

interTwin

D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational- wave Astrophysics

Status: FINAL

Dissemination Level: public




Funded by the
European Union

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

Document Description

D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

Work Package number 4

Document type	Deliverable		
Document status	FINAL	Version	1
Dissemination Level	Public		
Copyright Status	 <p>This material by Parties of the interTwin Consortium is licensed under a Creative Commons Attribution 4.0 International License.</p>		
Lead Partner	CSIC		
Document link	https://documents.egi.eu/document/3938		
DOI	10.5281/zenodo.8321134		
Author(s)	<ul style="list-style-type: none"> • Kalliopi Tsolaki (CERN) • Sofia Vallecorsa (CERN) • Sara Vallero (INFN) • Massimiliano Razzano (INFN) • Javad Komijani (ETHZ) • Yurii Pidopryhora (MPG) • Isabel Campos (CSIC) • Andrea Manzi (EGI Foundation) 		
Reviewers	<ul style="list-style-type: none"> • Jorge Gomes (LIP) • Rakesh Sarma (FZJ) 		
Moderated by:	<ul style="list-style-type: none"> • Andrea Manzi (EGI Foundation) • Levente Farkas (EGI Foundation) 		
Approved by	Paul Millar (DESY) on behalf of TCB		



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

Abstract

Key Words

Architecture, design, capabilities, Digital Twin Engine, High Energy Physics, GW Astrophysics, Radio Astronomy

This deliverable describes the capabilities that the architecture design of a Digital Twin Engine (DTE) has to provide in order to be able to support the use cases coming from the High Energy Physics, Radio Astronomy and GW Astrophysics and the implementation of the related DT Applications. It details the functional specifications and requirements analysis for these use cases. Finally, it provides insights into the architecture design decisions made when developing the blueprint architecture of the DTE to specifically address the needs of the use cases.

Revision History

Version	Date	Description	Contributors
V0.1	04/07/2027	ToC	Andrea Manzi(EGI), Levente Farkas (EGI), Isabel Campos (CSIC)
V0.2	10/08/2023	First version ready for internal Review	Kalliopi Tsolaki (CERN) and the other authors
v0.3	18/08/2023	Reviewed version	Jorge Gomes (LIP) and Rakesh Sarma (FZJ)
v0.4	29/08/2023	Version Addressed reviewers comments	Kalliopi Tsolaki (CERN) and the other authors
v0.5	1/09/2023	Version reviewed by TCB	Paul Millar (DESY)
v0.6	05/09/2023	Version ready for QA	Kalliopi Tsolaki (CERN) and the other authors
V1.0	06/09/2023	Final	Andrea Manzi (EGI)

When the terminology/acronyms are available via link below, please remove this table.

Terminology / Acronyms

Term/Acronym	Definition
ML	Machine Learning
GAN	Generative Adversarial Networks
GW	Gravitational Wave
QCD	Quantum Chromodynamics
HEP	High Energy Physics
MC	Monte Carlo
HC-LHC	High Luminosity - Large Hadron Collider
DT	Digital Twins



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

CNN	Convolutional Neural Network
DTE	Digital Twin Engine
HPC	High Performance Compute
WP	Work Package
HPO	Hyper Parameter Optimization
HDF5	Hierarchical Data Format version 5
ONNX	Open Neural Network Exchange
ML-PPA	Machine Learning-based Pipeline for Pulsar Analysis
TB	Terabyte

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



Table of Contents

1	Introduction	7
1.1	Aim of this deliverable	7
1.2	Intended audience of this document	7
1.3	Structure of the document	7
2	DT Applications User Stories	8
2.1	DT Application: Lattice QCD simulation.....	8
2.2	DT Application: Detector simulation	9
2.3	DT Application: Noise simulation for radio astronomy	11
2.4	DT Application: VIRGO Noise detector	12
3	DT Applications Design	14
3.1	DT Application: Lattice QCD simulation.....	14
3.1.1	Advanced Data management for Lattice QCD	15
3.1.2	Generative models using Machine Learning.....	15
3.2	DT Application: Detector simulation	16
3.3	DT Application: Noise simulation for radio astronomy	19
3.4	DT Application: VIRGO Noise detector	21
4	Conclusions	23
5	References	24



Executive summary

This document is deliverable 4.2 of the interTwin project, part of work package 4. It is a report collectively written by the partners of tasks 4.1, 4.2, 4.3 and 4.4, who are directly involved in designing digital twins for the physics domain (High Energy Physics, Radio Astronomy, and Gravitational Waves Astrophysics). This report aims to explain the motivation behind development of specific digital twins, the architecture design, who will be the beneficiaries, and what may be expected in the future from the specific digital twins. The deliverable will be followed by a final version of the document (D4.6) to be delivered at M30 of the project, which will highlight the final architecture design of the Digital Twins.

Furthermore, each DT Application in the physics domain requires interaction with different types of stakeholders, either DT operators or DT end users (these stakeholder can be experts in the field or users who are not experts but are interested in using the results), hence user stories have been used, discussed, and presented in the document to understand the requirements of the different stakeholders involved.

Finally, the various modules from different Work Packages (WPs) within the project are highlighted to show how the DT Applications will interact with the interTwin DTE for the purposes of processing data, composing workflows, and visualising results.



1 Introduction

1.1 Aim of this deliverable

The overall objective of deliverable 4.2 is to provide an overview of the DT Applications, their features, and their architecture design for the physics domain (T4.1, T4.2, T4.3, T4.4) and their key requirements in the interTwin project. A **Digital Twin Application** is a user interface implementation of a DT. DT applications are the consumers of the capabilities offered by the interTwin Digital Twin Engine (DTE), therefore they introduce use case-specific requirements.

1.2 Intended audience of this document

The main audiences for the deliverable 4.2 are the developers and end users

For the DT application and DTE **developers**: this deliverable provides them with an opportunity to gain insight into different components, data integration strategies, and computational models required to build an effective DT. Thereby enabling them to incorporate new features, leverage components and workflows, improve scalability, support evolving problems over time, and ensure interoperability. In particular, developers of the DTE need to understand the requirements from the DT Applications while developers of the DT Applications will use this document as a reference for their own requirements and for the way they plan to interact/integrate with the underlying DTE modules.

For DT **end users** and **operators**, this deliverable facilitates data sharing, integration, and analysis among various stakeholders. By establishing a common framework for communication, stakeholders will be able to exchange information, validate models, and collaboratively address their challenges.

