Just-in-time workflow management for DUNE

Andrew McNab[1,*] and *Christopher* Brew[2,**]
On Behalf of the DUNE Collaboration

[1]Department of Physics and Astronomy, Schuster Laboratory, University of Manchester, Oxford Road, Manchester, United Kingdom
[2]Department of Particle Physics, STFC Rutherford Appleton Laboratory, Didcot, United Kingdom

Abstract. We describe the justIN workflow management system developed by DUNE to address its unique requirements and constraints. The DUNE experiment will start running in 2029, recording 30 PB/year of raw data from the detectors, with typical readouts at the scale of gigabytes, but with regular supernova candidate readouts of several hundred terabytes. DUNE benefits from the rich heritage of neutrino experiments at Fermilab, including the use of the SAM system to manage data, metadata, and processing campaigns. Due to the increase in scale required for DUNE, SAM's metadata database has been replaced by a new system, MetaCat, and its data management role is now taken up by Rucio. A new workflow system, justIN, has been developed since 2021 to replace the remaining functionality of SAM, and to tie together MetaCat, Rucio, and the GlideInWMS job management system to allow data processing campaigns involving hundreds of thousands of files and jobs using CPU and storage on four continents. Crucial to the design of justIN is an evolution of SAM's just-in-time philosophy, in which running jobs ask the central service for the optimal file to process next given their location. This model and the SAM interface are directly supported by the liquid argon data processing application used by DUNE, and justIN's design allows us to continue to support both. justIN goes a step further by assigning the workflows themselves to running GlideInWMS pilot jobs based on the locations of unprocessed files in active workflows at that time. This architecture allows the system to rapidly respond to problems such as downtimes, and to the sudden appearance of higher priority tasks such as processing supernova candidates.

1 Introduction

DUNE is a future liquid argon TPC neutrino oscillation and astrophysical neutrino experiment that will take data at a rate of 30 PB/year. The DUNE computing system is described in its conceptual design report [1] and has evolved from the heritage of neutrino and collider experiments based at Fermilab. To achieve the increase in scale required by DUNE it has been necessary to modernise the framework that was largely built around SAM ("Serial Access to Metadata") [2], dating back to the Tevatron collider experiments. Where possible, DUNE is reusing tools developed for other High Energy Physics experiments and for Worldwide LHC Computing Grid.

*e-mail: andrew.mcnab@cern.ch
**e-mail: chris.brew@stfc.ac.uk

Before the start of the work described in this paper, DUNE chose to adopt Rucio [3] as the basis for its data management strategy, and worked with Fermilab to develop a new metadata database, MetaCat [4]. Together these systems allow for the discovery of which files match particular criteria and where the replicas of those files are located.

The arrival of large amounts of storage at DUNE sites other than Fermilab has also exposed limitations with the SAM-based framework when matching work to sites. To address this, the UK has led the DUNE workflow system design and development, which aims to run work at or near sites where the relevant data is stored.

2 justIN workflow system

One of the strengths of the SAM model was that the choice of which file a job should process next was made at the last minute, based on which files were still unprocessed. Experiments were started in 2021 to extend this model to the matching of workflows to available job slots. This led to a complete just-in-time workflow system named justIN [5], in which the decision about which workflow to run in a job slot is based on a complex database query over all the running workflows, the unprocessed files of those workflows, and the distances from the execution site in question to replicas of those files. Once the job starts, the choice of which input files to process is also delayed until the last moment, again based on the relative locations of unprocessed files for that workflow.

This contrasts with systems such as DIRAC [6] where "late binding" is typically done at the level of payload jobs with a fixed list of input files, which are matched to suitable slots within pilot jobs at sites. If no suitable site is available with access to local storage that can satisfy all of the input requirements then either the job cannot run or reading across the network must be tolerated. With justIN, these decisions are taken during the execution of the job, which allows the system to react to unexpected problems with sites and storage and make optimal choices based on the current situation.

3 Services and Agents

The justIN system consists of several centrally operated services and agents and a command line client provided to users through the same channels as the rest of the DUNE offline software. All major components are written in Python3, with some ancillary Unix shell scripts.

The central services and agents are:

- **justIN Database** A MySQL/MariaDB database holding cached information about files, replicas, sites, storages, and jobs.
- **Info Collector agent** Gathers information about sites from the Open Science Grid pilot factories and about storages from the DUNE Rucio service.
- **Finder agent** Queries MetaCat for the list of files needed by a workflow; queries Rucio to discover the available replicas of lists of files; checks HTCondor for the state of justIN jobs and removes stalled jobs.
- **Job factory agent** Submits justIN wrapper jobs to HTCondor.
- **Archiver agent** Saves summaries of completed workflows and removes detailed records which are no longer needed.
- **Allocator service** Provides REST APIs to justIN wrapper jobs and user jobscripts.
- **User Interface service** Provides REST APIs to the justin command line tool to allow for the creation and management of workflows.

