EPJ Web of Conferences **108**, 01001 (2016)
DOI: 10.1051/epjconf/201610801001


# PanDA: Exascale Federation of Resources for the ATLAS Experiment at the LHC

Fernando Barreiro Megino[1,a], Jose Caballero Bejar[2], Kaushik De[1], John Hover[2], Alexei Klimentov[2], Tadashi Maeno[2], Paul Nilsson[2], Danila Oleynik[1,3], Siarhei Padolski[2], Sergey Panitkin[2], Artem Petrosyan[3], and Torre Wenaus[2] on behalf of the ATLAS collaboration

[1] *University of Texas at Arlington, 502 Yates Street, Arlington, TX 76019-0059, USA*
[2] *Brookhaven National Laboratory, Upton, Long Island, New York 11973, USA*
[3] *Joint Institute for Nuclear Research, Joliot-Curie 6, 141980 Dubna, Russia*

**Abstract.** After a scheduled maintenance and upgrade period, the world's largest and most powerful machine – the Large Hadron Collider(LHC) – is about to enter its second run at unprecedented energies. In order to exploit the scientific potential of the machine, the experiments at the LHC face computational challenges with enormous data volumes that need to be analysed by thousand of physics users and compared to simulated data. Given diverse funding constraints, the computational resources for the LHC have been deployed in a worldwide mesh of data centres, connected to each other through Grid technologies.

The PanDA (Production and Distributed Analysis) system was developed in 2005 for the ATLAS experiment on top of this heterogeneous infrastructure to seamlessly integrate the computational resources and give the users the feeling of a unique system. Since its origins, PanDA has evolved together with upcoming computing paradigms in and outside HEP, such as changes in the networking model, Cloud Computing and HPC. It is currently running steadily up to 200 thousand simultaneous cores (limited by the available resources for ATLAS), up to two million aggregated jobs per day and processes over an exabyte of data per year. The success of PanDA in ATLAS is triggering the widespread adoption and testing by other experiments. In this contribution we will give an overview of the PanDA components and focus on the new features and upcoming challenges that are relevant to the next decade of distributed computing workload management using PanDA.

## 1 Introduction

The PanDA – Production and Distributed Analysis – system was developed for the ATLAS experiment [1] to handle the full workload management of the experiment on the distributed Worldwide LHC Computing Grid (WLCG) [2] resources. Since its inception in the summer of 2005, the PanDA system has grown to accommodate new requirements from the ATLAS experiment and also integrate upcoming computing paradigms. The core and main ideas have not changed much:

[a] e-mail: barreiro@uta.edu


Article available at http://www.epj-conferences.org or http://dx.doi.org/10.1051/epjconf/201610801001

- a well tuned central service that is aware of all available compute resources to abstract the complexity of the underlying Worldwide LHC Computing Grid (WLCG).
- a late binding pilot model, where jobs are assigned to a site at the last moment.

However, in order to keep up to date and benefit from the fast-paced evolution of networks and computing paradigms, namely HPC and Cloud Computing, PanDA has put effort to adapt these ideas and concepts into its core and thus offer a better usage of all available resources. As a result, PanDA has served the ATLAS experiment during Run1 without breathing heavily and is already demonstrating its capabilities during Run2 by managing up to 200k simultaneous jobs, pertaining to a user community of around 3k analysis users and production power users.

## 2 Overview

Figure 1 shows the high level overview of the PanDA system and how it interacts with the WLCG.

