PROCEEDINGS OF SCIENCE

Data Analysis Integrated Software System in IHEP, design and implementation

HU Yu[a] **and TIAN Haolai**[a,b,c,*]

[a] *Computing Center, Institute of High Energy Physics, Beijing, 100049, China*
[b] *University of Chinese Academy of Sciences, Beijing, 100049, China*
[c] *Spallation Neutron Source Science Center Dongguan, 523803, China*

E-mail: tianhl@ihep.ac.cn

Daisy (Data Analysis Integrated Software System) has been designed for the analysis and visualisation of data from synchrotron radiation facility and other scientific facilities hosted by Institute of High Energy Physics (IHEP). To address the requirements of the Chinese scientific community, spanning an extensive range from purely algorithmic problems to scientific computing infrastructure, Daisy sets up a cloud-native platform to support on-site data analysis services with fast feedback and interaction. Furthermore, the plug-in based application is convenient to process the expected high throughput data flow in parallel at next-generation facilities such as the High Energy Photon Source (HEPS) and astronomical satellites. The architecture and some applications of Daisy are described in this article.

International Symposium on Grids & Clouds 2022 (ISGC 2022)
21 - 25 March, 2022
*Online, Academia Sinica Computing Centre (ASGC), Taipei, Taiwan****

*Speaker

PoS(ISGC2022)028

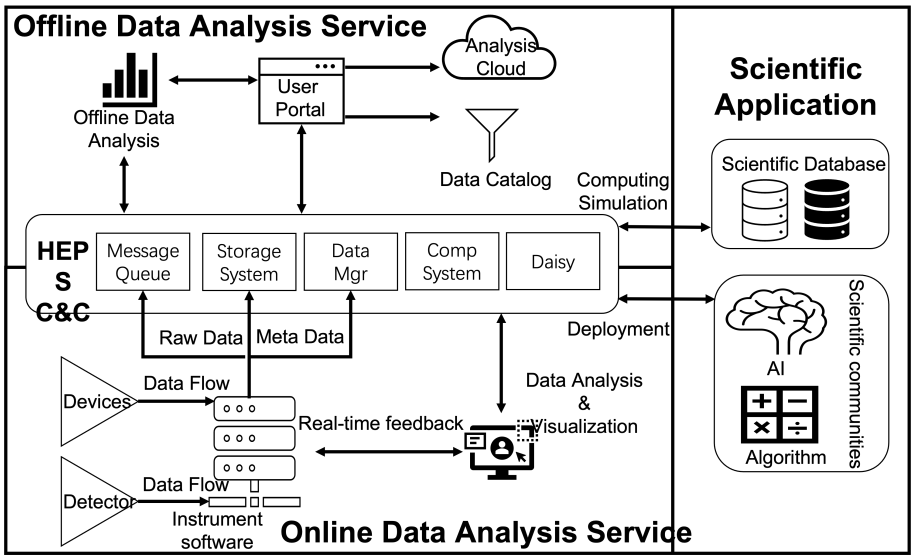

Figure 1: The integrated computing platform of the HEPS enable on-sit data analysis services

1. Introduction

Large scale research facilities, including radiation facilities and astronomical satellites, are becoming prevalent in the modern scientific landscape. One of these facilities' primary responsibilities is to make sure that users can process and analyse measurement data for publication. In order to allow for barrier-less access to those highly complex experiments, almost all beamlines in photon sources and space-based experiments require fast feedback capable of manipulating and visualising data online to offer convenience for the decision process of the experimental strategy.

And recently, the advent of beamlines at fourth-generation synchrotron sources and high resolution with high sample rate detector has made significant progress. On top of this, most synchrotron light sources have shifted to prolonged remote operation because of the outbreak of a global pandemic, with the need for remote access to the online instrumental system during the operation. Multi-messenger observations in astrophysics combining the gravitational waves, gamma-ray burst and the electromagnetic spectrum etc. require independently detected by multiple teams within a relatively short time period, which need extensive observing campaigns launched automatically across facilities. Those technologies push the demand for computing resources to the edge of current workstation capabilities. Another issue is the vast data volume produced by specific experiments makes it difficult for users to create local data copies. In this case, as shown in Figure.1, on-site data analysis services are necessary both during and after experiments.