1.3 Structure of the document

The structure of this deliverable is as follows. Section 2 describes the user interface and requirements for each digital twin. A detailed table is provided for each digital twin application where details are provided regarding user stories, their requirements (following the MoSCoW method¹), expectations and timeframe for completing the tasks. Section 3 explains architecture design which illustrates the workflow composition within each DT application. It depicts sequential or parallel steps involved in operating each DT, highlighting the input, processing, and expected outcomes.

¹ https://en.wikipedia.org/wiki/MoSCoW_method



2 DT Applications User Stories

2.1 DT Application: Lattice QCD simulation

Task 4.1 aims to produce a Digital Twin (DT) application for simulation of quantum field theories on a lattice, and in particular, lattice Quantum ChromoDynamics (QCD). Two use cases are being explored: a conventional scenario for lattice-QCD simulations, with large scale simulations in High Performance Computing (HPC); and a second scenario, ML-based simulations, which is an area under development in the community.

Large scale Monte Carlo simulations of Lattice QCD take place in major HPC infrastructures either at national or international level. In the framework of the project we want to explore and implement the new federated data capabilities (WP5) with several purposes. From the infrastructure support point of view increasing redundancy and enabling fast data transfer is fundamental, as many challenges in current competitive simulations today require the transfer of several 100s of TBs between HPC centres. An additional beneficial point for end users is the flexibility of data lakes when it comes to access and sharing data. At the moment users are subject to the policies of HPC centres, which prevent sharing of data between users not having a classic account ssh-based in their system.

In the particular case of data analysis, Jupyter Notebooks are used to take care of the full analysis workflow (of the several steps involving different codes). Connecting those notebooks with the data lake is therefore an interesting possibility.

For the second scenario we aim at exploring how ML can support conventional Monte Carlo simulations. For example, finding the right parameter point to perform a large-scale simulation in QCD (“tuning the simulation”) is in general tricky, and requires the application of very advanced statistical methods and a consistent investment in CPU hours. The identification of phase transitions points is a clear example. The usage of Normalizing Flows [R1] for the purpose of generating field configurations to support the conventional Monte Carlo simulations is a very active research field. The trained models will be used in a pipeline for generating field configurations that can be used to calculate various observables of interest relevant to physics applications, a reference observable is the energy of the lattice plaquette, whose autocorrelation length is often used as a reference to estimate the statistic properties of the simulation. In section 3.1 a description of the pipeline stages is given. The main stakeholder of the DT is the expert who designs architectures for various quantum field theories, trains the model, and investigates the efficiency of the trained model. The other stakeholders are the physicists that exploit the trained model in their research.

Table 1 – User stories for DT Application: Lattice QCD Application

Ref N	As a	I want to	So that	And it's considered done when	MoSCoW
4.1-1	User performing Monte	Be able to retrieve and transfer Lattice configurations	The user can restart the simulation in a different	The user can access configurations in a data lake using	Must



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

	Carlo simulations	between different HPC systems using a Data Lake	machine of perform data analysis	the AAI solution of the project	
4.1-2	User performing model training.	be able to train a generative model for a given field theory. In particular, the generative method considered here is the method of Normalizing Flows.	field configurations for various quantum field theories can be generated.	the model is so well trained that the efficiency is comparable with the traditional methods of generating field configurations. To quantify the efficiency one needs to measure the autocorrelation in the generated configurations.	Must
4.1-3	User performing data analysis.	generate field configurations using the trained model.	various observables of interest relevant to physics applications can be calculated.	The accuracy of the observables matches the desired level.	Must

2.2 DT Application: Detector simulation

In Task 4.2 a DT application for particle detector simulation is being designed and will be developed.

A methodology that accelerates particle detector simulations, leveraging generative deep learning methods, has already been described and is available in deliverable 7.2 (D7.2) [R3]. Our methodology is using Geant4², a software toolkit for the simulation of the passage of particles through matter, and Generative Adversarial Networks (GAN), a class of machine learning frameworks for approaching generative AI. The technical requirements have been identified, defined, and reported in detail in D7.2.

This section outlines the user stories that define the key functionalities and requirements for our Geant4 and Generative Adversarial Network (GAN) DT Application. These user stories have been identified to reflect the needs of DT operators, including physicists, data scientists, and machine learning engineers. Each user story represents a specific goal from the perspective of the interested stakeholder to guide the development process.

² <https://geant4.web.cern.ch/>



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

Table 2 – User stories for DT Application: Detector Simulation

Ref N	As a	I want to	So that	And it's considered done when	MoSCoW
4.2-1	DT operator	use the Geant4 application to simulate particles passing through a specific detector setup (full/Monte Carlo-based simulation).	the operator can generate data for various scenarios	the system successfully simulates particles passing through the specified detector setup, and generates and stores simulation data for further usage.	Must
4.2-2	Data Scientist	preprocess the simulated data.	it can be used to train a GAN model.	the data scientist has access to the raw simulation data, the system allows for data preprocessing and preparation steps, and then the preprocessed data is suitable for GAN training.	Must
4.2-3	Machine Learning Engineer	train a GAN model on the preprocessed simulated data, with specified model input conditions (e.g. particle's entrance angle, initial energy and type).	the model can generate data that is similar to the original simulated data.	the machine learning engineer can access and input the preprocessed data into the GAN model, the DT provides tools for monitoring and tuning the training process, and additionally the system validates the trained GAN model by providing performance metrics.	Must
4.2-4	Physicist	use the trained GAN model within the Geant4 application during the inference step (fast/ GAN-	they can produce GAN-based simulation data faster, in contrast to using traditional	the physicist can import the trained GAN model into the Geant4 application, the system successfully generates GAN-based simulation data when	Must



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

		based simulation).	Geant4 simulation.	given initial conditions (e.g. particle's entrance angle, initial energy and type), and then compares GAN-based simulation data with traditional Geant4 simulation data for consistency and speed.	
--	--	--------------------	--------------------	--	--

2.3 DT Application: Noise simulation for radio astronomy

Task 4.3 aims to develop a DT of an astronomical source-telescope system, able to generate synthetic output signals identical to the data recorded by a real telescope, which includes both scientifically valuable data and various interference and noise signals. The DT is physics-based: a set of the control parameters will allow adjustment of the output to various sources, detection instruments, and observing conditions. The resulting data is to be used to train ML data-classification tools. A detailed overview of the project has already been provided in the deliverable 7.2 [R3].