- **Dashboard service** Provides the justIN web dashboard to monitor and manage workflows.

The Job Factory agent submits justIN wrapper jobs to the DUNE HTCondor [7] Global Pool, with parameters specifying the workflow's memory and processor requirements, and the reference number of the workflow. For operational reasons and to ease development, dedicated HTCondor schedulers for justIN jobs are installed on the STFC cloud service at the Rutherford Appleton Laboratory and are members of the DUNE HTCondor Global Pool.

GlideInWMS [8] pilot jobs are submitted by the OSG pilot factory, and a custom DUNE script within the pilot job contacts the justIN Allocator service to obtain the correct parameters for this job slot to advertise to the DUNE Global Pool of waiting jobs. This procedure matches jobs from workflows based on the proximity of *unprocessed* files. Since the final matching is still done by HTCondor's ClassAd system, HTCondor's accounting system is able to prioritize justIN vs non-justIN jobs, and between different justIN users based on their recent usage history.

Figure 1 shows how the above processes work together to direct jobs to appropriate sites based on replica locations.

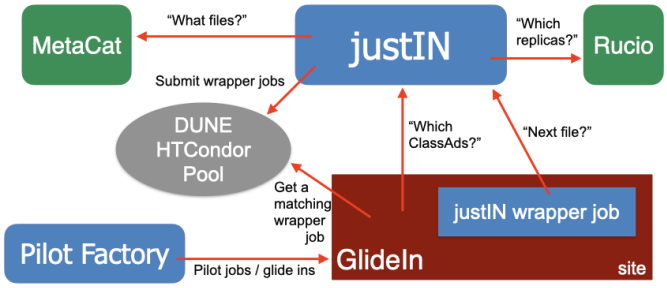

Figure 1. justIN, Rucio, MetaCat, and sites

4 justIN wrapper jobs

The justIN wrapper jobs themselves execute the Unix shell jobscripts created by the user, in an Apptainer [9] container to enforce privilege separation. Users' X.509 and VOMS credentials are not used, but a low privilege DUNE X.509 proxy is made available to jobscripts which can only read files from storage. The wrapper job fetches information about the jobscript to be executed from the justIN allocator service and sends updates back to the service, both in the form of JSON [10] dictionaries.

Users' jobscripts request the Rucio identifiers or full URLs of suitable files to process from justIN's central services. A "next file" REST API is provided and a wrapper script, justin-get-file, is also available to further simplify this for users. The user's jobscript is required to maintain a list of successfully processed input files so that files which were not successfully processed and reported as processed can be retried in another job. Output files are discovered by the justIN job wrapper using file name patterns declared when the workflow was originally defined.

Once the jobscript is finished, the wrapper job uses a higher level X.509 proxy or token to write the output files to storage and register them in Rucio and MetaCat. This model allows DUNE to enforce rules about how the storage is used by different groups of users without the storage systems at sites needing to enforce these distinctions. Successes or failures of file uploads are logged as events to the justIN central services. If the upload of the output files fails, the input files are allocated to another job to be processed again. A limit on the number of retries can adjusted to prevent indefinite attempts at operations which cannot currently succeed for some reason.

The structure of the wrapper jobs is show in figure 2.

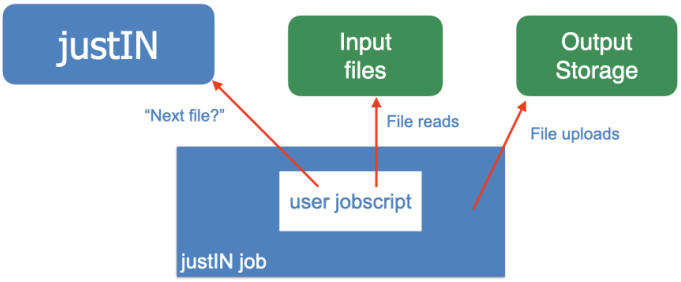

Figure 2. justIN wrapper job structure

5 Authentication and Authorization

Users authenticate to justIN via the justIN web dashboard whether using the dashboard itself or working at the command line.

Authentication is currently via CILogon [11] and the Fermilab Single Sign-on service, and other eduGAIN compatible sign-on systems will be available in the future. Figure 3 shows the justIN web dashboard login page. CILogon releases the user's DUNE SciTokens / WLCG Tokens to justIN, which justIN is able to use on the user's behalf and to parse to discover the user's group memberships.