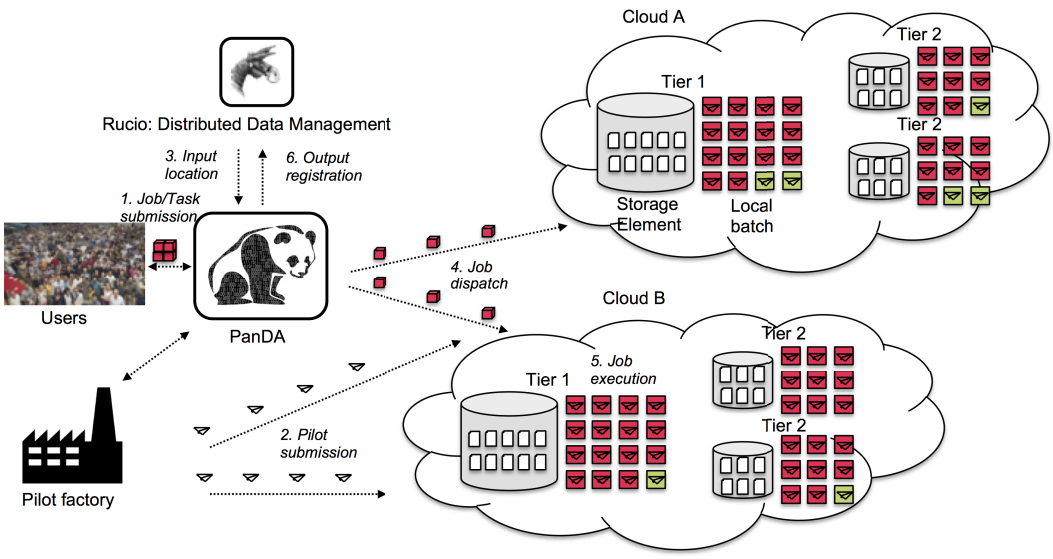


**Figure 1.** High level diagram of the PanDA system

1. Job submission is done either through a thin Python client for analysis users, or through an advanced web GUI (Prodsys2) for production users, who can define complex tasks[1].

2. The Pilot Factory [3] submits pilot jobs to the sites' compute elements, being compatible with the usual batch flavours (e.g. Condor-G, glite). The number of pilots sent to a site depends on the site configuration, the number of queued jobs in PanDA for the site and also the status of the batch system.

3. The PanDA Pilot [4] is a generic wrapper that starts on the worker node and requests the real payload from the PanDA Server [5]. As seen in Figure 2, the pilot is responsible for:

[1]Tasks are sets of jobs with defined relationships and order between them.

(a) checking the local resources in the worker node, such as disk space and available memory;

(b) asking the PanDA job dispatcher for a suitable job and downloading the matching job;

(c) setting up the execution environment according to job specifications;

(d) staging the input files from the local grid storage element. The pilot supports many different stage-in methods, which can be configured for the site;

(e) running the job, monitoring its progress and sending heartbeat messages to the PanDA Server. Tests to run the job with gLExec [6] identity switching have been completed successfully, but it is not mandatory;

(f) staging out the output files to the local storage element once the job has completed and communicating the final job status back to the PanDA Server;

(g) cleaning up the worker node to leave it in the original state.

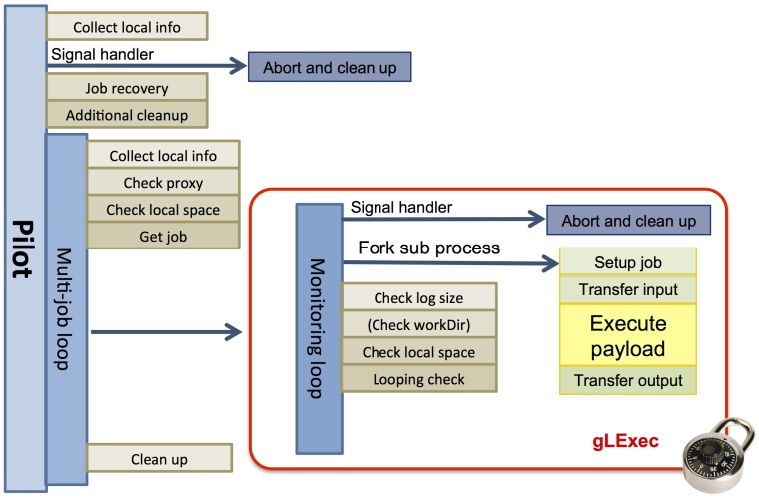


**Figure 2.** High level diagram of the PanDA Pilot

4. The PanDA Server is the core of the system. It consists of a set of load balanced web servers, and a highly tuned database backend. In the ATLAS setup, there are 8 web servers in order to cope with the load of O(100 k) pilots and few thousand users hitting the server simultaneously. The PanDA Server queues the tasks and jobs, and queries Rucio [7], the ATLAS Distributed Data Management (DDM) system, for the input dataset (file collections) locations in the WLCG. Based on this information, the PanDA Server performs task brokerage to a cloud[2] and job brokering to the sites inside the cloud. The jobs will execute in several sites of the cloud and PanDA will aggregate the output in the cloud's Tier1. Further concepts the PanDA Server takes care of:

[2]A cloud in the WLCG context is a regional grouping of Grid compute centres, with one Tier1 (the main site with tape archival) and multiple Tier2s. Tasks are executed inside a cloud, so that the output is aggregated in the Tier1. This concept is unrelated from the Cloud Computing context.

