



Dynamic Distribution of High-Rate Data Processing from CERN to Remote HPC Data Centers

T. Boccali¹ · D. Cameron² · N. Cardo³ · D. Conciatore³ · A. Di Girolamo⁴ · G. Dissertori⁵ · P. Fernandez^{3,15}  · A. Filipcic⁶ · M. Gila³ · C. Grab⁵ · J. Elmsheuser⁷ · V. Jankauskas⁸ · A. Klimentov⁷ · D. Kovalskyi⁹ · S. Lammer¹⁰ · D. Petrucic³ · T. C. Shulthess³ · F. G. Sciacca¹¹ · C. Serfon¹² · R. Walker¹³ · C. Wissing¹⁴

Received: 31 March 2020 / Accepted: 27 December 2020 / Published online: 8 February 2021
© The Author(s) 2021

Abstract

The prompt reconstruction of the data recorded from the Large Hadron Collider (LHC) detectors has always been addressed by dedicated resources at the CERN Tier-0. Such workloads come in spikes due to the nature of the operation of the accelerator and in special high load occasions experiments have commissioned methods to distribute (spill-over) a fraction of the load to sites outside CERN. The present work demonstrates a new way of supporting the Tier-0 environment by provisioning resources elastically for such spilled-over workflows onto the Piz Daint Supercomputer at CSCS. This is implemented using containers, tuning the existing batch scheduler and reinforcing the scratch file system, while still using standard Grid middleware. ATLAS, CMS and CSCS have jointly run selected prompt data reconstruction on up to several thousand cores on Piz Daint into a shared environment, thereby probing the viability of the CSCS high performance computer site as on demand extension of the CERN Tier-0, which could play a role in addressing the future LHC computing challenges for the high luminosity LHC.

Keywords Containers · CERN Tier-0 · HPC · Data distribution · ATLAS · CMS · CSCS · Spill-over

✉ P. Fernandez
pablo.fernandez@cscs.ch

- ¹ INFN Sezione di Pisa, Pisa, Italy
- ² University of Oslo, Oslo, Norway
- ³ ETH Zürich/Swiss National Supercomputing Centre (CSCS), Lugano, Switzerland
- ⁴ European Laboratory for Particle Physics (CERN), Geneva, Switzerland
- ⁵ ETH Zürich, Zürich, Switzerland
- ⁶ Joseph Stefan Institute, Ljubljana, Slovenia
- ⁷ Brookhaven National Laboratory (BNL), Upton, NY, USA
- ⁸ Vilnius University, Vilnius, Lithuania
- ⁹ Massachusetts Institute of Technology (MIT), Cambridge, MA, USA
- ¹⁰ Fermi National Laboratory (FNAL), Batavia, IL, USA
- ¹¹ University of Bern, Bern, Switzerland
- ¹² University of Innsbruck, Innsbruck, Austria
- ¹³ Ludwig-Maximilians-University, München, Germany
- ¹⁴ Deutsches Elektronen Synchrotron, Hamburg, Germany
- ¹⁵ CSCS, Lugano, Switzerland

Introduction

At the Large Hadron Collider (LHC) [1] at CERN, large data sets are being collected from four major detectors. The data are stored and processed by the CERN Tier-0 facility and subsequently distributed around the world by the Worldwide LHC Computing Grid (WLCG) [2]. The processing of primary data from the experiments at CERN will be increasingly challenging to scale in the future, specifically when the high luminosity LHC (HL-LHC) [3] will come into operation, from 2027 on.

The goal of this work is to show how computational peaks and on-demand data-processing workloads that might exceed the CERN Tier-0 capacity can be moved to CSCS (225 km away from CERN) to the Piz Daint supercomputer [4]. We evaluate a demonstrator in preparation of future LHC operation runs (from 2022 on). Since CERN Tier-0 workloads have never been run on a high performance computer (HPC), we aim at tackling all the technical challenges posed by such demanding workloads.

CSCS already operates a Tier-2 center (i.e. a regional compute center to support LHC Computing workflows for

the ATLAS [5], CMS [6] and LHCb [7] experiments), which runs on the CPU-only section of Piz Daint. This way of operating the LHC workflows on a high performance computing (HPC) system became feasible as a result of the *LHConCRAY* project, developed over the last two years in a collaboration between CSCS and the Swiss Institute of Particle Physics (CHIPP), and which forms the basis for the testing procedures described in this article.

Use Cases

There are scenarios in which each experiment could have frequent needs on resources outside CERN, depending on the duty cycle of the LHC collider. For example in phases of excellent LHC performance the available compute resources might be insufficient to ensure an immediate processing of the recorded events.

In an hypothetical scenario, ATLAS will in the future generate so much data that once or twice per week the workload on the Tier-0 site would have to be “spilled over” to other sites. For this demonstrator, we agreed to base our test on an average ATLAS data taking run as recorded during the LHC Run-2. We didn't aim at addressing the scaling up to Run-4 data volumes, which could be the subject of a future project. The size considered is such that around 10,000 cores (150 Piz Daint CPU-only compute nodes) would be needed for processing during 1–2 days. Such resources will be dynamically allocated, as it is the case for other communities using the scheduler of the system, and need to be made available via standard WLCG middleware. This poses challenges, as a non standard node setup is needed on demand on the nodes.

More specifically, the ATLAS workload being looked at consists of the Tier-0 reconstruction of the experiment *RAW* data (byte-stream detector output), with heavier I/O patterns and an increased number of files, compared to other Tier-2 workloads. This also requires making available around 800 TB of storage in the local Grid storage at the site, for the staging of input files and results. In parallel to our efforts, ATLAS have commissioned the Tier-0 spill-over to the Grid. We refer to such tests for evaluating the effectiveness of spill-over to one site only, CSCS, comparing processing of the same input dataset at CSCS and on the Grid.

During LHC data taking periods the CMS experiment performs a first reconstruction of the data, labeled as prompt reconstruction, within 48 h. This workflow is the main driver for the required CPU capacity that is needed at the Tier-0. In order to sustain Tier-0 activities during peak performance of the LHC within a constrained budget, only a subset of data is selected for prompt reconstruction. For Run-2 CMS commissioned the possibility to spill over Tier-0 workloads to other Grid sites. However, there was no need for CMS to use this functionality during Run-2. Before the run CMS

invested massively into optimisation of the reconstruction code [8]. The achieved improvements turned out to be sufficient to accomplish the Tier-0 workloads on the CPU resources provided by CERN. Nevertheless CMS wants to maintain the possibility for Run-3 and beyond to be able to select additional datasets for prompt reconstruction, that would require resources beyond the CPU capacity available at CERN. Having access to an additional 10,000 CPU cores during peak data recording at expected Run-3 conditions would allow CMS to promptly reconstruct up to 20% more data, that would have needed to be ‘parked’ alternatively. Data would be just safed and only be reconstructed after the run, typically some years after recording.