The High Energy Photon Source (HEPS) [1] is the first high-performance fourth-generation synchrotron radiation light source with a diffraction-limited storage ring in China. It creates opportunities for researchers to perform multi-dimensional in-situ characterisation of complex structures and dynamic processes with a high beam energy of 6 GeV and ultra-low emittance of 0.06 nm*rad. Some state-of-the-art experimental techniques, such as phase-contrast tomography and ptychography approaches, will be deployed. The Hard X-ray Modulation Telescope (HXMT)

[2], named "Insight", is Chinas first X-ray astronomy satellite. The main scientific objectives include the discovery of new transient sources, monitoring gamma-ray bursts and gravitational wave electromagnetic counterparts, and observation of X-ray binaries. However, customised algorithms developed in the scientific communities are needed to use these techniques to the full extent.

However, it poses a critical problem of integrating this algorithmic development into a novel computing environment used in the experimental workflow [3]. The solution requires collaboration with the extramural research groups, in-house scientists and computational scientists. A unified software platform that provides an integrated working environment with generic functional modules and services is necessary to meet these requirements [4–6]. Scientists can work on their ideas, implement the prototype and check the results following some conventions without dealing with the technical details and the migration between different HPC environments. Thus, one of the vital considerations is integrating extensions into the software in a flexible and configurable way. Another challenge resides in the interactions between instrumental sub-systems, such as control system, data acquisition system, computing infrastructures, data management system [7], data storage system and so on, which can be quite complicated.

Based on an object-oriented plug-in architecture, the Daisy (Data Analysis Integrated Software System) [8] has been designed to address these challenges. The project aims to bridge the gap with an all-encompassing framework between sophisticated computing infrastructure and the users focusing on scientific issues. This article describes the main components of this software and the status of the project.

2. Architecture

As shown in Figure.2, the Daisy framework describes four layers and their interconnections as top down from the scientific applications layer that directly serves the end user, down to the computing resource layer referring to the computational node, distributed file systemmessage queue, etc.

The application layer is used by end-user software Web and native applications based on Jupyter notebook [9] and PyQt [10] integrate script editors, data visualising and processing widgets. . It employs jupyter message protocols that allow software to call data processing routine, receive, manipulate and present data to users. The built-in workflow builders allow users to build a workflow and parameterise it without having to code. The user interface layer prepares data for the application layer. Model-View-Controller (MVC) architecture is introduced for handling data objects, analysis workflow and presentation. The middleware layer has two main functions. One is scientific domain focusing on the algorithm and workflow (a sequence of algorithms). The other is running time domain that interacts with the computing environment and other instrumental sub-systems The computing resource layer is responsible for the persistent data and stream access and computing resource scheduling within the computing infrastructure.

The architecture supporting customised plug-in functions can easily access visualisation tools and the Python-based scientific computing ecosystem. The algorithm, the handler of data processing software code developed by the in-house and external scientific research groups, gets input data objects from the data store, passes them out to the external libraries, generates output data objects and sends them back to the data store. Python and C/C++ algorithms are compatible with Daisy