- calculating the job priorities based on various fair share mechanisms. When the pilot asks for a job from the server, the server returns the single job having the highest priority in the requested queue.
- users have a share of $x$ CPU-hours per 24h. Once the user exceeds the limit, additional jobs are de-prioritized.
- priority boosts are available for users and groups at particular sites or regions. In this way sites can allocate resources to users beyond that pledged to ATLAS as a whole.
- preventing resource starvation, by increasing priorities for jobs while they wait in the queue.
- reducing the job priority of failed jobs to delay them slightly.
- re-brokerage capability. When a user job is submitted to PanDA, the server decides in some conditions to trigger additional replicas of the job's input dataset. This feature, known as PanDA Dynamic Data Placement (PD2P), enables a global placement of data based on the popularity of input datasets. Since PD2P results in a constantly-changing state of input dataset locations, PanDA periodically re-brokers the waiting jobs. In this way, jobs that have been unfortunately assigned to a busy site, can be cancelled at that site and moved to a new site once the dataset becomes available and the queue is less busy at the new site.

5. PanDA Monitor [8] (see figure 3) is the monitoring interface for users and operators to follow the status of tasks, jobs, sites, users, error codes and other parameters. It also allows to download the log files (pilot log and batch stderr&stdout) of finished jobs. The PanDA Monitor is built on Django (a python web framework) and shares the database backend with the PanDA Server.

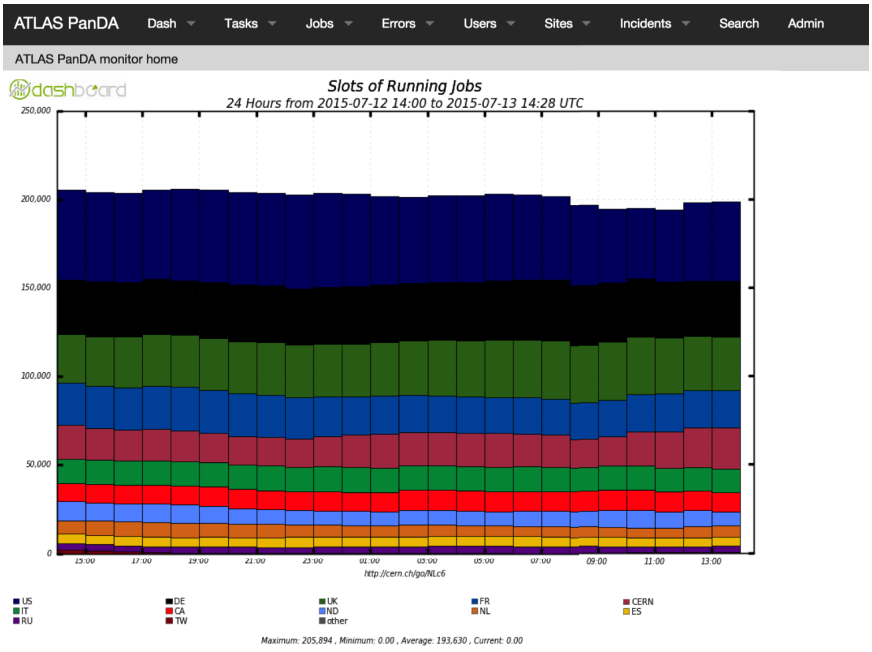


**Figure 3.** PanDA Monitor

## 3 PanDA and opportunistic resources

Predictions for the next years show that the ATLAS allocated resources might not cover the increasing computing necessities to fulfill the physics program. It is for this reason that the ATLAS Distributed Computing community is creatively searching for additional opportunistic resources from external sources. As a result, ATLAS has found several agreements and a major source of additional resources in the upcoming computing paradigms of HPC and Cloud Computing:

- commercial cloud providers, who are interested in advertising themselves to the research community and explore common paths, occasionally offer grants for temporary allocations.
- diverse research clouds collaborating with High Energy Physics through different research programs.
- diverse research supercomputers that are interested in finding heavy computational challenges for their installations.

The above implies a key message, the WLCG will stay and we are not trying to replace it by the Cloud or HPC, but we are rather integrating all together. The next sections will describe the different integration architectures that we are successfully pursuing to slowly, but steadily (see figure 4) offload onto opportunistic resources.