Background and Motivation

The Swiss High Energy Physics (HEP) computing community (the LHC members of CHIPP) and CSCS have started working on a shared Tier-2 since 2006 [10] and focused on the integration of the WLCG workloads into the existing HPC infrastructure since 2014 [11, 12]. Some important milestones since then are worth being mentioned:

- In 2014, ATLAS Geant4 simulation ran in production for 6 months on a Cray XK7 named *Tödi* at CSCS. Integrated by means of a modified ARC Compute Element [13], submitting remotely from the University of Bern to CSCS via *ssh* commands [14].
- In 2016, initial work on using containers [15] for HEP workloads was presented at the Cray User group (CUG).
- In 2016–2017, the *LHConCRAY* project (ATLAS, CMS, LHCb) developed the capability for integrating Piz Daint with the Swiss Tier-2 LHC experiment frameworks for all experiment workflows (including user analysis). The service went in production with 1.6k cores of the CPU-only partition in 2017.
- At the end of 2017 a decision was taken by the CHIPP Computing Board to gradually migrate the WLCG Tier-2 facility to Piz Daint, starting in April 2018 with about 4k cores and completing the migration in April 2019 the full Tier-2 (about 10k cores).

Classic HPC systems like Piz Daint are shared among a large number of communities. As a big shared resource, they have the potential of absorbing computational peak loads, and as such, they might be attractive for accommodating extra processing from the LHC experiments.

However, HPC systems are usually optimized for scalable parallel software and have network and I/O patterns that are atypical for HEP workflows. To optimize the memory footprint on the compute nodes and reduce operating system jitter, these systems run a stripped-down version of Linux with a heavily tuned kernel and without local disk. Container

technology has gone a long way to make such supercomputers look more like conventional Linux systems, but this does not solve all the challenges related to the integration with the complex LHC experiment frameworks. To that end, the centre already hosts as addition to the Piz Daint facility a full range of Grid services, as part of the CHIPP Tier-2 provisioning for ATLAS, CMS and LHCb.

In view of the challenges posed by the foreseen scale of the HL-LHC computing needs [3] ATLAS and CMS proposed to CSCS at the end of 2017 to investigate the implementation of an environment supporting Tier-0 spill-over activities on Piz Daint to act as a demonstrator. The project goals were (1) the elastic provisioning of Tier-0 prompt reconstruction of the experiment RAW data, (2) support steady and on-demand spill-over computational peaks, and (3) evaluate solutions and interaction between the centre and the experiments for such operations.

Existing Tier-2

Most of the work presented here builds on top of *Phoenix*, the existing Tier-2 site at CSCS described below. *Phoenix* is a Linux cluster dedicated to run Tier-2 WLCG workloads a [10] for ATLAS, CMS and LHCb, that has been in operation at CSCS for over ten years. Since April 2017, yearly site expansions have been accomplished by incrementally adding computational power in the form of reserved nodes on Piz Daint, and most of the components

of the Tier-2 are connected to both Phoenix and the LHC Computing Grid. The general architecture can be seen in Fig. 1.

The most relevant components in the overall HPC area of the Tier-2 are:

- Compute Nodes in the CPU-only section of Piz Daint. Each node has two Intel Xeon E5-2695v4 processors with hyperthreading enabled for a total of 72 cores, and 128 GB RAM. These nodes are statically reserved for the WLCG Tier-2, and configured to interface to the Grid middleware.
- A GPFS [16] scratch file system, shared between the Piz Daint HPC nodes and the Phoenix nodes, used exclusively for WLCG workloads. This file system is optimized for a large number of small files.
- Two ARC Compute Element (ARC-CE) [13] servers, one for production (*arc04*) and one for testing (*arc05*).
- 4 PB of dCache central storage [18], also shared between the two computing blocks.
- VO-boxes (for CMS and ATLAS), Squid [19] proxies for CVMFS [20], BDII and other relevant services (monitoring, accounting).
- Eight Data Virtualization Servers (DVS) [21], used to project the GPFS scratch file system to the compute nodes within Piz Daint.
- Cray DataWarp storage (DWS) [22] to provision iSCSI devices on demand for the compute nodes.

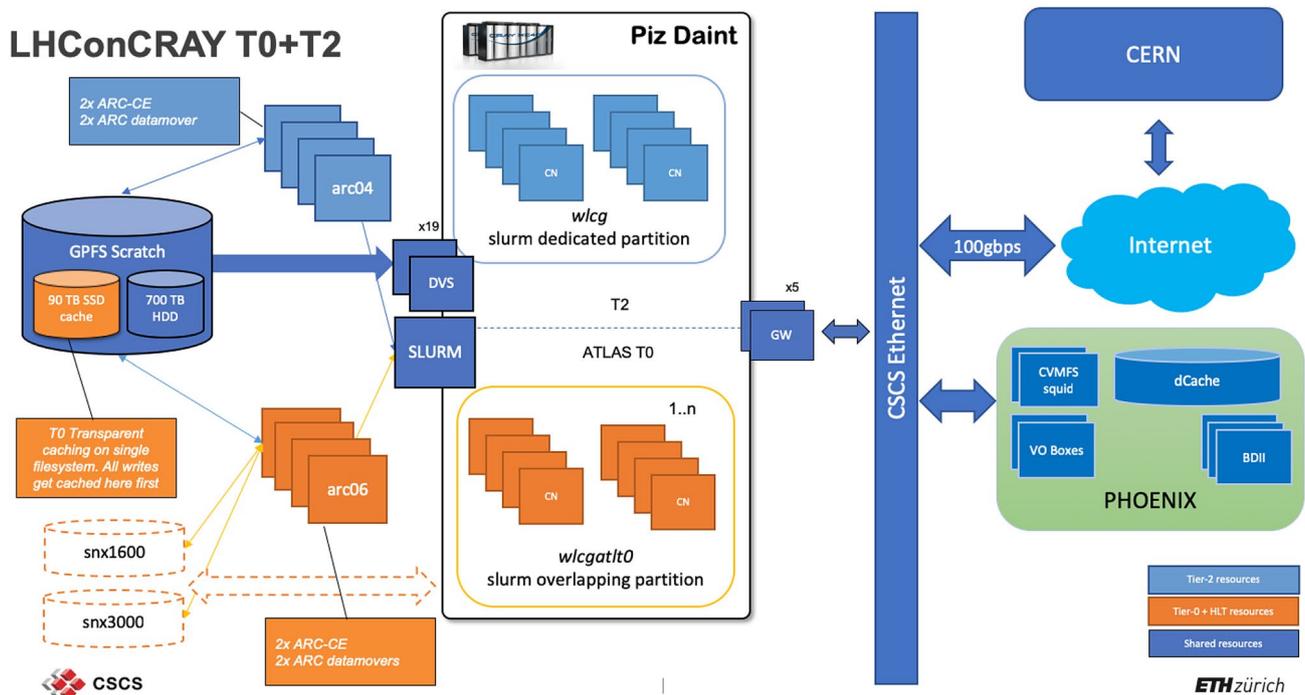


Fig. 1 Site diagram

Method and Implementation

Site Preparation

The initial part of the work started in April 2018 and consisted in determining the appropriate size of the test. As mentioned earlier, we decided for the demonstrator to base our estimate on the average size of an ATLAS run in 2018. This corresponded to around 10000 cores, or up to 150 compute nodes on the 1813-node CPU-only (multicore) XC40 section of Piz Daint. Most nodes have 64 GB RAM and a subset of them has 128 GB RAM (where Phoenix is located). These nodes are regularly used by other communities most of the time and would only be allocated to the Tier-0 tests dynamically, when needed. Given the memory footprint of the reconstruction code, we choose to use the nodes with more memory.