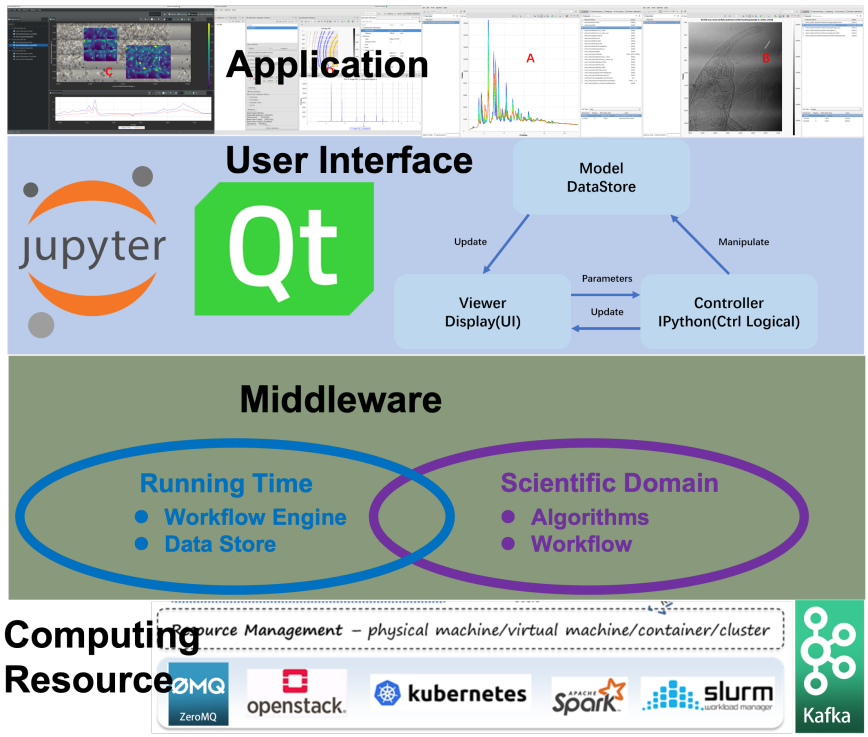

Figure 2: Daisy four layers model

based on Boost.python [11], and Fortran supported by F2PY [12] is on the way. The workflow, a sequence of algorithms, defined by scientists, is scheduled by the workflow engine. The primary data object supported in Daisy is NumPy numerical types. And tree structure complex data objects are registered in the data store. The data store is the data object container that passes on data objects between algorithms and workflow. It creates communication channels between algorithms and workflow, and it is responsible for shared memory and distributed memory management within and between compute nodes, while data object is being transferred, and released when workflow ends. Workflow engine schedule algorithms and sub workflow in sequential order, assign tasks dynamically to the distributed heterogeneous system considering data form, conditions, exceptions, etc. The in-process workflow engine is based on an in-house developed framework, SNiPER [13], which provides the function of calling dynamic link libraries, in-memory data object management and C++/Python binding. In the cloud based computing environment, the workflow engine for large-scale data processing employs Apache Spark [14] running over clusters managed by Kubernetes [15].

PyQt5 based Daisy Workbench has implemented for end-user data visualisation, manipulation and integrated development environment, whose default layout is inspired by the design of Mantid Workbench [6] and Spyder [16]. As shown in Figure.3.A, several parts of the workbench are presented, including data object toolbox, algorithm toolbox, ipython console, script editor, log and status windows. The algorithm objects and data objects represented in the algorithm toolbox and data object toolbox can be accessed by the whole application scope. Plot window and workflow builder window can pop up when needed. Specific scientific techniques can be implemented in the