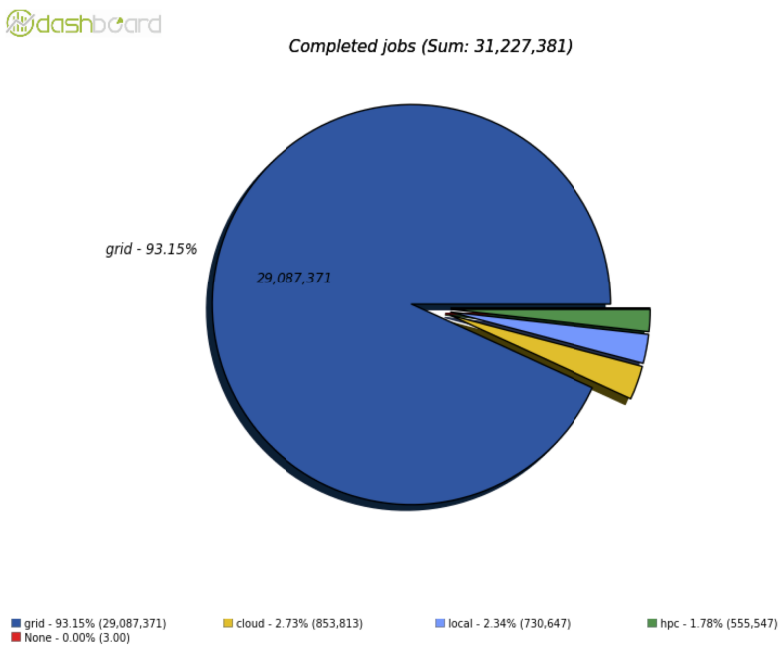


**Figure 4.** Jobs by resource type. Period 6 June to 5 July 2015

### 3.1 PanDA and Cloud Computing

When we talk about Cloud Computing [9], we are mostly referring to Infrastructure as a Service. A user is able to boot Virtual Machines (VMs) at a remote compute centre through an API. The

VM will be based on a certain image, in our case normally Scientific Linux 6, although compatible flavours have been used as well. The VMs can be booted manually if needed, or preferably in an automated way by the Pilot Factory, when a compatible API is available [3]. Note here the current lack of standardisation of cloud technologies, since it is a relatively new technology and competitors are pushing in different directions. These VMs will join a Condor pool [10] and represent a new site in the PanDA system, or can be added to an existing, distributed site. Since the High Energy Physics community still has not decided on a durable model to use cloud storage (object stores), currently all the VMs will read their input from and write their output to a Grid storage element. Since the network bandwidth between the storage element and the Cloud Computing worker nodes is not guaranteed, we are interested in running jobs with low I/O in the cloud. Here Geant 4 [11] Monte Carlo simulation jobs come to the rescue: the jobs require low I/O (1MB per event), but are highly compute intensive (300s per event), making them ideal to be offloaded to the cloud.

Another point to note is that all the experiment software dependencies are imported through CVMFS [12], a software distribution system that can be mounted on the Virtual Machines. Hereby the size of the Virtual Machine images is kept small and independent from the ATLAS software release cycles. Figure 5 summarises the setup.

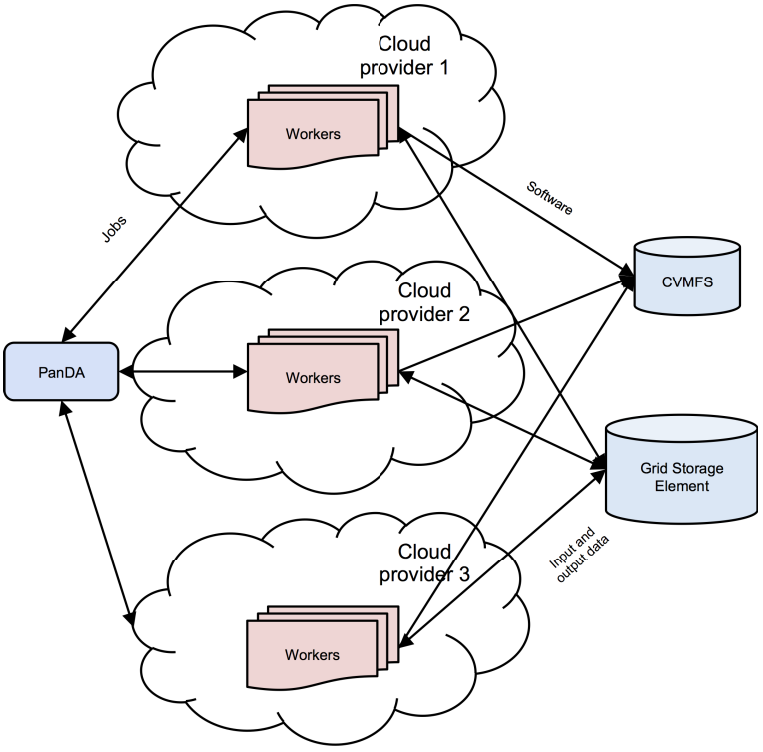


**Figure 5.** Cloud Computing integration schema

At this point, tens of cloud providers have been integrated with the PanDA system, both for short trial periods and for stable operation over sustained periods of time.

