en/software-and-data/products/floodadapt) |
| Technical Documentation | [**https://github.com/Deltares/FloodAdapt#readme**](https://github.com/Deltares/FloodAdapt#readme) |
| Responsible | **Deltares** |
| Licence | MIT |
| Source code | [**https://github.com/Deltares/FloodAdapt**](https://github.com/Deltares/FloodAdapt) |
| Language | Python |

### HydroMT-SFINCS

|  |  |
| --- | --- |
| Component name and logo | **HydroMT-SFINCS** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-hydromt-sfincs/**](https://www.intertwin.eu/article/thematic-module-hydromt-sfincs/) |
| Description | HydroMT (Hydro Model Tools) is an open-source Python package that facilitates the process of building and analysing spatial geoscientific models with a focus on water system models. It does so by automating the workflow to go from raw data to a complete model instance which is ready to run and to analyse model results once the simulation has finished. This plugin provides an implementation of the model API for the SFINCS model. |
| Value proposition | Easily build and update the SFINCS model with a single line of code. |
| Users of the Component | * Expert users * Flood modellers |
| User Documentation | [**https://deltares.github.io/hydromt\_sfincs/latest/index.html**](https://deltares.github.io/hydromt_sfincs/latest/index.html) |
| Technical Documentation | [**https://deltares.github.io/hydromt\_sfincs/latest/getting\_started/intro**](https://deltares.github.io/hydromt_sfincs/latest/getting_started/intro) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/hydromt\_sfincs**](https://github.com/Deltares/hydromt_sfincs) |
| Language | Python |

### HydroMT-FIAT

|  |  |
| --- | --- |
| Component name and logo | **HydroMT-FIAT** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-hydromt-fiat/**](https://www.intertwin.eu/article/thematic-module-hydromt-fiat/) |
| Description | HydroMT is an open-source Python package, developed by Deltares, to build and analyze hydro models. It provides a generic model API with attributes to access the model schematization, (dynamic) forcing data, results, and states. This plugin provides an implementation for the Delft-FIAT model. |
| Value proposition | With the HydroMT-FIAT plugin, users can easily benefit from the rich set of tools of the HydroMT package to build and update Delft-FIAT models from available global and local data.  This plugin assists the FIAT modeller in:   * quickly setting up a Delft-FIAT model based on existing hazard maps, global and user-input exposure layers, and a global database of vulnerability curves; * adjusting and updating components of a FIAT model and their associated parameters in a consistent way, e.g., to test measures that affect the exposure or vulnerability of a FIAT model or to improve an existing FIAT model with better quality data; * building FIAT models in a reproducible and consistent way. |
| Users of the Component | * Expert users * Flood risk specialists |
| User Documentation | [**https://deltares.github.io/hydromt\_fiat/latest/index.html**](https://deltares.github.io/hydromt_fiat/latest/index.html) |
| Technical Documentation | [**https://deltares.github.io/hydromt\_fiat/latest/index.html**](https://deltares.github.io/hydromt_fiat/latest/index.html) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/hydromt\_fiat/tree/main**](https://github.com/Deltares/hydromt_fiat/tree/main) |
| Language | Python |

### SFINCS

|  |  |
| --- | --- |
| Component name and logo | **SFINCS** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-sfincs/**](https://www.intertwin.eu/article/thematic-module-sfincs/) |
| Description | SFINCS is a new fast numerical model to simulate 2D compound flooding dynamically for large scale coastal systems, within a fraction of the time required by the Delft3D-1D2D models. |
| Value proposition | Compound flooding during extreme events can result in tremendous amounts of property damage and loss of life. Early warning systems and multi-hazard risk analysis can reduce these impacts. However, traditional approaches either do not involve relevant physics or are too computationally expensive to do so for large stretches of coastline. The SFINCS model (Super-Fast INundation of CoastS) is a new reduced-complexity engine recently developed by Deltares, that is capable of simulating compound flooding including a high computational efficiency balanced with good accuracy. |
| Users of the Component | * Expert users * Flood modellers |
| User Documentation | [**https://www.deltares.nl/en/software-and-data/products/SFINCS**](https://www.deltares.nl/en/software-and-data/products/SFINCS) |
| Technical Documentation | [**https://sfincs.readthedocs.io/en/latest/**](https://sfincs.readthedocs.io/en/latest/) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/SFINCS**](https://github.com/Deltares/SFINCS) |
| Language | Fortran |