At the command line, the justin command invites the user to visit a one time URL on the dashboard website to authorise the command line environment to act as the user and the web based authentication proceeds as normal. The simplicity of this approach allows for a self-contained Python3 client script which can be installed by simply copying it in place on Linux or macOS.

6 justIN in use

Following its successful validation during a global readiness exercise, Data Challenge 24 (DC24), in early 2024, justIN has been the basis for all DUNE central productions and is being rolled out for general users and working groups.

Figure 4 shows the cumulative number of files processed by justIN during 2024. The short DC24 period is visible early in the year with a steep increase in processed files, a period

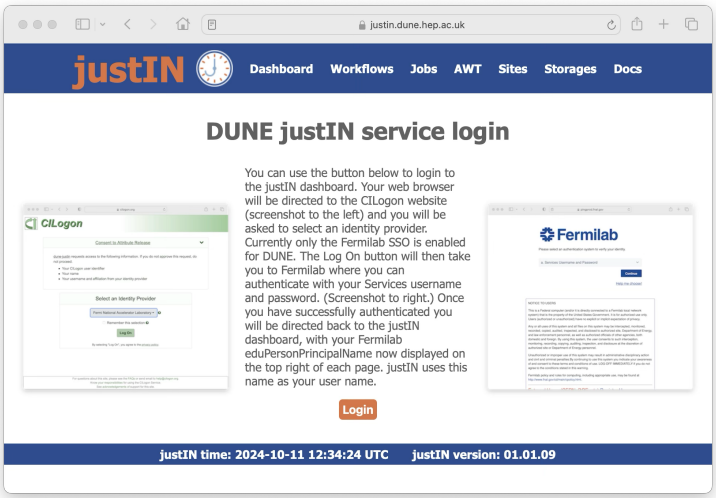

Figure 3. Login page on the justIN web dashboard

of development and low level testing of justIN version 1.0, and then justIN becoming the basis for all centrally managed DUNE production workflows. In the second half of the year, justIN became the basis for "keep-up" processing of data being recorded by protoDUNE HD at CERN as it became available to the offline computing system. Almost all of these jobs executed simple jobscripts in which only one file was processed per job and took of order one hour to execute.

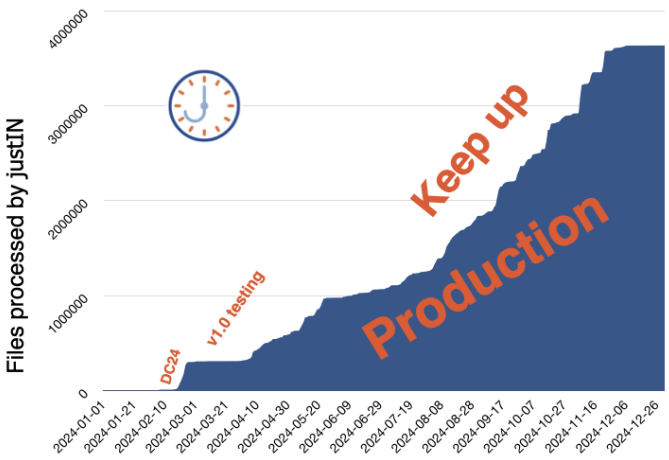

Figure 4. DUNE file processing with justIN in 2024

References

[1] DUNE Collaboration. DUNE Offline Computing Conceptual Design Report. 2022 https://arxiv.org/abs/2210.15665

[2] R. A. Illingworth, A data handling system for modern and future Fermilab experiments. J. Phys.: Conf. Ser. **513** 032045 (2014)

[3] M. Barisits, T. Beermann, F. Berghaus, et al., Rucio: Scientific Data Management. Comput Softw Big Sci **3:11** (2019)

[4] I. Mandrichenko, MetaCat - metadata catalog for data management systems. EPJ Web of Conferences **251** 02048 (2021)

[5] justIN https://justin.dune.hep.ac.uk/

[6] DIRAC, http://dirac.readthedocs.io/

[7] D. Thain, T. Tannenbaum, M. Livny, Distributed Computing in Practice: The Condor Experience. Concurrency - Practice and Experience **17** 323 (2005)

[8] GlideInWMS https://glideinwms.fnal.gov/

[9] Apptainer https://apptainer.org

[10] T. Bray et. al., The JavaScript Object Notation (JSON) Data Interchange Format, Internet Engineering Task Force, RFC8259 (2017), https://www.rfc-editor.org/rfc/rfc8259

[11] J. Basney, et al., CILogon: Enabling Federated Identity and Access Management for Scientific Collaborations. Proceedings of the International Symposium on Grids and Clouds (ISGC), PoS (ISGC2019) **351** 031 (2019)