The work is split into three parallel and interacting subprojects: astrophysical analysis of the real data, theoretical modelling of the source/telescope system, and development of a fast and scalable C++ implementation. The first of these includes building of the ML-data classification tool for the analysis of the real data, which assigns labels to each data fragment based on the type of signal detected or not detected in it. The label describes the fragment on a basic level as “scientifically important data”, “no signal”, “interference” etc., and in the future may also include more detailed properties. Since the proportions of each data type in the real data flow are very different (e.g. scientifically important data might constitute less than 1% of the data sample), efficient ML training requires synthetic data to be used. That is where the DT comes in, which is developed in the second subproject based on a physical model of the source, its signal transmission and registration. Finally, within the third subproject both the ML tool and the DT are implemented as efficiently as possible.

Table 3 – User stories for DT Application: Noise Simulation for Radio Astronomy

Ref N	As a	I want to	So that	And it's considered done when	MoSCoW
4.3-1	Radio Telescope Operator / On-Site	get DT-generated synthetic data tailored to the	the ML data classifier can be used in flagging the	DT-trained ML data classifier labels the real data by type	Must



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

	Radio Astronomer .	specific observation type, target and conditions, and train the ML data classifier with them.	scientifically worthless data during the observation run to keep the recorded data volume as low as possible.	(science, empty, interference etc.) with a high degree of certainty (~95%).	
4.3-2	Radio Astronomer (responsible for processing and assessing the data).	be able to run DT- generated synthetic data through data processing pipelines and analytic tools.	the pipelines and tools can be debugged and correctly configured prior to the arrival of real data, improving the efficiency and shortening the time before the data release.	there is no apparent difference when running the synthetic data through the relevant pipelines and tools.	Must
4.3-3	Radio Astronomer (scientific analyst, "end user").	use the DT-generated and processed data.	hypotheses about the real data can be tested.	the synthetic data is tried in a scientific analysis of a real project, and the end users are happy with the results.	Should
4.3-4	Radio Astronomer or Software Engineer (data acquisition/ processing pipeline developer).	run the DT and ML data classifier training in parallel configuration on computing clusters.	run time can be decreased to achieve (near) real-time data processing.	near real-time data processing is achieved.	Should

2.4 DT Application: VIRGO Noise detector

The goal of Task 4.4 is to produce a Digital Twin (DT) of the Advanced Virgo interferometer to realistically simulate transient noise in the detector. We will use Generative Adversarial Networks (GANs) to determine the relationship between *strain* data (that measure the deformation induced by the passage of a gravitational wave) and *auxiliary* data (that monitor the status of the detector's subsystems as well as the environmental conditions).



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

The trained model will be used in a pipeline for vetoing and denoising the strain signal in low-latency searches, i.e. those data analysis pipelines that search for transient astrophysical signals on shorter timescales and in almost real time. The high-level architecture of the DT has been defined in deliverable 7.2[3]. The main stakeholder of the DT is the expert user operating the vetoing/denoising pipeline (the *DT Operator*), as well as people working in the Rapid Response Team on shift during the observing period. The other stakeholders are the *physicists* operating the *downstream pipelines* that will use the information provided by the DT. In table 4, for each of the two stakeholders we report a list of requirements that will drive the design of the DT.

Table 4 – User stories for DT Application: VIRGO Noise Detector

Ref N	As a	I want to	So that	And it's considered done when	MoSCoW
4.4-1	DT Operator	make sure that the GAN model is periodically re-trained on most recent data.	the DT realistically simulates the detector response following any change in the experimental conditions.	the model re-training converges and has a good accuracy in reproducing the flux of incoming data.	Must
4.4-2	DT Operator	make sure that the DT is able to identify transient noise (glitches) in incoming data.	the information about an identified glitch can be used to issue a veto decision.	the DT outputs a probability for a given time span of data to contain a glitch.	Must
4.4-3	DT Operator	make sure that the DT is able to reproduce transient noise (glitches) in incoming data.	the information about an identified glitch can be used to denoise the incoming signal.	the DT outputs a signal in which the glitch has been removed (denoised).	Should
4.4-4	DT Operator	make sure that the DT delivers the correct veto flag to downstream pipelines.	downstream pipelines can use this information to decide if further processing of the data or not.	the DT delivers to downstream pipelines a veto decision in the expected format.	Must
4.4-5	DT Operator	make sure that the DT is able to	downstream pipelines can	the DT delivers to downstream	Should



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

		denoise the incoming signal.	search for a gravitational wave signal in incoming data without being biased by glitches.	pipelines a stream of denoised data in the expected format.	
4.4-6	Physicist	be able to use the DT veto information in downstream pipelines.	data containing glitches are discarded from processing.	the information about the probability for data to contain a glitch is received in the expected format.	Must
4.4-7	Physicist	ensure that downstream detection pipelines receive an input of denoised data.	the search for a gravitational wave signal is unbiased by glitches.	the stream of denoised data is received in the expected format and the denoising procedure has not removed any astrophysical signal.	Should

3 DT Applications Design

3.1 DT Application: Lattice QCD simulation

The aim of Lattice QCD is shedding light on the properties of Quantum Chromodynamics in the limit of low energies/strong couplings, where perturbation theory breaks down, and numerical approaches become mandatory. In interTwin the objectives are exploring two use cases addressing the status of Lattice QCD simulations: a classical scenario, with large scale simulations in HPC; and a second scenario, Machine Learning-based simulations, an area under development in the community, at the proof of concept level, therefore requiring few resources.



3.1.1 Advanced Data management for Lattice QCD

Simulations are executed at large scale in HPC systems controlled by a batch system (such as slurm³). The workflow involves generation of configurations and data analysis, both are computing-intensive tasks.

In order to facilitate data analysis, the configurations should be made readily available to the members of the collaboration in a controlled way, for example by using federated identities, and group-based access control. In the most frequent scenario the members of the collaboration should have group-access enabled to read the data. A few of them, those in charge of generating data, should also have writing access rights. A data sharing model following a Data Lake architecture (WP5) would be desirable.

3.1.2 Generative models using Machine Learning

Machine learning techniques are being explored in Lattice QCD in order to facilitate the generation of configurations in complex areas of the parameter space. In this respect, the training of the models is done by comparing the result of the ML technique, with the result of a standard Monte Carlo simulation as described in the scenario above. The accessibility to Lattice configurations is therefore essential to perform the training of the model.

The efficiency of general purpose Monte Carlo algorithms decreases dramatically when the simulations need to take place near critical points due to critical slowing down. This is a general phenomenon in simulations in Physics related to phase transitions, which happens as well in Lattice QCD, for example with simulations at very fine distances that are needed for extrapolation to the continuum limit. Simulations need to take place in areas of the parameter space where topology freezing (among other factors) induce very large autocorrelations.