### Delft-FIAT

|  |  |
| --- | --- |
| Component name and logo | **Delft-FIAT** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-delft-fiat/**](https://www.intertwin.eu/article/thematic-module-delft-fiat/) |
| Description | Delft-FIAT is a fast, free, Python-based tool developed and continuously improved by Deltares to rapidly assess direct economic impacts on buildings, utilities, and roads for user-input flood maps. |
| Value proposition | Fast impact modelling removes traditional bottlenecks in climate adaptation planning, making it possible to (1) understand the effectiveness of adaptation options and (2) quantify changes in damage and risk as climate and socio-economic conditions change.  **Fast and automated**  Delft-FIAT is fast and can be automated. This makes it possible to evaluate future risks caused by changing drivers like growing populations and economies. It also makes it possible to evaluate the effectiveness of interventions by assessing flood damages - now and under changing conditions (and combinations of) interventions, like home elevations, buy-outs, or floodproofing.  **Flexible**  Delft-FIAT has a flexible architecture and is data-agnostic. Exposure data can easily be modified, and hazard data - the flood maps - can come from any source.  For example, a user may want to try out different depth-damage functions or include a different class of damage than the traditional structure and content damages.  Furthermore, any damage type that can be described with a depth-damage function can be analysed in Delft-FIAT.  **Customisable**  Delft-FIAT is also customisable. It can be connected to a tailored user-interface to make a custom damage modelling tool for less-technical users. |
| Users of the Component | * Expert users * Flood risk specialists |
| User Documentation | [**https://www.deltares.nl/en/software-and-data/products/delft-fiat-flood-impact-assessment-tool**](https://www.deltares.nl/en/software-and-data/products/delft-fiat-flood-impact-assessment-tool) |
| Technical Documentation | [**https://github.com/Deltares/Delft-FIAT#readme**](https://github.com/Deltares/Delft-FIAT#readme) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/Delft-FIAT**](https://github.com/Deltares/Delft-FIAT) |
| Language | Python |

### WFLOW.jl

|  |  |
| --- | --- |
| Component name and logo | **WFLOW** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-wflow**](https://www.intertwin.eu/article/thematic-module-wflow) |
| Description | WFLOW is a free, open-source, hydrological modelling tool developed by Deltares to simulate the complete terrestrial water cycle. It allows users to model key processes such as precipitation, interception, snow accumulation and melt, evapotranspiration, soil moisture, surface runoff, groundwater recharge, and water demand and allocation. Based on gridded topography, land use, soil, and climate data, WFLOW calculates all hydrological fluxes at each model grid cell over time.  WFLOW is successfully applied worldwide for assessing flood hazards, droughts, climate change impacts, and land-use changes. Its framework-based design supports multiple model concepts and promotes flexibility and scalability, making it suitable for a wide range of hydrological applications, particularly in data-scarce environments. |
| Value proposition | WFLOW is a fast, flexible, and open-source hydrological modelling tool that empowers users to simulate the full water cycle—now and under future climate and land use scenarios. Designed for integration and scalability, it supports water availability assessments, flood and drought analysis, and climate adaptation planning in any data environment.  Built in Julia for performance, WFLOW is highly customisable, easy to couple with other models, and fully transparent. Its open architecture encourages collaboration and continuous development. Originally Python-based, WFLOW has evolved to meet the demands of modern, distributed hydrological modelling. |
| Users of the Component | * Expert users * Hydrologists |
| User Documentation | [**https://deltares.github.io/Wflow.jl/stable/**](https://deltares.github.io/Wflow.jl/stable/) |
| Technical Documentation | [**https://deltares.github.io/Wflow.jl/stable/**](https://deltares.github.io/Wflow.jl/stable/) |
| Responsible | **Deltares** |
| Licence | MIT |
| Source code | [**https://github.com/Deltares/Wflow.jl**](https://github.com/Deltares/Wflow.jl) |
| Language | Julia |