### 3.2 PanDA and HPC

HPC supercomputers are extremely powerful facilities, specially built to process massively parallel applications that occupy hundreds or thousands of cores simultaneously. A grosso modo, we estimate that the five top HPC supercomputers in the world have an equivalent computational power as the full ATLAS Grid. When a HPC supercomputer is considered full, it probably does not mean that every single core is occupied, but that it currently can not allocate another massive task. HPC facilities are interested to run at full capacity and give generous allocations (e.g. 10 M CPU hours on Titan for 2014–15), where ATLAS can reserve a dedicated slice of the supercomputer. On the other hand, HEP applications (such as Geant or ROOT) can effectively use a single core and could be used to backfill the unused cycles, which at the end of the year can add up to a non despicable slice of resources. This can be seen as adding sand to a jar full of stones and filling all the spaces.

HPCs require to be integrated on a case by case basis, because of their features and differences:

- unique architectures and hardware.
- specialised operating systems, i.e. we can not import our favourite Linux distribution.
- "weak" worker nodes and limited memory per WN.
- outside connection usually only through interactive node.
- code cross-compilation, since the processor might not be x86 architecture.
- unique internal job submission systems: PBS, TORQUE, SLURM etc.
- parallel file systems shared between nodes.
- special services to help merging output files and transfer them externally.
- unique security environment and policies.

One line of work is on extending the pilot [13] (see figure 6) to run on the interactive node and take care of spawning workload to the worker nodes. This implementation leverages on the SAGA-Python framework (Simple API for Grid Applications [14]) by the RADICAL group of Rutgers University to interface the internal batch system. The job outputs are written to the shared filesystem and then transferred asynchronously to the final storage endpoint in the Grid.

### 3.3 The Event Service

Opportunistic resources have the drawback that the worker nodes might have to be preempted without notice, e.g.:

- Amazon Spot Market, a bidding service for idle compute resources – VMs can be rented at a fraction of the on-demand price. ATLAS can participate in the bidding and use worker nodes only during the time where the price is below a certain threshold.
- HPC, where backfilled resources might want to be allocated to a demanding task.
- BOINC volunteer computing, where volunteer citizens offer their personal computers and laptops to run ATLAS jobs in the background.

The job length in these resources is critical, since the job could be interrupted before completing and hence all the work done would be lost. This motivates another important line of work: the ATLAS Event Service [15] (figure 7), a fine grained approach to event processing. Events are the lowest significant units of data in High Energy Physics, representing a collision captured in the detector. Lowering the granularity of jobs to event level processing allows an almost continuous event streaming

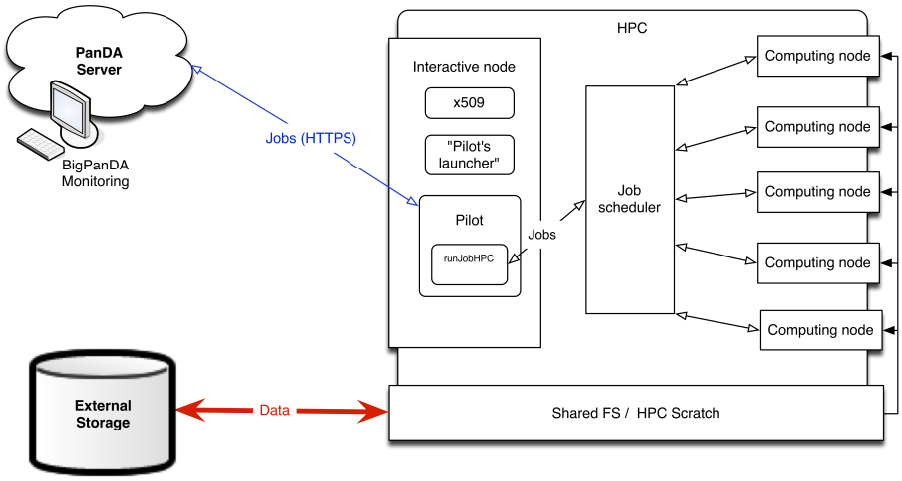


**Figure 6.** HPC integration in PanDA

to the worker nodes that write their output quickly and hence represent a negligible loss if the worker vanishes. The Event Service relies on the dynamic job definition of the PanDA JEDI extension. JEDI is able to partition jobs dynamically based on the site to execute on (see figure 8) [16]. Future work for the Event Service targets to fully optimising the data delivery by refining the data staging to the worker nodes through an event streaming service.