If Machine Learning could help speed-up the field configuration generation in those parts of the parameter space is a subject under investigation. A series of recent studies suggest that using Normalizing Flows (a class of deep generative models) may help to improve this situation (a block diagram illustrating the method is shown in Figure 1). The underlying idea is using Machine Learning techniques to map the theory of interest to a “simpler” theory, easier to simulate. This approach has the potential to become more efficient than traditional sampling especially when the concept of transfer learning is utilised.

However, the costs associated with the (highly complex) sampling from the path integral, are transferred to the training of a model. The question under investigation is therefore how expensive it is to train a model compared with making a classical Monte Carlo simulation.

³ <https://slurm.schedmd.com>



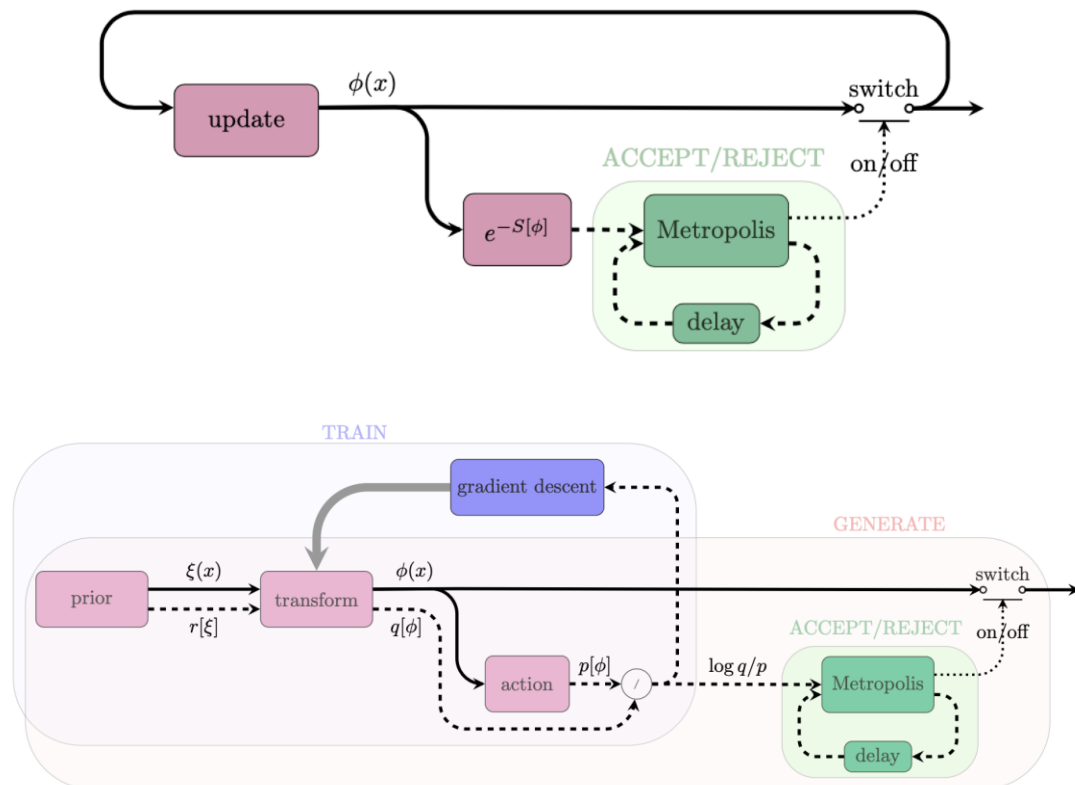


Figure 1. Upper part: Graphical representation of the classical generation of configurations using Monte Carlo algorithms; Lower part: Graphical representation of the Normalizing Flows method including a correcting accept/reject step to account for the fact that the model cannot be perfectly trained.

The purpose of this work is designing better architectures for Machine Learning models so that the acceptance rates become reasonable ($\sim 50\%$ or more) as the volume of the lattice increases. The requirements in terms of resources are not as in the classical Monte Carlo simulation since the methodology is still at the proof of concept level.

3.2 DT Application: Detector simulation

In Deliverable 7.2, the underlying challenges of detector simulation for CERN and the High Energy Physics (HEP) community, as well as the importance of developing a DT digital twin system that integrates simulation methods with machine learning, were analysed and described.

This section provides a comprehensive overview of CERN's digital twin application of a detector simulation. It describes the key steps, from particle simulations to event generation, and subsequent data comparison with real data. The process is explained in detail, highlighting the functionalities at each stage. Furthermore, it illustrates the flexibility in tuning the system to accurately represent various detectors' responses. This explanation is designed to give readers an understanding of the entire workflow design, shedding light on current practices and potential areas of future improvement. It also



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

opens the way for a deeper discussion on the challenges faced, decisions made, and future strategies in the ongoing development of this innovative simulation application. This application consists of two components, the component that incorporates the Geant4-based simulation framework and the deep learning component, which uses deep generative models based on a specified particle detector set up. The two components are encapsulated into two main workflows, the training workflow and the inference workflow, as illustrated in Figure 2. Below, the application functionalities and their specifications included in each workflow are described.