### HydroMT-WFLOW

|  |  |
| --- | --- |
| Component name and logo | **HydroMT-WFLOW** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-hydromt-wflow**](https://www.intertwin.eu/article/thematic-module-hydromt-wflow) |
| Description | [**HydroMT**](https://deltares.github.io/hydromt/latest/) (Hydro Model Tools) is an open-source Python package that facilitates the process of building and analyzing spatial geoscientific models with a focus on water system models. It does so by automating the workflow to go from raw data to a complete model instance which is ready to run and to analyze model results once the simulation has finished. This plugin provides an implementation of the model API for the [**Wflow**](https://github.com/Deltares/Wflow.jl) model. |
| Value proposition | Setting up distributed hydrological models typically requires many (manual) steps to process input data and might therefore be time consuming and hard to reproduce. Especially improving models based on global-local geospatial datasets, which are rapidly becoming available at increasingly high resolutions, might be challenging. HydroMT-Wflow aims to make the Wflow model building and updating processes fast, modular and reproducible and to facilitate the analysis of the model results.  The HydroMT-Wflow plugin can be used as a command line application, which provides commands to build, update and clip a Wflow model with a single line, or from Python to exploit its rich interface. You can learn more about how to use HydroMT-Wflow in its online documentation. For a smooth installation experience, we recommend installing HydroMT-Wflow and its dependencies from conda-forge in a clean environment, see the installation guide. |
| Users of the Component | * Expert users * Hydrologists |
| User Documentation | [**https://deltares.github.io/hydromt\_wflow/latest/**](https://deltares.github.io/hydromt_wflow/latest/) |
| Technical Documentation | [**https://deltares.github.io/hydromt\_wflow/latest/**](https://deltares.github.io/hydromt_wflow/latest/) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/hydromt\_wflow**](https://github.com/Deltares/hydromt_wflow) |
| Language | Python |

### RA2CE

|  |  |
| --- | --- |
| Component name and logo | **RA2CE** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-ra2ce**](https://www.intertwin.eu/article/thematic-module-ra2ce) |
| Description | RA2CE helps to quantify the resilience of critical infrastructure networks, prioritize interventions and adaptation measures and select the most appropriate action perspective to increase resilience considering future conditions. |
| Value proposition | The RA2CE - Resilience Assessment and Action perspective for Critical infrastructurE – model has been developed to support infrastructure owners and operators in resilience assessment and adaptation decision-making and has been applied in several settings such as the Netherlands, Philippines, Myanmar, Dominican Republic and Albania.  The current capabilities focus on mapping the exposure, criticality, and vulnerability as well as the forthcoming prioritisation of locations to take actions based on cost benefit assessment. For further assessment of indirect impacts, inclusiveness and equity principles can be applied. In adaptation and planning studies the platform enables to perform cost-benefit assessments including an uncertain future |
| Users of the Component | * Expert users * Climate resilience specialists |
| User Documentation | [**https://deltares.github.io/ra2ce/index.html**](https://deltares.github.io/ra2ce/index.html) |
| Technical Documentation | [**https://deltares.github.io/ra2ce/index.html**](https://deltares.github.io/ra2ce/index.html) |
| Responsible | **Deltares** |
| Licence | GNU GPL v3 |
| Source code | [**https://github.com/Deltares/ra2ce**](https://github.com/Deltares/ra2ce) |
| Language | Python |