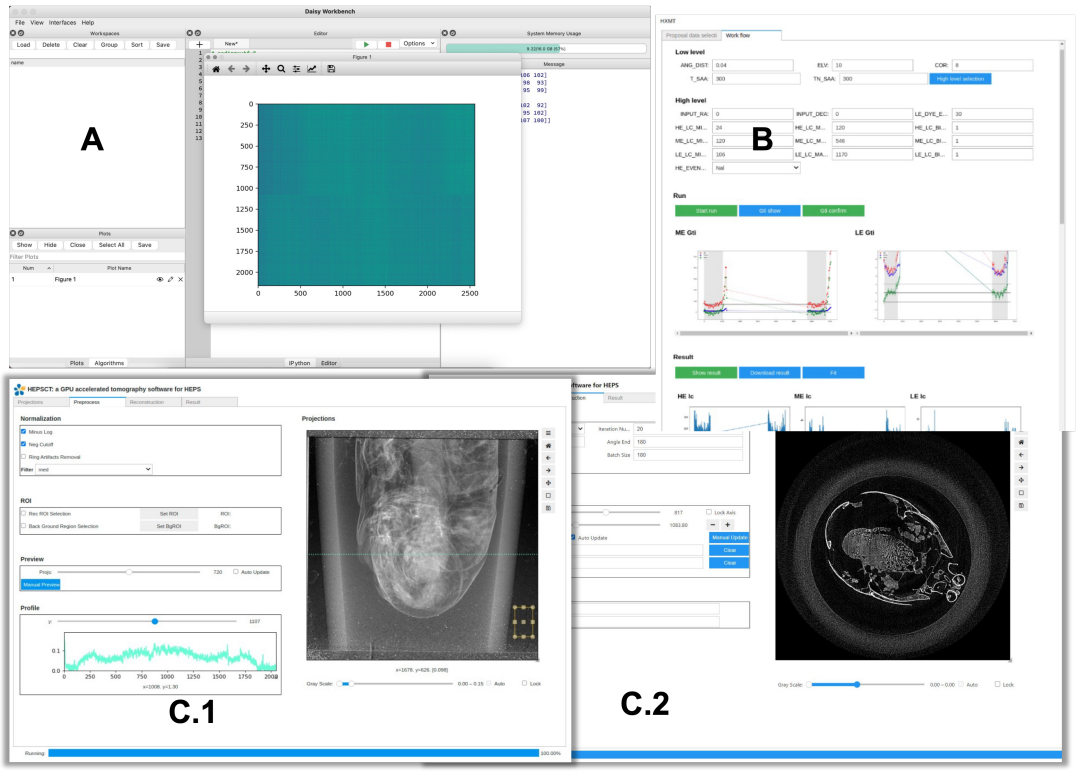

Figure 3: The scientific applications based on Daisy framework. A. the PyQt5 based Daisy Workbench providing an integrated development environment and data visualisation window; B. the HXMT Web data processing platform; C. one Daisy Cloud Application focusing on 3D tomographic reconstruction, C.1 the raw data and pre-processing image display tab, C.2 the reconstruction image display tab.

'Interface' widget, which employs a predefined workflow and custom GUI layout. Daisy Cloud App also has developed for interactive data visualisation in the Jupyter notebooks which can be accessed by remote users, where Daisy framework and ipywidget based GUI applications are packaged in the Docker container [17] image and launched by Kubernetes. Voila [18], a Jupyter server extension, is used to convert a embedding notebook widget into a standalone Web APP which provides an integrated and customised user experience like PyQt5 Applications.

3. Results and discussion of application examples

HXMT Web data processing platform allows users to query target data from open data platform based on specific rules through RESTful API. Users can set the corresponding algorithm operation parameters interactively on the web page, and click the execute button to start the preliminary analysis workflow which will call the HXMT data analysis software HXMTDAS. HXMT data analysis is divided into three main processes: HE, ME and LE, corresponding to three energy segments of high energy (20-250 keV), medium energy (5-30 keV) and low energy (1-15 keV) respectively. Each main process is a workflow which can be divided into several steps. Visualisation results can be displayed visually on Jupyter after the final analysis completed, as shown in Figure.3.B.

Integrating a in-house developed 3D tomographic reconstruction package, cumopy, into the framework, Daisy provides a remote accessing client employing the Jupyter message protocol and data visualisation based on Jupyter Widget. This application can conduct real-time computing to remove the background, control the white balance, carry motion correction, detect the position of rotation axis, etc. After the parameters are set, reconstruction algorithms are executed on GPUs, as shown in Figure.3.C.