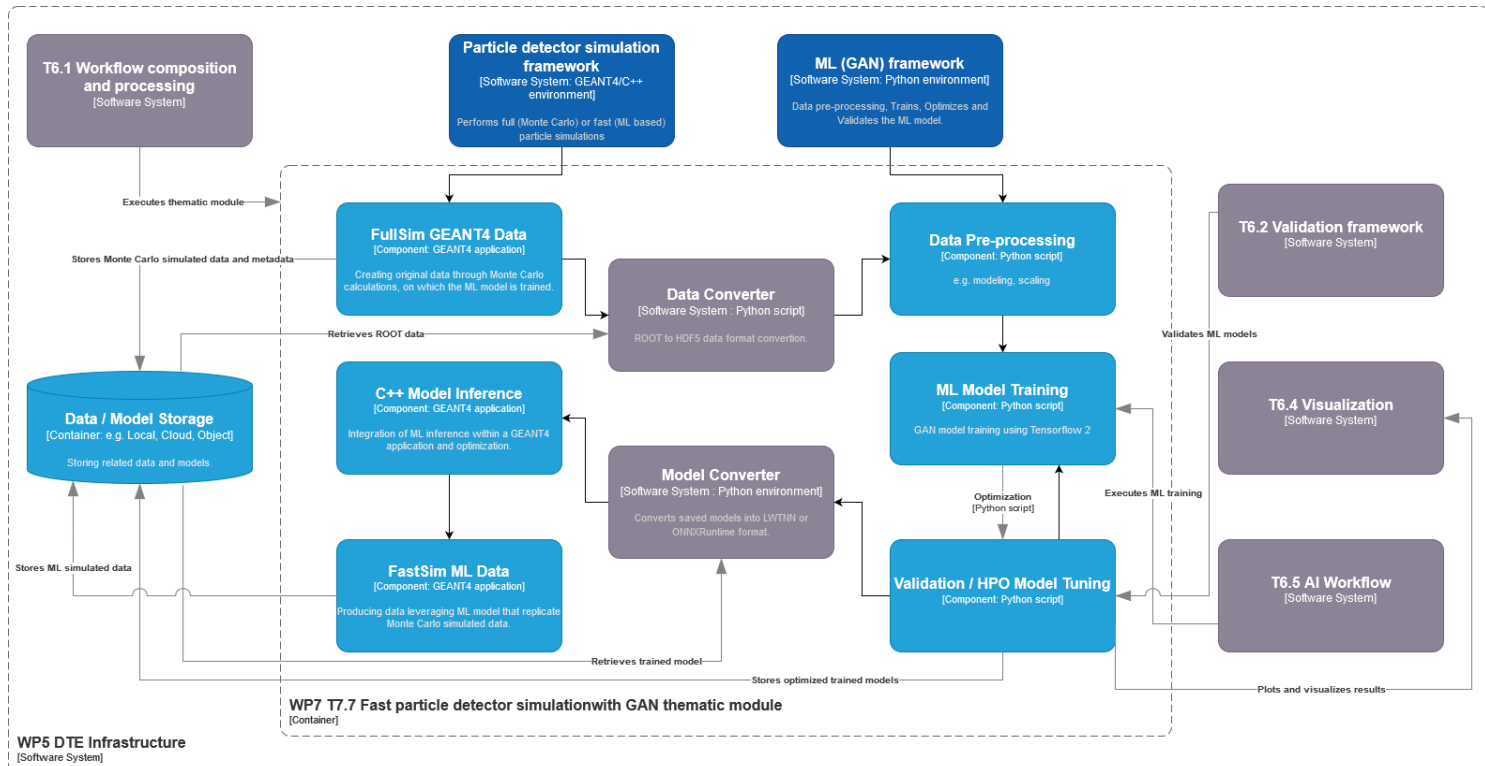


Figure 2 – Fast particle detector simulation using ML techniques high level workflow composition and its connections with other work packages' components

The Geant4 simulation toolkit that consists of an important component of CERN's application, performs particle physics simulations based on Monte Carlo (MC) methods. It constitutes a set of components which include geometry and tracking descriptions, detector response modelling, event management, user interfaces and many other functionalities. Geant4 toolkit is typically used in HEP research projects for complex detectors of which single components (i.e. the calorimeters) are simulated using GANs, as an alternative to the classical MC techniques. Calorimeters are key components of the whole experimental setup, which are responsible for measuring the energy of the particles. Simulating the calorimeters' response using Geant4 is usually a bottleneck for the related research projects. For that reason, generative AI based fast simulation is being leveraged, which generates directly the detector output, without reproducing, step by step, each single particle that interacts with the detector material, in contrast to MC methods.

The training workflow design includes the following functionalities, which run on HPC systems managed by Kubeflow containerized components. Geant4 simulates particle interactions, producing data based on a detector-specific configuration. The produced



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

data consists of the energy measured by the detector sensors, the properties of the initial particle, such as its type, energy, and its trajectory angle with respect to the detector volume, and other metadata. The produced data, in ROOT format, is stored at different data centres, with CERN currently serving as the primary storage site. The Geant4 application will run on HPC systems in a containerized environment.

The data produced from the traditional Geant4 simulation in ROOT format requires conversion into the HDF5 format for further preprocessing before being input into the GAN model. This conversion is currently performed using a Python script, though future considerations include whether to incorporate this conversion within the GEANT4 application or into the training loop or keep it as a separate process. The converted data will then be stored at data centres. Following the ROOT to HDF5 format conversion, the HDF5 data is further preprocessed and transformed into numpy arrays, a process currently incorporated within the model training scripts.

A Generative Adversarial Network (GAN) is trained [R2] on the preprocessed 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 validation and HPO, the model generated data and the Geant4 simulated data distributions will both be visualised. Additional validation techniques are currently being established. Training, validation and HPO processes will run on HPC systems.

At the end, the training workflow stores the optimised models, selected based on validation results, and converts them into the ONNX format for use during inference. Currently, the transformation of the model architecture and weights is performed within a Python script. The model registry where the GAN models are stored is managed by Task 6.5.

The inference workflow design includes the following functionalities which run on HPC systems managed by Kubeflow containerized components. The Geant4 application at this stage initiates a particle, guiding it through the detector until it reaches the bottleneck detector part (the calorimeter), at which the GAN model performs inference. This functionality is incorporated within the Geant4 application, which of course requires the retrieval of the stored ONNX formatted models. The model's output undergoes a detector-specific transformation to convert it into a Geant4 suitable input: the 3D images that the model generates are mapped into the so-called "hits" data consisting of the position (x, y, z coordinates) in the detector (i.e. the sensors positions) and the corresponding energy measurements.

The transformed data is used by the Geant4 framework to complete the process of generating events, simulating the passage of particles through the remaining components of the detector. Data distribution comparisons are drawn between the GAN-generated data and real data (either derived from a traditional Geant4 simulation or data derived from accelerator test beams). These comparisons are essential for validating the efficacy and accuracy of the GAN-generated data.

Finally, based on the results visualised, two possible workflows are proposed for simulation tuning, shown in Figure 3. 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



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

two possible workflows allow for greater flexibility and adaptability in tuning the detector's responses to various particle interactions.