### Hython Wflow\_SBM Hydrological Model

|  |  |
| --- | --- |
| Component name and logo | **Hython\_sbm** |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-hython-wflow\_sbm-hydrological-model/**](https://www.intertwin.eu/article/thematic-module-hython-wflow_sbm-hydrological-model/) |
| Description | The Hython package enables the development of deep learning based surrogates of grid-based semi-distributed and distributed hydrological models, and it enables the calibration of the model’s parameters exploiting satellite-based products. In particular, Hython\_sbm is customized to emulate Wflow\_sbm’s vertical fluxes and states (soil moisture, evapotranspiration, snow water equivalent, etc.), and to calibrate the parameters by leveraging satellite-based products. |
| Value proposition | Traditional distributed hydrological models are complicated to set up, computationally expensive, and challenging to calibrate. One of the negative consequences is that they often lack an estimation of the output uncertainty. Hython\_sbm, as a faster and reliable surrogate model, enables the pixel-by-pixel calibration of the hydrological model parameters, by leveraging satellite products. In addition, thanks to its increased performance and flexibility, it provides an estimation of the output uncertainty.  The publication of the Hython\_sbm module as an application package and openEO process, exposes its functionalities to the openEO user interface, where it can be consequently integrated in custom data and modelling workflows. This reduces dramatically the costs for setting up, training, evaluating the model’s outputs and facilitating the experimentation with different data inputs and model parameters.  The module can be useful to researchers investigating drought prediction and forecasting, and to public authorities in the field of agriculture and river basin management to identify areas potentially affected by hydrological or agricultural drought. |
| Users of the Component | 1. Researchers, 2. local/Regional public authorities in the field of agriculture, 3. hydrology and river basin management authorities, 4. journalist for environmental topics with little expertise about technical data. |
| User Documentation | [**https://github.com/interTwin-eu/hython/blob/main/README.md**](https://github.com/interTwin-eu/hython/blob/main/README.md) |
| Technical Documentation | [**https://github.com/interTwin-eu/hython/blob/main/README.md**](https://github.com/interTwin-eu/hython/blob/main/README.md) |
| Responsible | **EURAC** |
| Licence | [**CC-BY-4.0 Licence**](https://github.com/masawdah/model_ecaas_agrifieldnet_silver/blob/main/LICENSE) |
| Source code | [**https://github.com/interTwin-eu/hython**](https://github.com/interTwin-eu/hython) |
| Language | Python, PyTorch |

### Functionalities developed since the last release

Since the previous release (D7.5), the flood-related thematic modules under Task 7.6 have expanded in scope and maturity of the Digital Twin workflows. Two major new modules have been added to the interTwin ecosystem: WFLOW, a distributed hydrological model for rainfall-runoff simulation, and RA2CE, a road accessibility model that quantifies network disruptions caused by flooding. These modules are now integrated into the climate impact assessment DT. The SFINCS module has been further consolidated, including integration with the EO-based flood mapper to support first-order validation of simulated inundation. A Docker container has been published for Delft-FIAT, enhancing its portability and ease of deployment. In parallel, FloodAdapt has undergone structural improvements to its API and scenario configuration workflow. The Jupyter notebooks, which serve as the interface to the DTs, have been expanded to support new scenario definitions, output visualisation, and simulation orchestration.

### Integrations with other DTE components

Workflow execution is coordinated through OSCAR, the DTE’s orchestration service, which enables containerized Common Workflow Language (CWL) pipelines to be automatically triggered based on scenario files and configuration inputs. This integration allows seamless offloading of computationally intensive tasks—such as SFINCS and WFLOW simulations—to Kubernetes or HPC resources, ensuring scalability and reproducibility. Parallel to this, work is ongoing to connect the DT modules to the interTwin Rucio-based Data Lake. This integration will enable structured storage and access to input datasets (e.g., EO-derived flood maps, terrain models, climate forcings) and simulation outputs (e.g., NetCDF, GeoTIFF, CSV). The goal is to ensure that all DT workflows can interface directly with the federated data infrastructure, allowing users to trace, share, and re-use results efficiently across different scenarios and deployments. Currently, all required input data for the demonstrators is managed on the Data Lake and integrated into the demonstrator workflows. Similar management of intermediate data is work in progress, particularly the capability to trigger OSCAR services directly from the Data Lake will be completed by the end of the project.

### Pilots and testing activities

The flood Digital Twin modules developed have been tested and demonstrated in two key pilot regions: Northern Germany and the Humber Estuary (UK). These pilots have validated the end-to-end workflows implemented using Jupyter Notebooks, which serve as the primary user interface for both model developers and DT users. In the Northern Germany pilot, the workflow was used to simulate a dike breach scenario during Storm Babet (October 2023), comparing inundation outputs from SFINCS against EO-derived flood maps from Sentinel-1. This helped validate the combined use of EO data and hydrodynamic modelling for post-event analysis. In the Humber Estuary, testing focused on the application of the climate impact assessment DT, integrating WFLOW, SFINCS, Delft-FIAT, and RA2CE to explore long-term changes in flood hazard and infrastructure vulnerability under different land use and climate scenarios. The Jupyter Notebooks allow users to define areas of interest, configure model parameters, and run “what-if” simulations with visual outputs that support decision-making.