Then followed the design of a technical solution with a good balance between components from the pre-existing Tier-2 cluster and HPC platform that could be re-used, and the new ones that were needed for the Tier-0 evaluation. Re-utilizing existing hardware and infrastructure components was possible and desired: the GPFS scratch file system and most WLCG components (BDII, Squids, dCache) could be shared, but 4 additional ARC-CE servers were required for this work, two to work as entry points to the HPC infrastructure, and two to work as staging nodes for ATLAS. These 4 ARC-CE servers remain idle when there is no workload from the Tier-0.

After analyzing the input/output patterns of the Tier-2 workloads on the scratch file system, it was decided to extend it with an SSD layer of about 90 TB, which allowed it to reach a theoretical performance of about 11 GB/s and approximately 25'000 ops/s. This is sufficient to support the 150 extra compute nodes (in total) that would process the Tier-0 ATLAS and CMS workloads. On the HPC side, it was considered that the number of DVS nodes were insufficient for the task, so 12 additional nodes were commissioned and deployed from other parts of Piz Daint. These expansions (the SSD layer and the DVS nodes) were added in order to support this work, but remained in the Tier-2 cluster after it concluded in order to support the next regular yearly expansion.

Additionally, 1 extra Petabyte of storage capacity was made available on the dCache central storage to be shared between ATLAS (800 TB) and CMS (200 TB), to reach a grand total of 5 PB. Even though this extra space was made available to support this work, the space itself was not restricted exclusively to the Tier-0, but rather a sort of additional buffer of capacity for Tier-0 workloads.

The deployment of the SSD layer and the addition of storage space to dCache were transparent operations to the Tier-2. The addition of DVS nodes required a downtime for the HPC infrastructure.

Overall Configuration and Evolution from Tier-2 Setup

The schematic shown in Fig. 1 reflects the final configuration for both tiers. The changes introduced for the Tier-0 are an evolution of the original Tier-2 setup, which had been running in production for about 1.5 years. What follows is a recollection of the relevant configuration and hardware changes needed to accommodate Tier-0 workloads. All configuration changes were validated by the SAM tests [23] that are regularly sent to the site by the experiments.

Storage

The 730 TB GPFS scratch file system, shared by both Tier-0 and Tier-2, is exposed to the compute nodes using 19 DVS nodes connected to the CSCS Ethernet network with 40 Gbps links. This is an increase from the previous 8 nodes. The file system has a SSD cache layer in front of it, taking care of reducing the load on the backend disk storage. Each ARC-CE creates and keeps the job files under a directory tree on the scratch file system (*session directory*) that is unique to the job. Input files are cached by each ARC-CE on a *cache directory* on the scratch file system, which is accessible by all seven servers.

As mentioned in Sect. 2.1, the capacity of the dCache central storage has been increased to 5 PB to accommodate for Tier-0 workloads.

ARC-CE Endpoints and Job Containerization

In total there are 7 ARC Compute Element (CE) servers:

- Two pre-existing CEs, dedicated to the Tier-2 operations: a production server, named `arc04`, and a test server, named `arc05`.
- Two, newly deployed, CEs dedicated to the Tier-0: `arc06` and `arc07`.
- Three CEs for data staging. One dedicated to the production Tier-2 service: `arcds1`; and two dedicated to the Tier-0 service: `arcds02` and `arcds03`.

All these servers were deployed with a modified version of *nordugrid-arc-5.4.3-1* that includes minor changes permitting the generation of batch jobs compatible with container environments.

Generally speaking, Tier-0 and Tier-2 workloads are expected to run natively on CentOS-based Linux

distributions. At the time of running these tests, CMS is as well capable of running on any Linux operating system through the usage of Singularity containers, but ATLAS (and LHCb) require a CentOS-based environment to run.

The compute nodes on Piz Daint run the Cray Linux Environment (CLE) operating system, based on SUSE Linux Enterprise. This is an infrastructure requirement that, until recent times, has constrained WLCG workloads, preventing them access to many HPC platforms.

However, thanks to the advent of container technologies, when the Tier-2 was being commissioned, we created a publicly available [24] Docker container image that provides a suitable environment for all Tier-0 and Tier-2 WLCG workloads. All WLCG jobs that are not natively ready to run on any Linux operating system are then presented with a CentOS environment by means of containerization: the Docker image mentioned is instantiated using Shifter [25], and the job moved to the new environment before actually starting. Shifter is an HPC-focused container runtime engine available to all users on Piz Daint.

The Docker image is stored in *squashfs* format within one of the shared Cray Sonexion scratch file systems available on the system (in Fig. 1, `snx1600` and `snx3000`). These filesystems are not dedicated to WLCG and are only used for accessing the mentioned container image.

EGI accounting was disabled on all Tier-0 ARC-CEs to avoid disrupting the Tier-2 records or creating a new site. Local scheduler accounting was available.

Workload Management

The Workload Manager (WLM) of Piz Daint is *Slurm*. The ARC-CE endpoints for the Tier-2 submit jobs to a partition named *wlwg*, and the endpoints for the Tier-0 submit jobs to another partition, named *wlwgatlt0*. In *Slurm* terminology, a partition is conceptually similar to a queue in other WLMs.

- Partition *wlwg* consists of compute nodes dedicated exclusively to the Tier-2, where no HPC workloads from other users are allowed. These nodes were not used for this work.
- Partition *wlwgatlt0* overlaps with other partitions of the system and consists of about 150 nodes that are not dedicated exclusively to the Tier-0. These nodes are normally allocated with jobs from other communities on the regular HPC partitions, and only gets used by Tier-0 jobs when the partition has jobs.

Jobs landing on the partition with Tier-0 resources get higher priority than regular HPC jobs by means of a *Quality of Service* (QoS). This has the effect that, as soon as a node is assigned to run a Tier-0 job, the node will continue to be dedicated to these workloads until there are no more jobs

queued in the partition. Once a node is full of Tier-0 workloads, a new one will be assigned as soon as possible. The maximum wall-time for regular HPC and Tier-0 jobs is 24 h.