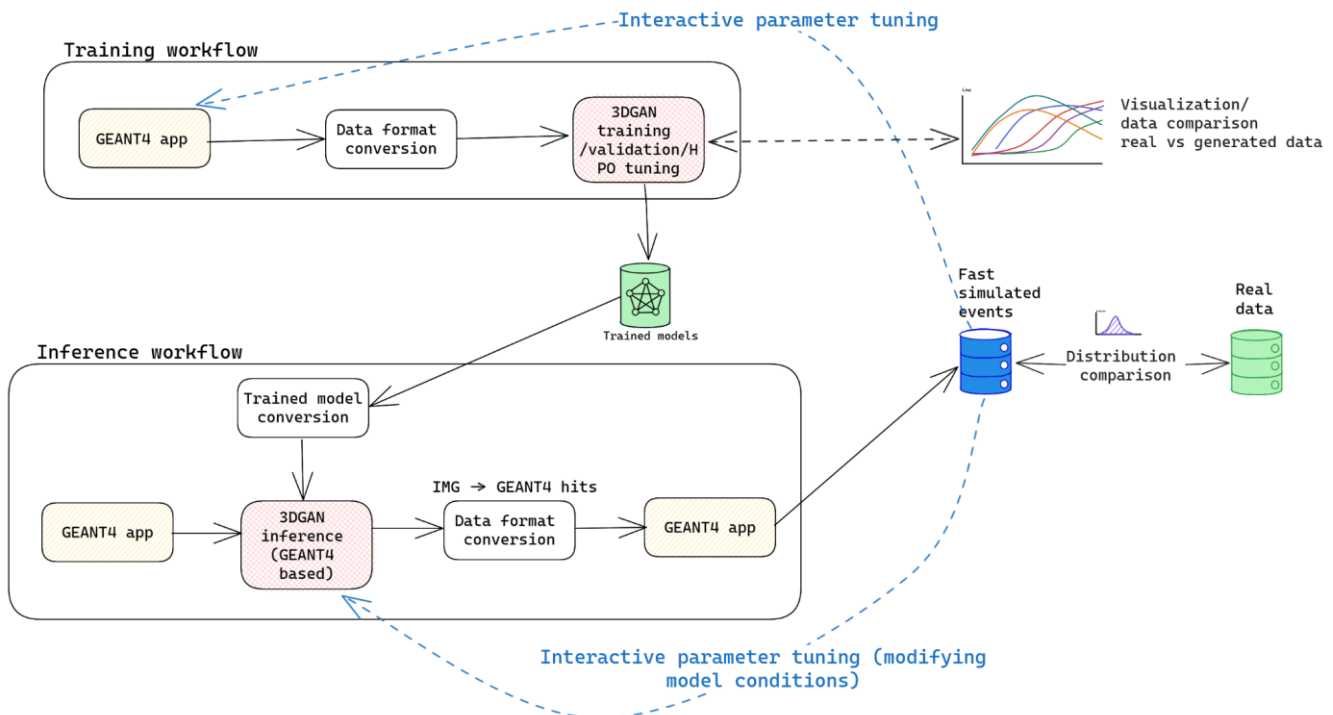


Figure 3 – Detailed graph representation of the training and inference workflows composition (as described above) of the fast particle detector simulation DT utilising 3DGAN approach

3.3 DT Application: Noise simulation for radio astronomy

This DT application addresses the challenge of identifying radio signals from intermittent astrophysical sources, so called ‘transients’ and ‘pulsars’, from large-volume data streams during the data acquisition phase. One of the main tasks is to identify noise and interference signals coming from different sources. The DT recreates the propagation of pulsar signals from the source to radio astronomical antennas and then processing them by radio telescope electronics (see Figure 4), aiming to generate synthetic output signals identical to the data recorded by real telescopes.

It is a part of the development effort of a larger framework called ML-PPA for Machine Learning-based Pipeline for Pulsar Analysis⁴. In addition to the DT, it includes a CNN-based ML classifier of the pulsar data, which can be trained using the DT-supplied data. An essential part of this project is analysis of real astronomical data collected in a dedicated observation (about 20 minutes of data collected by the Effelsberg 100m radio telescope⁵ observing one of the brightest and well-studied pulsars, the Crab pulsar) and

⁴ Andrei Kazantsev, Tim Oelkers, Yurii Pidopryhora, Tanumoy Saha, Marcel Trattner, and Hermann Heßling, “ML-based Pipeline for Pulsar Analysis (ML-PPA)” (~50pp, in preparation)

⁵ Radio Telescope Effelsberg: <https://www.mpifr-bonn.mpg.de/en/effelsberg>



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

creation of empirical simulated data based on it, which provides the material for testing both the ML-classifier and the DT.

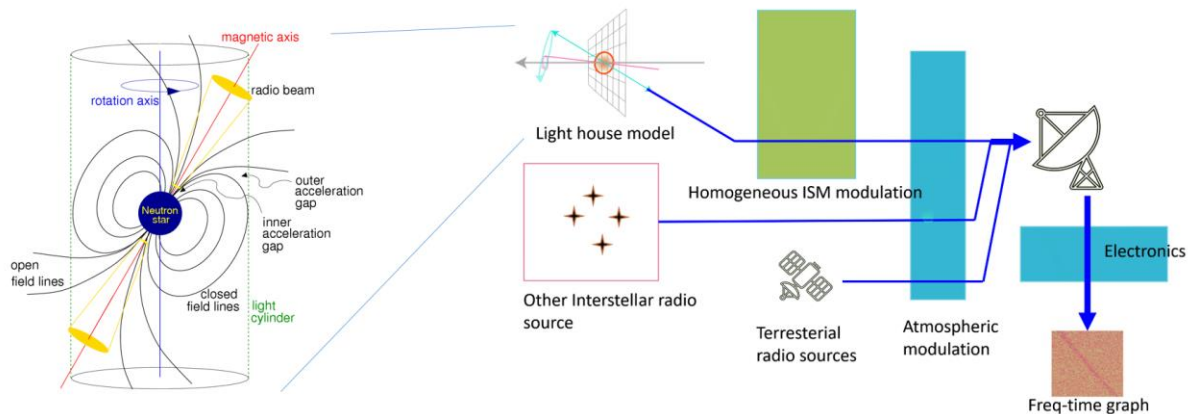


Figure 4: General outline of the DT structure: modelling the astrophysical source (pulsar), transmission of the signal through the interstellar matter, receiving and processing by a radio telescope, adding sources of both natural and artificial interference and noise.

The overall software architecture follows the 3-layer design (see Figure 5). The top layer provides interfaces for the user to develop pipelines, which are developed in WP4, by combining tools and algorithms that are held in the middle layer, which has been already documented in D7.2. The bottom layer enables the creation of containers that can be distributed to data centres via the integration of Workflow tools developed in WP6, where the built-in pipelines can be used to analyse or generate data. For logistical reasons most components are first developed and tested in Python, and then some parts are rewritten in C++ to ensure the best speed and efficiency.