## T7.7 Fast particle detector simulation

### 3DGAN and CaloINN

|  |  |
| --- | --- |
| Component name and logo | 3DGAN  (Deep Learning models for generation of calorimeter energy depositions) |
| Page on interTwin website | [**https://www.intertwin.eu/article/thematic-module-3dgan**](https://www.intertwin.eu/article/thematic-module-3dgan) |
| Description | Two generative models: 3DGAN - a generative adversarial network, and CaloINN - a normalising flow model, use different approaches to simulate High Energy Physics (HEP) calorimeter output. Calorimeters are special HEP detectors that record particles through the measurement of the energies deposited by them [[**R11**](#_References)]. |
| Value proposition | Detector simulations allow scientists to design detectors and perform physics analyses. The simulation toolkit that has been developed and performs particle physics simulations based on Monte Carlo (MC) methods is Geant4.  The detailed particle MC simulations are inherently slow, especially in simulating a particle passage through a calorimeter. Simulations have a crucial role in HEP experiments, and at the same time are very resource-intensive from the computing perspective. Therefore, HEP community is highly motivated to explore fast alternatives, with deep learning based fast simulation being the most promising.  Generative models are a fast alternative to MC, with remarkable results in terms of speed up. 3DGAN was the first effort where the detector output was generated employing three dimensional convolutions, an approach for retaining correlations in all three spatial dimensions [[**R11**](#_References)]. CaloINN [[**R14**](#_References)] is a more modern model implementing a sequence of invertible layers to learn a transformation from a simple known distribution to a complex distribution of calorimeter output. |
| Users of the Component | Expert Users and Developers |
| User Documentation | [**https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main**](https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main) |
| Technical Documentation | [**https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main**](https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main) |
| Responsible | **CERN** & **CNRS** |
| Licence | MIT |
| Source code | [**https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main**](https://github.com/interTwin-eu/DetectorSim-3DGAN/tree/main) |
| Language | Python |

### Functionalities developed since the last release

CaloINN pipeline has been implemented in itwinai (excluding the inference step whose migration to itwinai is being finalised). The logging on the loss function values and metrics have been connected with the MLFlow tool within itwinaI.

Study on CaloINN performance improvement is being finalised, targeting the issue of distribution tails mismodelling, common for generative models.

### Integrations with other DTE components

Both models are integrated with itwinai component (T6.5). This provides simplified access to distributed computing, experiment tracking tools and pipeline setting via configuration files. The configuration files can be used to set the modules execution order, network architectures and their training parameters. 3DGAN uses OSCAR for workflow execution (T6.1), InterLink for federated computing (T5.1) and SQAaaS for code quality assessment (T62). Both 3DGAN and CaloINN are using model catalogues on MLflow.

### Integrations with DT Applications

The components developed are part of the DT Application on Detector Simulation from T4.2.

### Pilots and testing activities

3DGAN was tested on a custom dataset produced with Geant4 for a CLIC[[21]](#footnote-21) detector. CaloINN implementation in itwinai was tested on CaloChallenge dataset 1 [[**R12**](#_References)] containing sets for two different particle types passing through ATLAS[[22]](#footnote-22) detector.

# Conclusions

As the interTwin project approaches its conclusion, Work Package 7 has successfully delivered a rich and diverse suite of 29 thematic modules. This deliverable provides the final report on the development and integration status of these modules designed to support the DT applications within the interTwin project. They cover both the environmental and physics domains and form a crucial layer of the interTwin ecosystem by offering domain-specific functionalities that address real scientific and operational challenges. Their development has been guided by a continuous co-design with user communities and by alignment with the technical vision and architecture defined in the project, ensuring they meet scientific needs while fully integrating into the broader DTE infrastructure.

To support the environmental DT applications, a total of 20 thematic modules have been fully developed, with the Dask Flood Mapper module being newly added, providing advanced functionalities for data processing, modelling, and machine learning that are essential for building Digital Twins addressing climate change, extreme events, and hydrological hazards. In addition, to support the physics DT applications, WP7 has delivered 9 fully developed thematic modules, with the ANNALISA module being newly added, offering significant capabilities for simulation, data analysis, and anomaly detection in high-energy physics, gravitational wave astronomy, and radio astronomy. Together, these modules offer the scientific community state-of-the-art tools to build digital twins that can accelerate discovery and improve simulation fidelity.