The memory consumption by each job tends to vary significantly over time. Considering that the WLM never starts all the jobs in a node at once, in order to maximize resource utilization, *Slurm* has been configured to ignore memory requirements and constraints. This was done by setting the configuration option `SelectTypeParameters` to `CR_CORE` on each of the partition configuration flags. To avoid nodes exhausting their available memory, `MaxCPUsPerNode` was configured to 68 (4 less than the actual core count of each node) and swap was enabled using Cray's *DataWarp* (DWS) [22], a technology that is capable of provisioning swap space on compute nodes using remote iSCSI endpoints on SSDs.

DataWarp also allows jobs or nodes to access remote SSD storage as a sort of local, temporary scratch space for jobs. Some basic testing has been performed on this front, but the results were inconclusive. Future work could be done on this area.

The majority of the work that runs on Piz Daint is pure HPC: multi-node code that benefits from a fast interconnect and, in many cases, GPU acceleration. This determines the WLM configuration and defines the default minimal allocatable unit for regular HPC jobs, which is a single node. Within each node, a user can then select whether to use all the available cores or a subset of them through the affinity settings. These settings provide, among other capabilities, the possibility to use or not use hyperthreaded cores.

This functionality is very handy for certain HPC codes, for example those that are CPU bound and suffer performance penalties when running on hyperthreaded cores. But is problematic for traditional WLCG workloads, where the minimum allocatable unit needed is sometimes 1 core, irrespectively of whether the core is hyperthreaded or not.

In the case of the Tier-2, the WLM configuration of each node belonging to the Tier-2 has been modified to accommodate for WLCG use cases by setting `CPUs=72` and `CoresPerSocket=1`. This, in conjunction with the previous change introduced at the partition level to ignore memory requirements and constraints, permits two different jobs to actually run on a single physical CPU core (with hyperthreading), maximizing the utilization of each node up to the value defined in `MaxCPUsPerNode`.

However, since Tier-0 resources are not dedicated specifically to WLCG and belong to the general pool of compute nodes available in the system, the specific WLM node configuration matching WLCG requirements cannot be easily implemented. This is because the configuration of each node is only evaluated at WLM daemon start, which makes dynamic reconfiguration costly in terms of service disruption and stability. It was still possible to set `MaxCPUsPerNode`

and `SelectTypeParameters` at the partition level. As a result of all this, the minimum allocatable unit for Tier-0 jobs is the same as HPC jobs: a single hyperthreaded physical core (2 threads).

This does not pose a problem of efficiency for Tier-0 workloads because, when allocating multi core jobs, which are the majority, the system is capable of pinning tasks evenly between real and hyperthreaded cores (i.e. an 8-core job will run on 4 real and 4 hyperthreaded cores).

Isolation between ATLAS and CMS jobs is done at the node level: once a node gets an ATLAS job, it will run only ATLAS jobs until there are no more ATLAS jobs queued. Similarly, once a node gets a CMS job, it will run CMS jobs until there are no more CMS jobs queued in the partition. Because there are only two user accounts running the jobs, one for each VO, this isolation is implemented by setting the configuration option `ExclusiveUser` to `Yes` on the partition configuration flags. In order to avoid interference during the few full-scale tests, there was a human coordination to prevent both VOs to submit jobs at the same time.

Additionally, the WLM node prolog and epilog scripts needed customizations to allow for rapid configuration when each node shifts from running regular HPC workloads to WLCG Tier-0 workloads, and vice-versa. These customizations can be summarized as follows:

- Mount and unmount the GPFS scratch file system.
- Mount and unmount the CVMFS file systems based on which VO is running on the node.

Services

Other than network firewall corrections, the BDII, squid proxies for CVMFS and the VO-box node running the *ATLAS Frontier* do not need any modifications to run Tier-0 workloads. However, the number of CMS VO-box running *PhEDEx* [9] and squid proxies has been increased from 1 to 3.

Compute Nodes

WLCG workloads running on compute nodes need to have CVMFS file systems mounted when jobs start. In the case of the Tier-2, since the compute resources are dedicated, CVMFS file systems are mounted at boot time. The CVMFS configuration relies on a two layered approach: the higher layer is an in-memory 6 GB cache and the lower layer is a posix cache, living on the GPFS scratch file system. This permits nodes to share data quickly, which is particularly useful when we have many jobs running the same software releases.

In the case of the Tier-0, CVMFS is mounted and unmounted using the customizations to the WLM prolog

and epilog scripts mentioned in Sect. 2.2.3. The configuration differs slightly from the one for the Tier-2: it is also a layered configuration, but since there is only one VO per node, 2 GB of memory are sufficient. This configuration is now the default for all non Tier-2 compute nodes of Piz Daint. Potentially, almost any node on the system can automatically mount CVMFS.

There were no other customizations needed on the compute nodes of Piz Daint to run WLCG Tier-0 workloads.

Network

CSCS connects to the Swiss network backbone provided by the NREN SWITCH [26] with a 100 Gbps link. The Storage Element dCache is connected to the internal CSCS network via 80 Gbps uplinks and to Piz Daint via five gateway nodes with 40 Gbps links. Any given compute node uses a statically defined default gateway (the maximum network bandwidth used for any given node is 40 Gbps) and there are fail-over mechanisms that allow gateway nodes or network links to fail without losing connectivity. All compute nodes utilize public IPv4 network addresses and CSCS policies permit outgoing connectivity to the Internet, as well as specific incoming traffic that allows *GridFTP* active connections to/from the compute nodes. This effectively allows jobs to fetch data from the outside of the site using a variety of protocols, including *https* or *XRootD*. Other unauthenticated protocols such as *dcap* or *ftp* are restricted to the boundaries of CSCS. Node I/O-related network traffic is not affected by these limitations, as this type of traffic goes towards DVS nodes, which have their own dedicated network links to the scratch filesystem. The available bandwidth for any given node within the High Speed Network (HSN) is sufficient to accommodate I/O and external network traffic.

Resulting System

As seen, the resulting system is an evolution of the pre-existing Tier-2 with very few resources dedicated exclusively to the Tier-0. Almost every piece of hardware that has been introduced to accommodate these new workloads, directly or indirectly benefit the Tier-2 and could, ultimately, be utilized in future Tier-2 capacity extensions.

In terms of capabilities, all the software modifications and changes introduced allow the overall system to be more adaptable. For instance, partitions are statically defined in the WLM configuration file, but can be expanded or reduced by issuing a single WLM command. This permits for rapid resizing of the Tier-0 resources, in a matter of minutes.

Since there is no need for resource reservation or compute node draining, the overall system presented is capable of running Tier-0 workloads, steadily or in bursts, without any operational differences.

ATLAS PanDA Configuration

For the integration with the ATLAS workload management system PanDA [27], it has been decided to create a dedicated PanDA resource that would handle exclusively the Tier-0 like workloads and a dedicated space token on the local dCache store for the temporary storage of the input and output data. In such a way we have ensured the full decoupling of the Tier-0 activities from the routine Tier-2 production operations.