The ultimate goal of this framework is to empower astronomers in their pursuit of uncovering non-trivial astronomical signals and enhancing their ability to process, analyse, and interpret huge volumes of data coming from the next generation of radio telescopes, such as Square Kilometer Array (SKA)⁶ "pathfinders", for example South African MeerKAT⁷ or Australian ASKAP⁸, and then the SKA itself, when it comes online. In the near future we plan to add MeerKAT datasets to the materials used in this project. It is necessary to clarify that this is a future goal, not implemented in the current task. Here we just build a data-classification tool assigning a label⁹ to each data fragment. These labels can be used to filter the data flow in real time, which in turn should diminish significantly the recorded volume of data. But this is not our problem at the moment. Our tool on its own does not require the ability to work with large volumes of data, a trained ML model can handle a typical data acquisition rate of a radio telescope even when running on a mediocre computer. And the computationally intensive task of training the model (where the DT is used) is independent of the data acquisition.

⁶ SKAO: <https://www.skao.int/en>

⁷ MeerKAT Radio Telescope: <https://www.sarao.ac.za/gallery/meerkat/>

⁸ ASKAP radio telescope: <https://www.csiro.au/en/about/facilities-collections/atnf/askap-radio-telescope>

⁹ In the simplest terms boiling down to "scientifically important" and "can be discarded".



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

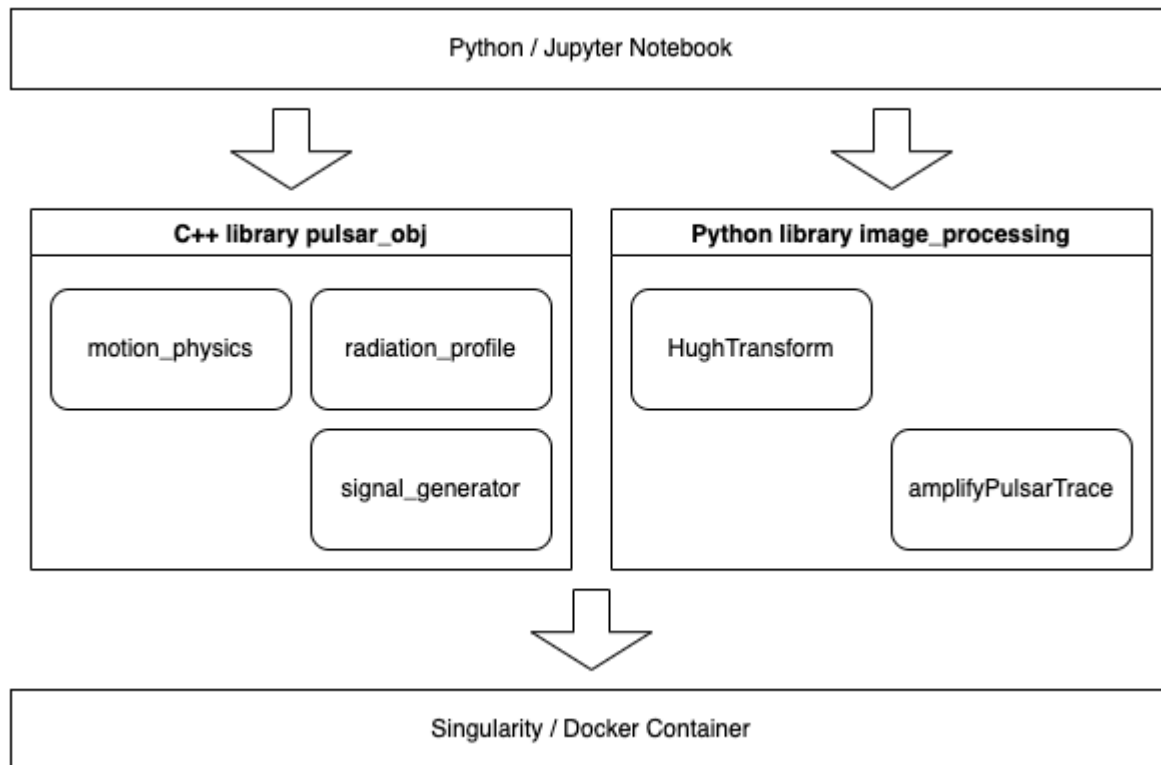


Figure 5: Layered software architecture of the framework ML-PPA

The development also tries to address the issue of poor scalability of available radio astronomical software tools, the aim is to create a package that can be efficiently used for massively parallel computing over the resources offered by the WP5 infrastructure

3.4 DT Application: VIRGO Noise detector

The core of the *VIRGO Noise detector* DT is a Generative Adversarial Network used to find the transfer function of the system producing non-linear noise in the detector output. The information produced by the GAN is used to identify the presence of transient noise (glitches) in incoming data, which could mimic a signal of astrophysical origin (the gravitational wave).

This is achieved by feeding the GAN with two samples of input data:

- the *strain* signal measures the change in the relative distance between two test masses, which is indicative of the deformation of the fabric of space-time induced by the passage of a gravitational wave. This signal results from the composition of several other signals coming from different detector subsystems, and
- the *auxiliary* signals from thousands of sensors that monitor the status of the detector's subsystems as well as the environmental conditions (wind, temperature, seismic motions). Although a subset of these signals is used to compose the *strain*, the rest is expected to be blind to the astrophysical signal and witness only noise.

The GAN model should learn to simulate the signal in the strain channel starting from the signal in a subset of auxiliary channels, therefore reproducing the noise component of

D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

the strain signal only. By comparing the simulated strain signal with the real signal, one should be able to infer the presence of an astrophysical signal.

The DT is organised as a pipeline operating in quasi real-time (low-latency), with the goal of identifying a glitch in the incoming data and passing the information to downstream low-latency pipelines that search for transient astrophysical signals. In the first phase, the information will be passed as a *veto* decision not to process the data if they contain a glitch. In a second phase, also in view of the Einstein Telescope, the DT will output denoised data, in which the glitch has been removed, to be further processed by the search pipelines.

Original Virgo data is in the form of 1D time series, but can be preprocessed in 2D time-frequency representations, which better capture the signal's properties at a visual inspection and are well suited for image-to-image translation GANs.

The usage of 1D or 2D representations of the data will depend on the latency constraints of the DT and has not been decided yet.

Also, in the first phase, data will be handled as files containing a defined time lapse of the signal. In a second phase, they will be handled as *streams*.

The DT is composed of two main subsystems, one for training and one for inference.