An important aspect of the development of these modules has been to ensure that they do not stand alone but are integrated into the Digital Twin Engine. This has been realised through the adoption of open standards, containerisation, and consistent APIs, which allow the modules to interact seamlessly with the core modules and infrastructure services developed in Work Packages 5 and 6. More specifically, openQxD, normflow, GlitchFlow and ANNALISA are implemented as itwinai plugins, enabling distributed training, streamlined experiment tracking and flexible configuration. Glitchflow, 3DGAN and CaloINN also use MLflow for model cataloguing, while 3DGAN and CaloINN benefit from InterLink for federated computing and SQAaaS for software quality. ML TC Detection and ML4Fires combine itwinai, MLflow, and yProv4ML for provenance tracking and logging, and connect to RUCIO and Ophidia for managing climate data pipelines, deploying via InterLink. xtclim also integrates with itwinai and RUCIO for streamlined data access. eddiesML focuses on transparent provenance with yProv4ML, while downscaleML uses OpenEO for distributed data processing and publishes outputs to the interTwin STAC catalogue. SFINCS, WFLOW, Delft-FIAT and RA2CE orchestrate containerised workflows with OSCAR and store data in RUCIO. All thematic modules are delivered as open-source software with thorough documentation and are easily reusable and extensible by the wider scientific community.

In summary, the work carried out in WP7 has delivered a prototype set of domain-specific modules that substantially enrich the functionality of the DTE. These modules enable users to build sophisticated, data-driven DTs that respond to complex scientific and societal challenges in environmental monitoring, climate adaptation, and fundamental physics. Through close integration with the DTE’s infrastructure and core components, they demonstrate the feasibility and value of an open and extensible DT platform. These results represent a significant step forward in operationalising the digital twin paradigm for science and will remain a durable asset for the community, supporting continued innovation and collaboration beyond the lifetime of the interTwin project.