The operational scheme devised does not depart from the one used for the Tier-2 workloads: the input data need to be made available on the local dCache storage area before jobs can be directed to the site. The pre-staging of the input data is thus performed asynchronously via FTS [28], so the RAW experiment data to be reconstructed are pre-placed at the site and made ready for processing. When the jobs are sent to the system via the ARC Compute Element, the inputs for each job are moved by ARC from dCache to the GPFS scratch file system. This is also the area where the outputs are written during job processing. After each job has completed execution in Slurm, the processing slots are released on Piz Daint and ARC takes care of the asynchronous stageout of the data to the dCache storage.

One last step in the chain is the final transfer of the data via FTS to the designated destination, typically CERN, but not necessarily. The performance considerations detailed in the next sections do not take into account the FTS data movements, which occur conveniently asynchronously and are not a specific feature of the Tier-0 workload.

CMS Configuration

All distributed computing resources that are used by CMS are consolidated in one large global HTCondor pool [29]. On the compute nodes a small pilot job gets executed to join the global HTCondor instance in order to allow the scheduling of payloads. On classical Grid sites the pilot jobs enter via Grid Computing Elements, like a CREAM-CE, a HTCondor-CE or an ARC-CE. For the presented Tier-0 use-case dedicated ARC-CEs were added to the configuration of the CMS global pool. Once a resource has been added to the global pool, it can be targeted by simply adding its site name to the white list of the job description that is submitted to HTCondor.

Although processing and prompt reconstruction of the CMS data at the Tier-0 are a special use-case, CMS has followed also here the same approach and has integrated the CPU resources at CERN in the global HTCondor structure. This allows a very flexible usage of the CERN resources. During LHC data taking most CERN resources are used for prompt reconstruction. If the demand for prompt reconstruction is reduced, any other task from CMS can utilize the

CPUs at CERN. The CPU allocation is driven by HTCondor priorities.

Due to the increasing reliability and capacity of wide area network links CMS adapted its computing model for Run-2 to allow for remote data access, which was basically excluded before. CMS commissioned a global data federation [30] that comprises all Grid Storage Elements (SE). It is sufficient to just know the logical file name (LFN) and the URL of an entry point to access any CMS file that is presently hosted on disk storage. After some optimization of the I/O layer in the CMS applications, jobs with low to medium I/O demands could run reading data from remote with a typical reduction in CPU efficiency of around 10% or less compared to local data access.

The reconstruction tasks of RAW data at the Tier-0 have medium I/O demands and qualify for remote data access. Therefore CMS built its spill-over setup for this Tier-0 test based on remote data reads. This simplifies the setup a lot, because no dedicated data transfer with additional book-keeping is required. During run-time reconstruction jobs write the output file to the local scratch disk of the processing node. At the end of the job that file gets copied to an SE, usually the one close to the CPU, but not necessarily.

The strong WAN link between CERN and CSCS allows the configuration of CPUs at CSCS for reconstruction of data that is hosted at CERN. For the Tier-0 spill-over the jobs running on the Piz Daint CPUs were reading directly from the CMS storage at CERN. Also files, that were produced by those jobs, were written back directly to CERN, i. e. no mass storage was involved at CSCS for the spill-over test. Jobs that run on the Tier-2 partition use the local storage as the primary option to read in data and to stage out produced files.

Validation Results

ATLAS Workloads Validation

As a starting step, the ATLAS PanDA configuration detailed in Sect. 2.2 has been validated using the HammerCloud test framework [31]. The tests consist of a continuous stream of lightweight MonteCarlo simulation jobs processing only a few events each, which test the functionality of every link of the processing chain and of the job lifecycle, from the ATLAS factories, through the submission chain, down to the compute and storage facilities at the site. No specific measure had to be introduced for this validation step.

Two typical Tier-0 workloads have been selected as the main validation tasks:

RAW data reconstruction on the *physics_BphysLS stream* $O(10\%)$ of *physics_Main* and RAW data reconstruction on *physics_Main*.

These have also been used by ATLAS for commissioning the T0 spill-over as tasks distributed to the Grid. We will therefore refer to the validation on the Grid as a meaningful term of comparison for the validation on Piz Daint, keeping in mind that the input data are exactly the same in both cases, thus a direct comparison is possible. The two workloads will be covered in the next two sections.

Validation for RAW Data Reconstruction on the *physics_BphysLS stream O(10%) of physics_Main*

This processing chain covers the reconstruction of one specific stream within the RAW data acquired for each Physics run at the LHC. The stream amounts to about 10% of the total data volume for each run, and in a realistic operational scheme this processing could take place on Piz Daint for each LHC run, thus we will refer to it as *steady spill-over* mode. On average, one run would include about 700k events for an input size of about 0.7 TB and would need about 20 Piz Daint nodes, or 1'300 cores, for a satisfactory turn-around to process this data in about half a day.

Contrary to the MonteCarlo workloads, the memory requirements for RAW experiment data reconstruction normally exceeds 4 GB per core, which by far exceeds the memory available on the Piz Daint nodes. By employing a multi-threaded approach, we can however take advantage of memory sharing between threads on the same node and thus attempt to bring the memory usage to manageable levels. The tuning of this involved considerable effort, as we found ourselves hitting the physical node memory limit (*Resident Set Size - RSS*) on the nodes. This caused job failures as the Linux *Out of Memory (OOM)* killer would try to free up resources on the nodes by means of killing processes, but also we occasionally experienced node crashes. The other consideration to be made for the multi-threaded approach is that the advantage of memory sharing, increasing as the thread-count increases, is counter-balanced by a disadvantage in CPU / WallClock efficiency, due to the intrinsic nature of the workloads: during the processing, each thread of a job produces its own output. At the end of the processing, all the job partial outputs need to be merged into one; this step only uses one thread, thus impacting more the CPU/WallClock efficiency for higher thread counts.

A laborious tuning involving thread-count per job, cgroups and Slurm configuration, amount of memory reserved for the CVMFS in-RAM cache and use of swap on the Cray Data Warp Service (DWS) resulted in a

configuration working with sufficient efficiency for the task. The optimal job thread-count was found to be 32, thus two jobs were allowed on each of the used nodes.

Additional complications arose due to the heavy Input/Output pressure of such jobs on the scratch file system. The number of DVS nodes exposing GPFS to Piz Daint had to be doubled in order to prevent instabilities on the DVS layer. Additional tuning was also needed on the side of the ATLAS WMS: the number of events to process assigned to each job had to be increased substantially from the default in order to improve the CPU/WallClock efficiency, and some PanDA brokering difficulties had to be overcome to be sure that the jobs would be forcedly sent to Piz Daint, rather than being freely brokered among the sites that have the input dataset available on their storage elements.

The main validation metric we use for the workload is the turn-around time, but we will also look at other quantities like CPU/WallClock efficiency, CPU time per event and Wall time per event. These are summarised in Table 1. The values quoted refer to the most performant configuration on Piz Daint (two 32-thread jobs per node, filling the whole node). On the Grid, sites might have different settings for number of threads per jobs. The most common is 8 threads, with CERN running 4-thread jobs.