Those applications allow us to demonstrate the benefit of using Daisy plug-ins architecture for online analysis, where users can access scientific workflow and analysis tools through web browsers. Large volumes of data can be retrieved and processed using Daisy framework for data analysis and visalisation in the HXMT Web data processing platform, which reduces the pressures of the data deluge challenge. And container-based Daisy applications running on heterogeneous parallel computing, such as the 3D tomographic reconstruction application, can accelerate the speed of compute-intensive tasks. In this case, researchers and programmers are able to access some expensive and scarce computing resources that only national laboratories, government agencies, and some universities can afford. And users can also benefit by the use of special programming techniques to exploit the speed, such as special-purposed FORTRAN compilers, PVM, MPI, and OpenMP. In the development procedures, to tackle the critical challenges in areas ranging from developing new materials to unlocking the basic secrets of the universe, world-class scientists and engineers from different organizations should work together to solve problems using the most sophisticated scientific and computing facilities, so an open and inclusive innovation ecosystem is necessary.

4. Outlook

Several features should be considered in the next stage of development. The cutting-edge artificial intelligence algorithms combining the scientific datasets, databases and models require the full power of heterogeneous parallel computing system co-located with the data management and storage system, which is developing for the extensions of Daisy including dashboard, content management and declarative interactive widgets. Moreover, HDF5 IO in parallel computer system is a challenge considering the huge volume of single dataset. So a distributed in-memory data access system is necessary for Daisy framework, and should be optimised to ensure big data orchestrated across frameworks, clouds, and storage systems of HEPS online computing system. Last but not the least, the Jupyter notebook should be accessed by world-wide users, which can connect to Apache Spark and create a new Spark application, whose work notes run on the Docker containers managed by Kubernetes.

References

[1] Yi Jiao, Zhe Duan, Yuanyuan Gao, et al., Physics Procedia 84, 40 46 (2016)

[2] Zheng, S.J., et al. (Insight-HXMT team), Spacecraft Engineering, 27, 162 (2018)

[3] Incardona, M.-F., Bourenkov, G. P., Levik, K., Pieritz, R. A., Popov, A. N. & Svensson, O., J. Synchrotron Radiat. 16, 872879 (2009)

[4] Basham, M., Filik, J., Wharmby, M. T., et al., J. Synchrotron Radiat. 22, 853858 (2015)

[5] Tian, H. L., Zhang, J. R., Yan, L. L., et al., Nucl. Instrum. MethodsPhys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip. 834, 2429 (2016)

[6] O. Arnold, J. Bilheux, J. Borreguero, A. Buts, S. Campbell, et al, Nucl. Instrum. MethodsPhys. Res. Sect. A: Accel. Spectrom. Detect.Assoc.Equip.764, 156166 (2014)

[7] Hao Hu, Fazhi Qi, Hongmei Zhang, Haolai Tian and Qi Luo, J. Synchrotron Rad. 28, 169175 (2021)

[8] Yu Hu, Ling Li, Haolai Tian, et al., EPJ Web of Conferences 251, 04020 (2021)

[9] Project Jupyter, https://jupyter.org (2022), accessed: 2022-04

[10] Qt for Python, https://doc.qt.io/qtforpython/ (2022), accessed: 2022-04

[11] Boost C++ libraries, https://www.boost.org/, accessed: 2022-04

[12] F2PY user guide and reference manual, https://numpy.org/doc/stable/f2py/, accessed: 2022-04

[13] J. H. Zou, et al., 2015 J. Phys.: Conf. Ser. 664, 072053 (2015)

[14] Apache Spark, https://spark.apache.org/ (2022), accessed: 2022-04

[15] Kubernetes, https://kubernetes.io/ (2022), accessed: 2022-04

[16] Spyder IDE, https://www.spyder-ide.org (2022), accessed: 2022-04

[17] Docker, https://www.docker.com/ (2022), accessed: 2022-04

[18] Using Voila, https://voila.readthedocs.io/en/stable/using.html (2022), accessed: 2022-04