# References

|  |  |
| --- | --- |
| **Reference** | |
| **No** | **Description / Link** |
| **R1** | **interTwin D7.3 First version of the thematic modules for the environment domain** (Version 1 Approved by the EC). Fiore, S. et al., (2023).  DOI: [**10.5281/zenodo.14974970**](https://zenodo.org/records/14974970) |
| **R2** | **interTwin D7.2 Report on requirements and thematic modules definition for the physics domain** (Version Final). Tsolaki K. et al., (2023).  DOI: [**10.5281/zenodo.8036996**](https://zenodo.org/doi/10.5281/zenodo.8036996) |
| **R3** | **interTwin D7.4 First version of the thematic modules for the physics domain** (Version 1 Approved by the EC). Sinha Ray G. et al., (2023).  DOI: [**10.5281/zenodo.14975005**](https://zenodo.org/records/14975005) |
| **R4** | **interTwin D7.6 Update report on requirements and thematic modules functionalities for the physics domain** (Version 1 Approved by the EC). Vallero S. et al., (2024).  DOI: [**10.5281/zenodo.14975072**](https://zenodo.org/records/14975072) |
| **R5** | **openQ\*D code: a versatile tool for QCD+QED simulations**. I. Campos,  P. Fritzsch, M. [Hansen](https://link.springer.com/article/10.1140/epjc/s10052-020-7617-3#auth-Martin-Hansen), M. K. Marinkovic, A. Patella, A. Ramos & N. Tantalo  The European Physical Journal C , 80, Article number: 195 (2020)  DOI:[**10.1140/epjc/s10052-020-7617-3**](https://doi.org/10.1140/epjc/s10052-020-7617-3) |
| **R6** | **Flow-based generative models for Markov chain Monte Carlo in lattice field theory**. M. S. Albergo, G. Kanwar, and P. E. ShanahanPhys. Rev. D 100, 034515 (2019)  DOI: [**10.1103/PhysRevD.100.034515**](https://link.aps.org/doi/10.1103/PhysRevD.100.034515) |
| **R7** | **Normalizing Flows: An Introduction and Review of Current Methods.**  **Ivan Kobyzev;** Simon J.D. Prince; Marcus A. Brubaker. IEEE Transactions on Pattern Analysis and Machine Intelligence (Volume: 43, Issue: 11, 01  November 2021)  DOI: [**10.1103/PhysRevD.100.034515**](https://link.aps.org/doi/10.1103/PhysRevD.100.034515) |
| **R8** | **Efficient modeling of trivializing maps for lattice**  **Φ4 theory using normalizing flows: A first look at scalability.**  Luigi Del Debbio, Joe Marsh Rossney, and Michael Wilson  Phys. Rev. D 104, 094507 – Published 15 November 2021  DOI: [**10.1103/PhysRevD.104.094507**](https://link.aps.org/doi/10.1103/PhysRevD.104.094507) |
| **R9** | **Generative models for scalar field theories: how to deal with poor scaling?**  Javad Komijani and Marina K. Marinkovic; The 39th International Symposium on Lattice Field Theory, 2022, Bonn, Germany;  DOI:[**10.48550/arXiv.2301.01504**](https://doi.org/10.48550/arXiv.2301.01504) |
| **R10** | **Normalizing flows for SU($N$) gauge theories employing singular value decomposition;** Javad Komijani and Marina K. Marinkovic; The 41th International Symposium on Lattice Field Theory, 2024, Liverpool, UK;  DOI:[**10.48550/arXiv.2501.18288**](https://doi.org/10.48550/arXiv.2501.18288) |
| **R11** | **Fast simulation of a high granularity calorimeter by generative**  **adversarial networks.** Khattak, G.R., Vallecora, S., Carminati, F. et al. Eur.  Phys. J. C 82, 386 (2022).  DOI: [**10.1140/epjc/s10052-022-10258-4**](https://doi.org/10.1140/epjc/s10052-022-10258-4) |
| **R12** | **CaloChallenge 2022: A Community Challenge for Fast Calorimeter**  **Simulation;** C. Krause et al. (2024)  DOI: [**10.48550/arXiv.2410.21611**](https://doi.org/10.48550/arXiv.2410.21611) |
| **R13** | **Normalizing Flows: An Introduction and Review of Current Methods;**  I. Kobyzev, S. J.D. Prince, and M. A. Brubaker (2020)  DOI: [**10.48550/arXiv.1908.09257**](https://doi.org/10.48550/arXiv.1908.09257) |
| **R14** | **Normalizing Flows for High-Dimensional Detector Simulations;**  F. Ernst et al. (2025)  DOI: [**10.48550/arXiv.2312.09290**](https://arxiv.org/abs/2312.09290) |
| **R15** | **Generative Adversarial Networks.** Ian J. Goodfellow et al. (2014)  DOI: [**10.48550/arXiv.1406.2661**](https://doi.org/10.48550/arXiv.1406.2661) |
| **R16** | **interTwin D7.8 Final version of the thematic module for the physics domain** (V1 Under EC review). Sinha Ray G. et al., (2025).  DOI: [**10.5281/zenodo.14931995**](https://doi.org/10.5281/zenodo.14931995) |
| **R17** | **interTwin D4.6 Final Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics** (V1 Under EC review). Tsolaki, K. et al., (2025).  DOI**:** [**10.5281/zenodo.15120027**](https://doi.org/10.5281/zenodo.15120027) |
| **R18** | **Fast radio bursts at the dawn of the 2020s.** Petroff, E., Hessels, J. W. T., Lorimer, D. R. (2021)  DOI: [**10.48550/arXiv.2107.10113**](https://doi.org/10.48550/arXiv.2107.10113) |
| **R19** | **ML-based Pipeline for Pulsar Analysis (ML–PPA);** Kazantsev, A., Oelkers, T., Pidopryhora, Y., Saha, T., Trattner, H., and Heßling, H.;  [**https://gitlab.com/ml-ppa/gitlab-profile/-/blob/main/PUNCH\_interTwin\_project.pdf**](https://gitlab.com/ml-ppa/gitlab-profile/-/blob/main/PUNCH_interTwin_project.pdf) |
| **R20** | **interTwin D7.1 Report on requirements and thematic modules definition for the environment domain** (Version Final). Claus M. et al., (2023).  DOI: [**10.5281/zenodo.10417158**](https://zenodo.org/records/10417158) |
| **R21** | **interTwin 7.5 Updated report on requirements and thematic modules functionalities for the environment domain** (Version Approved by the EC). Schramm, M. et al., (2024).  DOI: [**10.5281/zenodo.14975021**](https://zenodo.org/records/14975021) |
| **R22** | **interTwin D7.7 Final version of the thematic module for the environment domain** (Version 1 Under EC Review). Elia, D. et al., (2025).  DOI: [**10.5281/zenodo.14918025**](https://zenodo.org/records/14918025) |
| **R23** | **Spatial structures of emerging hot & dry compound events over Europe from 1950 to 2023,** Schmutz, J., Vrac, M., François, B., and Bulut, B. EGUsphere (2025) [preprint], **https://doi.org/10.5194/egusphere-2025-461.** |