## 4 Network as a resource in PanDA

Network capacities have multiplied in the order of a thousand in the last 15 years. Initiatives such as the LHCOPN [17], a private optical network connecting the WLCG Tier1 sites at multiples of 10Gbps, and LHCONE, the WLCG worldwide research network community establishing nominal 10Gbps Tier2 bandwidth, are ensuring excellent network connectivity beyond national boundaries. Old models such as MONARC, which only expected good connectivity inside the national research networks, make no longer sense and need to be evolved. The ATLAS' computing model is therefore progressively moving away [18, 19] from a strict hierarchic data flow inside clouds (figure 9) to a direct mesh (figure 10), where data transfer sources are dynamically decided based on available network metrics.

There are currently three sources of network information in ATLAS:

- FTS: the WLCG File Transfer Service, used by the ATLAS Distributed Data Management system (Rucio). Statistics are collected for each file transferred from site A to site B. A cron-like tool, known as the Sonar, ensures transfers across the whole mesh and ensures base measurements for each source destination pair.
- perfSonar: perfSonar is a service that collects precise low-level network metrics between participating sites.
- FAX measurements: transfer statistics covering transfers across the federated XRootD sites (explained in the next section).

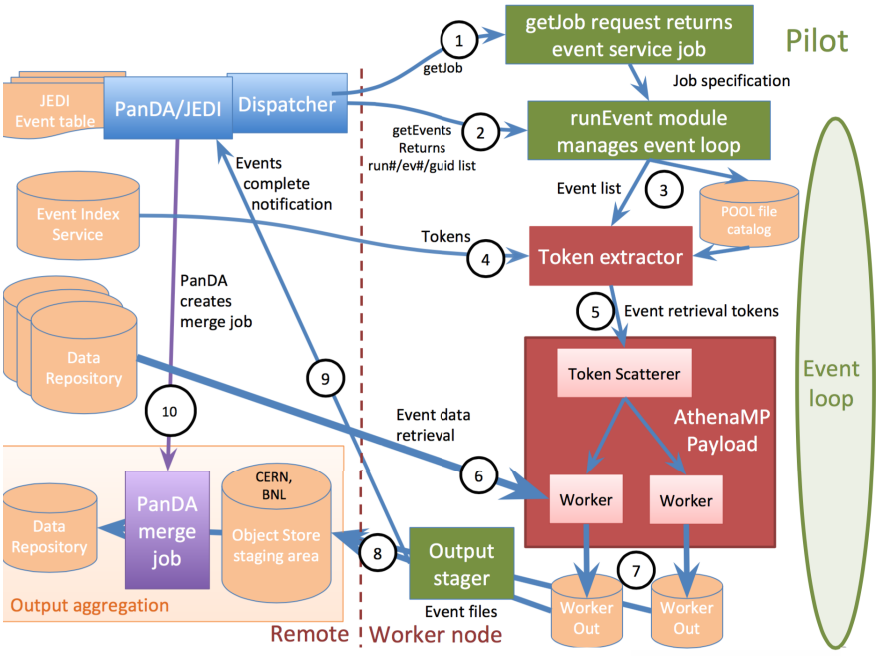


**Figure 7.** Event Service Schematic

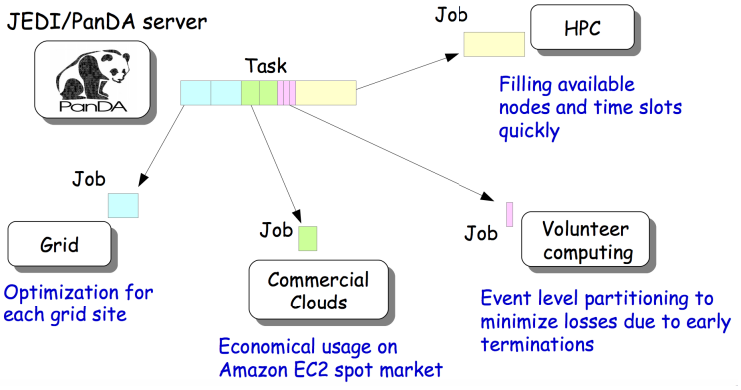


**Figure 8.** JEDI dynamic job definition

The three sources of network statistics are consolidated in the Site Status Board, and the latest values are also exported into the PanDA database for its use in network based brokerage. Jobs inside a task have to be brokered in the same cloud, in order to collect and archive the output in the Tier1. Currently the clouds are defined manually in an extended way: they include (see figure 11) both local and foreign sites. However there are many more foreign sites (Foreign Calculated) that pass diverse thresholds of internal performance and also network connectivity to the Tier1 (files are served at over