The task has completed in 13 h on Piz Daint without job failures, while on the Grid it has completed to the 90% level in 18 h and it then took well over twice as long to be processed to the 100% level. This was due to job failing at some sites and re-tried until they succeeded. The other metrics shown in Table 1 indicate that the absolute performance for the successful jobs is better on the Grid sites, for example CPU time per event and wall time per event are lower on average on Grid sites. This is mainly due to the fact that most of them are configured to run 8-thread jobs. In such a way, the weight of the serial portion of the workload on the total runtime is greatly reduced compared to a 32-thread job configuration used on Piz Daint. However, the deciding factor in terms of turn-around time is seen to be not the absolute CPU time per event, but rather the site stability: on this run no job failed on Piz Daint, while a sizeable number of failures on the Grid sites causes the total processing to last considerably longer, as the jobs that fail are re-tried, often multiple times until they succeed. We consider the 32-core configuration to be performing well on Piz Daint and consider the task validated.

Table 1 Validation summary for the ATLAS reconstruction of *physics_BphysLS stream O(10%) of physics_Main*

	Turn-around time	CPU/WC efficiency	CPU time/event	Wall time/event
Piz Daint	13 h	32% (32-thread)	20.8	78.4
Grid	18 h (90%), 46h (100%)	53% (various)	13.6	33.5

Validation for RAW Data Reconstruction on *physics_Main*

With this processing chain, we cover the full reconstruction of all RAW data for a given run, covering every Physics stream recorded. Although runs can vary in duration and volume of data recorded wildly, we have estimated from the LHC performance history that the average realistic size of a run would be of the order of about 32 M events, amounting to about 35 TB of RAW data. The spill-over of a full run to a remote centre would not occur on a regular basis like for the case treated in the previous paragraph, but it would be triggered whenever the CERN Tier-0 runs the risk of building a backlog of runs to process. We refer to this mode as *on demand spill-over* mode. An average run could be processed on about 150 Piz Daint nodes, or 10,000 cores, and the processing would be expected to last about one to two days, depending on operational factors. Also in this case, commissioning runs have been performed on the Grid, using the same input data that we have used for attempting validation on Piz Daint.

We found that, with the processing of the full run, the memory demand increases a bit compared to the *steady spill-over* mode. This posed the demand to instantiate swap on DWS on 150 nodes. However, this is an action that failed, eventually leaving us in the condition of being unable to instantiate swap on any node. At the time, this was understood to be due to a bug on the Cray DWS layer, which could not be fixed within the time frame of our validation project.

The optimal configuration we had worked with, running two 32-thread jobs on each node, did not work without swap, leading to jobs being killed and node crashes, which rendered the errors unrecoverable. With the goal of running the processing of all the events in the sample run to completion, we did explore several combinations of jobs per node and threads per job, and for each of them we did measure the CPU/WallClock efficiency. This involved tuning some slurm parameters specifically for this test. It became immediately obvious that for this test we could not make use of all the threads available on the node. It turned out that the only configuration that allowed us to run the processing to completion was to run two 16-thread jobs on each node, thus sacrificing half of the CPU resources. The efficiency figures are reported on Table 2 for some of the configurations we tried out. In parenthesis we report the efficiency value scaled

to take into account the cores left idle by the specific configuration we needed to put in place for this specific test. The values quoted for the Grid refer to a test carried out independently by ATLAS, using the same input dataset. Again, in such case, the number of threads per job might vary from site to site.

The configuration running three 16-thread jobs per node had an edge in terms of performance. However, with more jobs in the system, also the number of intermediate files created and used during processing increased to a level sufficient to overload the DVS layer. This was also understood to be due to a software bug. Although promptly identified and addressed by the Cray engineers, a patch could not be applied timely, as it would have required a downtime for the whole Piz Daint. This also prevented us from reaching the full scale of 10,000 cores for the test. The maximum number of concurrent nodes used was up to 80, or just over 50% of the foreseen full scale. Only half of the threads available on each node have been used for this test, thus further reducing the number of concurrent running cores to up to ~ 3000.

In conclusion: the *physics_Main* test completed successfully and can be considered partially validated, but from Table 2 we can see that the task turnaround time exceeded the one on the grid by 35% (~ 7 days on Piz Daint compared to ~ 5 days on the Grid). In both cases, this exceeds the initial expectations, at the time of testing. We expect to be able to reduce the run time to a similar or better level on Piz Daint, once the problems affecting the provisioning of the swap and the DVS nodes have been resolved (which could not be done within the time frame of the project), in order to claim the full validation of the test. We propose this for future work.

CMS Workloads Validation

CMS Health Check

Due to the site integration via ARC-CEs, common testing and monitoring tools from WLCG work out of the box. CMS employs the WLCG Service Availability Monitoring (SAM) [23] in order to validate basic site functionality like job submission, local data access, stage-out, presence of CMS software and availability of local Squid caches. After the initial configuration of the dedicated ARC-CE for the Tier-0

Table 2 Validation summary for the ATLAS reconstruction of *physics_Main*

	Jobs per node	CPU/WC efficiency	Turnaround time	CPU time/event	Wall time/event
Piz Daint	2 × 32-thread	27%			
Piz Daint	1 × 56-thread	23% (effective 11%)			
Piz Daint	3 × 16-thread	42% (effective 32%)			
Piz Daint	2 × 16-thread	62% (effective 31%)	7 days 7 h	22.4	55
Grid	Various	62%	5 days 9 h	18.2	31.6

test, the functionality was verified by SAM tests and the CE further exercised using the Hammercloud tool [31]. This tool sends a representative workload to all sites. The jobs running on Piz Daint showed in the beginning a rather high number of jobs failing due to problems with accessing the software in CVMFS. The succeeding jobs showed a surprisingly low CPU efficiency of $\sim 30\%$ while Hammercloud jobs typically have a CPU efficiency of $\sim 90\%$. Both problems got cured by adjusting the cgroups configuration. Many HPC machines are not configured to allow running multiple jobs per node, or even per core, so in order to maximise utilization of the resources we had to force the scheduler to allow pinning single tasks (or, ultimately, single core jobs) to single cores, even to those in hyper threading mode. This change was not straightforward as we had to fiddle first with oversubscribing cores (hence the low efficiency), and then properly tuning cgroups and node configuration to allow this.

Over a period of a few weeks the Piz Daint allocation for the Tier-0 test was used to process Monte Carlo (MC) simulation workflows. Some of the campaigns also involve the execution of the digitization and the reconstruction steps. During the digitization CMS nowadays employs 'PreMixing' to overlay pile-up events. These PreMixing libraries are typically large datasets, that easily reach sizes of several 100 TB. CMS can afford to store two to three replicas at big Grid sites with well performing storage and good network connectivity. All other sites read in the pile-up via remote reads.

During the commissioning the CPU allocation for the Tier-0 test was small, but sufficiently large, and typically only some hundred CPUs were utilized in parallel. The CPU efficiency as measured by the application was very similar to values measured on Grid sites as shown in Fig. 2. For the comparison only workflows that had a considerable amount

of jobs running on Piz Daint and on Grid sites were considered. Also the values for processing time per event were found to be compatible.