1. D4.7: Final version of the DTs capabilities for climate change and impact decision support tools including validation reports [↑](#footnote-ref-1)
2. D4.8: Final version of the DT capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics including validation reports [↑](#footnote-ref-2)
3. <https://github.com/jkomijani/normflow_> [↑](#footnote-ref-3)
4. <https://www.ibergrid.eu/2024-ibergrid-porto/> [↑](#footnote-ref-4)
5. <https://slurm.schedmd.com> [↑](#footnote-ref-5)
6. <https://gitlab.com/rcstar/openQxD> [↑](#footnote-ref-6)
7. <https://www.intertwin.eu/article/infrastructure-component-teapot> [↑](#footnote-ref-7)
8. SKAO: <https://www.skao.int/en> [↑](#footnote-ref-8)
9. MeerKAT Radio Telescope: <https://www.sarao.ac.za/gallery/meerkat/> [↑](#footnote-ref-9)
10. ASKAP-radio telescope: <https://www.csiro.au/en/about/facilities-collections/atnf/askap-radio-telescope> [↑](#footnote-ref-10)
11. Predicted data rate for an SKA pathfinder like MeerKAT is of order 10 Gbytes/s or up to 1 Pbytes/day. The SKA itself is expected to produce up to 200 Pbytes/day, which is ~70 exabytes per year. To put it into perspective, the latter is about the same as the expected data rate of CERN’s LHC after the High-Luminosity upgrade (60 exabytes per year) and at about the same time (SKA’s first light is expected in 2027 and the High-Luminosity LHC should go online in 2029). [↑](#footnote-ref-11)
12. Known pulsars have periods from a few milliseconds to 8 seconds, thus over a typical observational session of several hours one can observe many pulses, which makes pulsars relatively easy to detect and observe. However, if we imagine a transient phenomenon similar to a pulse of a pulsar, but either non-periodic or with periods of order of hours or days, discovering it is close to impossible except by sheer luck. [↑](#footnote-ref-12)
13. Examples of such transients that attract a lot of attention in the radio astronomical community are “fast radio bursts” (FRBs), see e.g. [**R18**](#_References) and references thereof. [↑](#footnote-ref-13)
14. Even in the best case scenario when a special “target of opportunity” (ToO) event is expected, and a change of scheduling is proposed in advance for all the observatories involved, the actual triggering of such an event is a complicated and disruptive procedure involving many exchanges between various personnel of many institutions, thus the response time is rarely shorter than a day. Using an automated decision-making system with pre-approved criteria can change this to minutes, most of the time taken to actually reposition the telescopes. [↑](#footnote-ref-14)
15. Radio Telescope Effelsberg: <https://www.mpifr-bonn.mpg.de/en/effelsberg> [↑](#footnote-ref-15)
16. itwinai: https://www.intertwin.eu/article/core-dte-module-itwinai [↑](#footnote-ref-16)
17. MLFlow:<https://mlflow.org/> [↑](#footnote-ref-17)
18. TensorBoard: <https://www.tensorflow.org/tensorboard?hl=it> [↑](#footnote-ref-18)
19. Vega:<https://en-vegadocs.vega.izum.si/introduction/> [↑](#footnote-ref-19)
20. GEANT4:[https://geant4.web.cern.ch](https://geant4.web.cern.ch/) [↑](#footnote-ref-20)
21. CLIC detector:<https://clicdp.web.cern.ch/> [↑](#footnote-ref-21)
22. ATLAS detector:<https://atlas.cern/Discover/Detector> [↑](#footnote-ref-22)