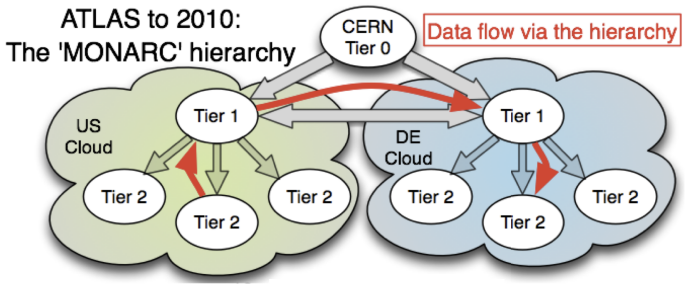


**Figure 9.** Data flows based on MONARC model

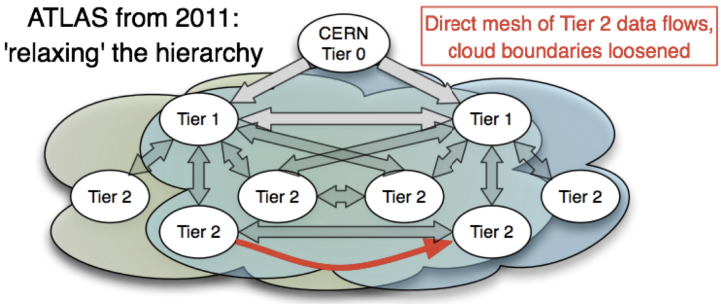


**Figure 10.** Data flows in mesh model

5 MBps). PanDA is currently moving towards a dynamic cloud management, where the foreign sites will be calculated and updated periodically. Ultimately, the desire is to move to a full mesh with no boundaries, since this would simplify operations and brokerage.

## 5 PanDA beyond HEP

PanDA has successfully served the ATLAS experiment at the exascale, processing over 1 Exabyte per year and showing no signs of scalability ceilings. Other experiments are therefore interested in joining the PanDA community. Currently AMS, ALICE, COMPASS, LSST and NICA are some of the experiments that are either testing or using PanDA. For this reason, the PanDa team is leading the "packaging" activity, with mandate to reduce the adoption cost for new joiners. The lines of work in this activity are:

- Refactoring: sanitise and generalise code fragments, where needed.
- Deployment: automate and simplify installation. Provide validation steps.
- Documentation: improve the documentation and share the knowledge.

and work with other experiments to understand their needs.