The **Training subsystem** is responsible for the periodic re-training of the DT model on a buffered (on disk) subsample of most recent data. The stream of detector data, including the strain and a set of auxiliary channels, needs to be converted to files and stored on a POSIX filesystem. Data is then preprocessed in a format suited for the GAN. We expect the preprocessing step to be executed on distributed resources, possibly the training step as well. From the infrastructure point of view, the DT will leverage the capabilities offered by Tasks 5.1, 5.2 and 5.4. Moreover, it will rely on Task 6.1 for the workflow composition, real-time acquisition and processing. The trained model could be passed to the Inference subsystem either via storage or through a model catalogue. The training step is executed asynchronously to the rest of the pipeline and time constraints are not very stringent.

The **Inference subsystem**, besides applying the trained GAN model to simulate the strain data starting from the auxiliary channels, should also include a functionality to compare the real and simulated signals to extract the probability for data to contain a glitch or to denoise the signal in a later stage. These functionalities will not necessarily be covered by ML models. A preprocessing step is needed to convert the stream of input data in a format suitable for the GAN model, as well as a post processing step to prepare the DT output in a format suitable to be used by the downstream pipelines. Also in this case, there is a dependency from the capabilities provided by Tasks 5.1, 5.2, 5.4 and 6.1.

Both subsystems need to be connected to a monitoring system that collects and displays metrics on training convergence and inference accuracy, to make sure that the DT realistically simulates the detector response following any change in the experimental conditions. The monitoring system should also be capable of sending alerts (e-mail, sms etc..). The functionalities will be provided by the WP6 in particular by the task T6.5 delivering the AI workflow subsystem.

The *DT Operator* is an expert user that monitors and reacts to any problems occurring within the pipeline. In the first phase, we also expect him/her to trigger the re-training step when something is known to have changed in the experimental conditions. In a later stage, we plan to develop an event-driven procedure, leveraging the framework to be provided by Task 6.1, to automatically trigger the re-training procedure.



D4.2 First Architecture design of the DTs capabilities for High Energy Physics, Radio astronomy and Gravitational-wave Astrophysics

Finally, the DT should be interfaced with the Virgo low-latency pipelines, which need to be able to consume the DT output in form of a veto decision (i.e. probability for a given time lapse of data to contain a glitch) or a stream of denoised data. In the latter case, particular care should be taken in making sure that the denoising procedure does not bias the capability to detect a gravitational wave signal in case it is overlapping with a glitch. Figure 6 already presented in D7.2 [R3] shows the C4 model of the DT veto pipeline.

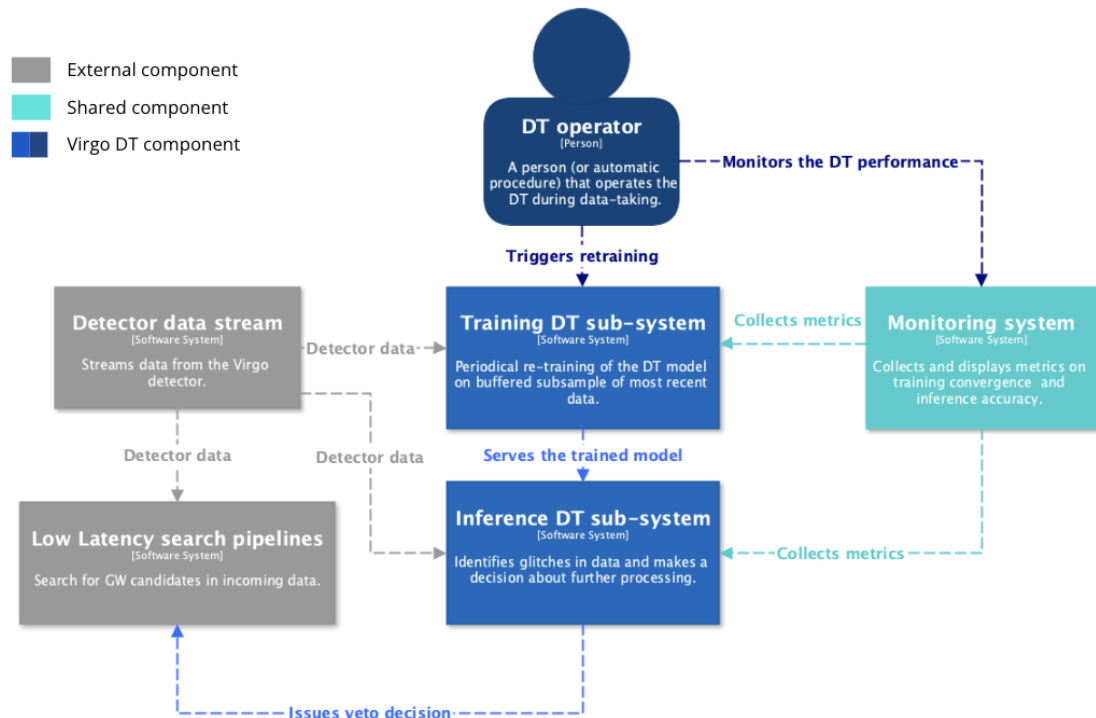


Figure 6: System Context diagram (in the C4 model) of the DT for the veto pipeline.

4 Conclusions

The first version of the design of interTwin DTs applications for WP4 concerned with the physics domain was developed during the first year of the project. In the current deliverable 4.2, the main focus is on defining a preliminary workflow and identifying possible beneficiaries for each DTs application. As part of each task (T4.1, T4.2, T4.3, and T4.4), we described user stories, key requirements, expected outcomes, and the steps that are planned to accomplish their specific goals. Additionally, in Section 3, we outlined the layout and necessary steps that should be considered when designing an individual digital twin.

The next step will be to complete the development of the first version of each DT Application, followed by their release and integration with the DTE corresponding to the deliverable 4.4 planned for April 2024. That deliverable will also serve as feedback for the first DTE components to be released in December 2023, and integrated by the DTs.

5 References

Reference	
No	Description / Link
R1	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: https://doi.org/10.1109/TPAMI.2020.2992934
R2	Khattak, G.R., Vallecorsa, S., Carminati, F. et al. Fast simulation of a high granularity calorimeter by generative adversarial networks. Eur. Phys. J. C 82, 386 (2022). DOI: https://doi.org/10.1140/epjc/s10052-022-10258-4
R3	Kalliopi Tsolaki, Sofia Vallecorsa, David Rousseau, Isabel Campos, Yurii Pidopryhora, Sara Vallero, Alberto Gennai & Massimiliano Razzano. (2023). interTwin D7.2 Report on requirements and thematic modules definition for the physics domain first version (V1 Under EC review). Zenodo. DOI: https://doi.org/10.5281/zenodo.8036997