CMS Tier-0 Replay Test

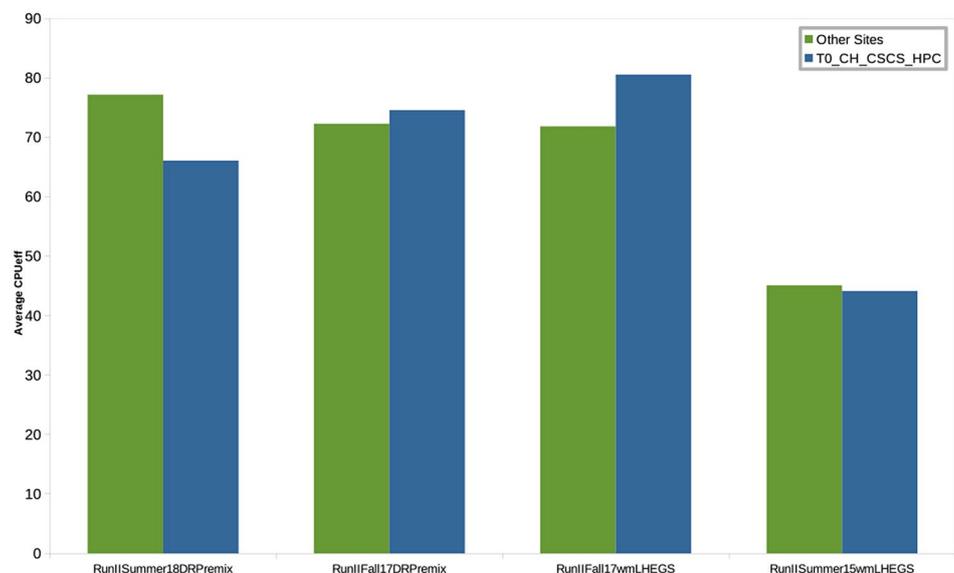
The possibility to spill-over workloads that traditionally are executed at CERN is of special interest and has been investigated in more detail. The main focus has been on the prompt reconstruction, that is applied to data that has just been taken by the CMS detector.

For testing purposes CMS uses so-called 'replays'. They are routinely exercised for every new release of the CMSSW software that is going to be used for prompt reconstruction or when any other significant change in the Tier-0 infrastructure is planned. Since the Piz Daint resources were already commissioned for Monte Carlo production and the execution of the prompt reconstruction on non-CERN resources was already implemented, a replay test on Piz Daint was mainly a small configuration change.

The CMS reconstruction software is a multi-threaded application and the number of active threads can be configured in a flexible manner. The default is to use 8 threads, which is based on an agreement across WLCG. The amount of thread-safe code in the CMS reconstruction application allows scaling to a higher number of threads while remaining CPU efficient. The WLCG standard is 2 GB RAM per core. For the reconstruction of usual proton-proton collisions the CMS application fits easily into 16 GB RAM with 8 CPU cores allocated.

One replay test was executed at a scale of ~ 2000 cores at CSCS. The replay could also utilize CPUs at CERN at the same time. Since the CERN resources were busy with processing for the ongoing data taking and other activities,

Fig. 2 CPU efficiency for various CMS MC production campaigns that run at CSCS and at other Grid sites. 'PreMix' campaigns involve remote data access



a fraction of ~ 80% of the jobs ran at CSCS. The replay was concluded successfully after processing all input data of about 13 TB. All input data was directly read in via WAN from disk storage at CERN. During the lifetime of the jobs output files were written to the local working directory, which is provided via the GPFS scratch file system. At the end of each job the produced output data was transferred back to disk at CERN for longer term storage. Figure 3 shows the time per event distribution of events processed at CSCS and CERN. The larger spread of the values for CERN is attributed to the variety of hardware types that are operated in contrast to the homogenous compute nodes of Piz Daint. Since the main purpose of excluding performance degradation at Piz Daint has been fulfilled, no further investigations were performed. The job failure rate was also investigated and found to be on the same level.

Another replay test was executed to reach a scale of several thousand cores. The general setup was like the one described above. The number of utilized cores easily reached 8000 cores, when an internal (artificial) limit of the CMS replay machinery was reached. The job success rate at CSCS was again close to 100% and no evidence was found that any limitation was reached for the achieved scale. At the time of this replay the CERN resources were completely occupied processing real data and no direct comparison of job classes running at the two locations was possible.

Summary and Conclusions

ATLAS Conclusions

We have come very close to fully validating a prototype implementation on Piz Daint for ATLAS Tier-0 spill-over.

We have identified a valid and performant configuration for the *steady spill-over* mode for the *physics_BphysLS stream*. For what concerns the *on demand spill-over* mode to be used to reconstruct full *physics_Main* runs, we have been limited in our quest by the surfacing of software bugs on the Cray burst buffer and DVS layers. We have good indications that we would be able to meet the target performance once that limitation would no longer be in effect.

It has been a very laborious exercise that has involved several experts on both sides. This is not surprising, since we aimed at commissioning a general purpose HPC system for a workload whose hardware demands in some ways exceed the system specifications. So a lot of tuning had to be put in place in order to overcome the limitations. We consider the results we have obtained a solid base on which to base possible future work

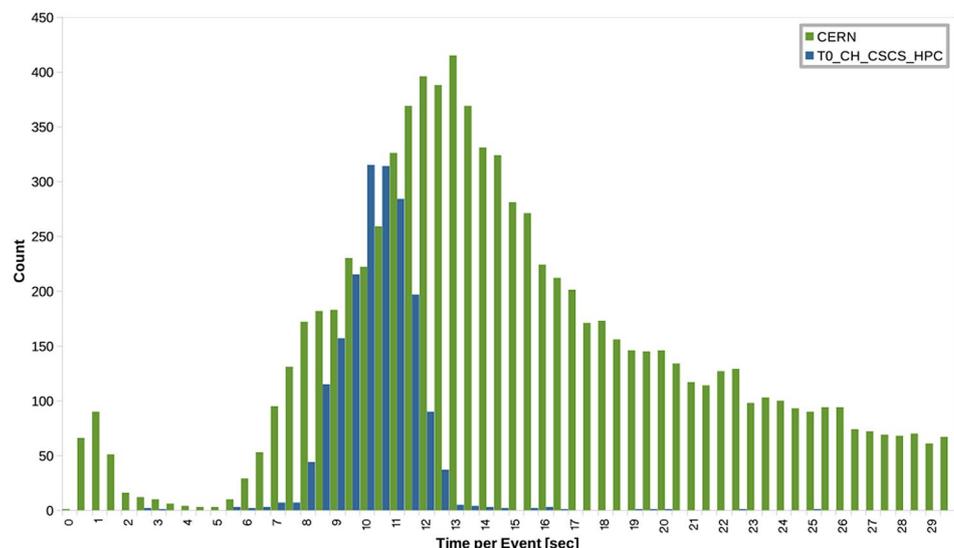
CMS Conclusions

CMS uses routinely a CPU allocation on Piz Daint at CSCS for usual workflows since several months. In order to test a spill-over of prompt reconstruction workflows, that are usually only run at CERN, a dedicated allocation was commissioned. This allocation was added as pure CPU extension, local disk storage was only used to write files during the life time of individual jobs. Data were directly read from CERN storage and got staged back at the end of the job.