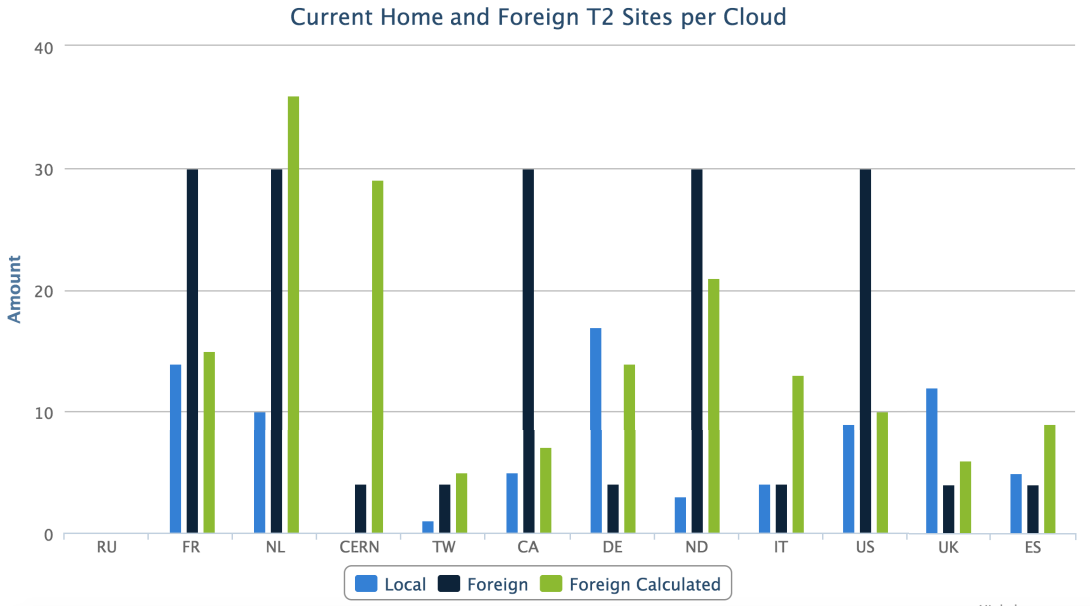


**Figure 11.** Dynamic cloud potential

## 6 Conclusions

ATLAS fully relies on PanDA to handle the experiment's analysis and production workloads, and to date PanDA has successfully delivered without showing scalability issues. To meet the future LHC program and its computing demands, PanDA must be able to utilise available resources as fully and efficiently as possible. The JEDI extension allows dynamic job generation with a granularity suited to Grid resources, as well as opportunistic resources as volunteer computing, HPC and Cloud Computing. In this context, the Event Service is a major game changer, allowing to backfill and exploit opportunistic resources at a fine grained level and presenting a despicable loss in the case a worker has to be preempted without notice. Thanks to these developments and dedicated effort, we have proven we are ready to embrace the diverse computing paradigms and for some are already using them at sustained operational levels. Another line of work is to adapt PanDA to the evergrowing research networks and use network metrics to break regional boundaries and the grouping of processing sites. Finally, in order to guarantee sustainability, the PanDA team is working on reducing adoption costs for other experiments and engaging with them to expand collaborative approaches.

## Acknowledgements

We wish to thank all our colleagues who contributed to Big PanDA project. This work was funded in part by the U. S. Department of Energy, Office of Science, High Energy Physics and ASCR Contract No. DE-AC02-98CH10886 and Contract No. DE-AC02-06CH11357, and also supported in part by the Russian Ministry of Science and Education under Contract No14.Z50.31.0024.

## References


[1] ATLAS Collaboration, J. Inst. **3**, S08003 (2008)

[2] *LHC Computing Grid: Technical Design Report*, document LCG-TDR-001, CERN-LHCC-2005-024 (The LCG TDR Editorial Board) (2005)

[3] Jose Caballero Bejar et al., J. Phys.: Conf. Ser. **396**, 032016 (2012)

[4] Paul Nilsson et al., J. Phys.: Conf. Ser. **119**, 062038 (2008)

[5] Tadashi Maeno et al., J.Phys.Conf.Ser. **119**, 062036 (2008)

[6] David Groep et al., J. Phys.: Conf. Ser. **119**, 062032 (2008)

[7] Vincent Garonne et al., J. Phys.: Conf. Ser. **513**, 042021 (2014)

[8] Maxim Potekhin et al., J. Phys.: Conf. Ser. **331**, 072058 (2011)

[9] Fernando Barreiro Megino et al., J. Phys.: Conf. Ser. **396**, 032011 (2012)

[10] Douglas Thain et al., "Distributed Computing in Practice: The Condor Experience", Concurrency and Computation: Practice and Experience, Vol. 17, No. 2–4, pages 323–356, February–April, 2005

[11] S. Agostinelli et al., Nuclear Instruments and Methods A **506**, 250–303 (2003)

[12] Jakob Blomer et al., J. Phys.: Conf. Ser. **396**, 052013 (2012)

[13] Sergey Panitkin et al., "Integration of PanDA workload management system with Titan supercomputer at OLCF", Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP 15), 13–17 April 2015, Okinawa, Japan (in press)

[14] Tom Goodale et al., "SAGA: A Simple API for Grid applications, High-Level Application Programming on the Grid", Computational Methods in Science and Technology **12**, 1 (2006)

[15] Torre Wenaus et al., "The ATLAS Event Service: A New Approach to Event Processing", Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP 15), 13–17 April 2015, Okinawa, Japan (in press)

[16] Tadashi Maeno et al., "The Future of PanDA in ATLAS Distributed Computing", Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP 15), 13–17 April 2015, Okinawa, Japan (in press)

[17] Edoardo Martelli et al., "LHCOPN and LHCONE: Status and Future Evolution", Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP 15), 13–17 April 2015, Okinawa, Japan (in press)

[18] Torre Wenaus et al., "Computing for the LHC: The next step up", International Conference on Computing in High Energy and Nuclear Physics (CHEP 13), 14–18 October 2013, Amsterdam, The Netherlands (plenary talk)

[19] Simone Campana et al., J. Phys.: Conf. Ser. **396**, 032019 (2012)