During commissioning MC workflows were executed on the scale of some hundred CPU cores. The performance regarding CPU efficiency, time per event and failure rate was found very similar to other Grid sites.

To test the spill-over two replay tests were done, one at a scale of 2000 CPU cores and another reaching ~ 8000 cores

Fig. 3 Time per event for prompt event reconstructions at CSCS and at CERN



utilized in parallel. The performance of the CSCS resources were observed to be similar to the CERN resources.

The executed tests did not uncover any obvious problems or scaling limitations at the exercised scales. A successful longer term test would still be required to declare the resource fully production ready. The presented setup is a viable option for future running conditions, where limited availability of CPU resources at CERN might be compensated in order to allow more flexible prompt reconstruction scenarios.

Summary

A demonstrator of a platform for running ATLAS and CMS Tier-0 workloads on Piz Daint at CSCS has been implemented and exercised at a scale of several thousand CPUs. The resources have been elastically provisioned at the centre, mimicking the use case of absorbing computational peaks from CERN, on resources that are generally used by other communities.

Some needed elements were already present, like pre-existing Grid middleware and outbound network connectivity from the compute nodes. Other integration efforts were costly, as this was the first time ever that Tier-0 workloads went to an HPC system. We found no major technical show-stopper and managed to validate most workflows exercised, with some room for improvement.

We hope our findings can help driving design of the next generation machine(s).

Funding Open Access funding provided by ETH Zurich.

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

- Evans L, Bryant P (eds) (2008) LHC machine JINST 3:S08001
- Bird I (2011) Computing for the Large Hadron Collider. *Annu Rev Nucl Part Sci* 61:99–118
- Schmidt B (2016) The high-luminosity upgrade of the LHC: physics and technology challenges for the accelerator and the experiments. *JPCS* 706:022002
- Zurich ETH (2019) CSCS Piz Daint supercomputer specifications and upgrade history. https://www.cscs.ch/computers/piz_daint/. Accessed 1 Jul 2019
- ATLAS Collaboration, Aad G et al (2008) The ATLAS experiment at the CERN LHC JINST 3:S08003
- CMS Collaboration, Chatrchyan et al (2008) The CMS experiment at the CERN LHC. JINST 3:S08004
- LHCb Collaboration, Alves A et al (2008) The LHCb detector at the LHC. JINST 3:S08005
- Cerati G (2015) CMS Collaboration (2014) Vertexing and tracking algorithms at high pile-up. *PoS Vertex*. <https://doi.org/10.22323/1.227.0037>
- Sanchez-Hernandez A, Egeland R, Huang C-H, Ratnikova N (2012) Magini N and Wildish T From toolkit to framework—the past and future evolution of PhEDEX. *J Phys Conf Ser* 396:032118
- CHIPP Collaboration, Fernandez P et al (2019) CSCS LCGTier2 web. <https://wiki.chipp.ch/twiki/bin/view/LCGTier2>. Accessed 1 Jul 2019
- Sciaccia FG, Haug S, ATLAS Collaboration (2017) ATLAS and LHC computing on CRAY. *J Phys Conf Ser* 898:082004
- Sciaccia FG, Weber M, ATLAS Collaboration (2019) Production experience and performance for ATLAS data processing on a Cray XC-50 at CSCS. *EPJ Web Conf* 214:03023
- Ellert M et al (2007) Advanced resource connector middleware for lightweight computational grids. *Future Gener Comput Syst* 23:219–240
- Hostettler M (2015) Enabling the ATLAS Experiment at the LHC for High Performance Computing, Masterarbeit an der philosophisch-naturwissenschaftlichen Fakultät der Universität Bern
- Benedicic L, Gila M, Alam S, Schulthess TC (2016) (CSCS) Opportunities for container environments on Cray XC30 with GPU devices. In: *Published in CUG conference Proceedings*. https://cug.org/proceedings/cug2016_proceedings/includes/files/pap175s2-file1.pdf. Accessed 1 Jul 2019
- Schmuck F, Haskin R (2002) GPFS: a shared-disk file system for large computing clusters. In: *Proceedings of the FAST'02 conference on file and storage technologies, USENIX*, pp 231–244
- Yoo AB, Jette MA, Grondona M (2003) SLURM: simple Linux utility for resource management JSSPP. *Lecture notes in computer science*, vol 2862
- Deutsches Elektronen Synchrotron, DESY. The dCache Project. <http://www.dcache.org>. Accessed 1 Jul 2019
- Duane Wessels Squid and ICP: past, present and future. *Proceedings of the Australian Unix Users Group* (1997)
- Blomer J, Fuhrmann T (2010) A fully decentralized file system cache for the CernVM-FS computer communications and networks (ICCCN), pp 1–6
- Sugiyama S, Wallace D (2008) Cray DVS: data virtualization service in cray user group conference (CUG)
- CRAY Inc (2019) CRAY DataWarp applications I/O accelerator. <https://www.cray.com/datawarp>. Accessed 1 Jul 2019
- Andrade P et al (2012) Service availability monitoring framework based on commodity software. *J Phys Conf Ser* 396:032008. <https://doi.org/10.1088/1742-6596/396/3/032008>
- ETH Zürich, Gila M et al (2019) Cscs/wlwg_wn Docker Hub. https://hub.docker.com/r/cscs/wlwg_wn. Accessed 1 Jul 2019
- Shifter, NERSC. <https://github.com/NERSC/shifter>. Accessed 1 Jul 2019
- SWITCH Foundation (2019) <https://www.switch.ch>. Accessed 1 Jul 2019

27. US ATLAS (2019) <http://news.pandawms.org/atlas.html>. Accessed 1 Jul 2019
28. CERN FTS service (2019) <http://information-technology.web.cern.ch/services/file-transfer>. Accessed 1 Jul 2019
29. Balcas J et al (2017) Stability and scalability of the CMS global pool: pushing HTCCondor and GlideinWMS to new limits. *J Phys Conf Ser*. <https://doi.org/10.1088/1742-6596/898/5/052031>
30. Bloom K et al (2015) Any data, anytime, anywhere: global data access for science, 2015 IEEE/ACM 2nd international symposium on big data computing (BDC). Limassol 85. <https://doi.org/10.1109/BDC.2015.33>
31. Schovancova J, Di Girolamo A, Fkiaras A, Valentina M. Evolution of HammerCloud to commission CERN compute resources. <https://cds.cern.ch/record/2646247>. Accessed 1 Jul 2019

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.