Enabling CMS Experiment to the utilization of multiple hardware architectures: a Power9 Testbed at CINECA

T. Boccali[1], D Spiga[2], Alan Malta Rodrigues[3], M. Mascheroni[4], F.Fanzago[5]

[1] INFN Sezione di Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy
[2] INFN Sezione di Perugia, Via A. Pascoli 23c, 06123 Perugia, Italy
[3] University of Notre Dame, USA
[4] University of California San Diego, La Jolla, CA, USA
[5] INFN Sezione di Padova, Via Marzolo 8, 35131 Padova, Italy

Abstract. The CMS software stack (CMSSW) is built on a nightly basis for multiple hardware architectures and compilers, in order to benefit from the diverse platforms. In practice, still, only x86_64 binaries are used in production, and are supported by the workload management tools in charge of production and analysis job delivery to the distributed computing infrastructure. Profiting from an INFN grant at CINECA, a PRACE Tier-0 Center, tests have been carried on using IBM Power9 nodes from the Marconi100 HPC system. A first study on the modifications needed to the standard CMS WMS systems is shown, and very positive proof-of-concept tests have been conducted up to thousands of computing cores, also including an initial utilization of the GPUs which the nodes host. The current status of the tests, including plans to support multi-architecture workflows, are shown and discussed.

1. Introduction

The Compact Muon Solenoid (CMS [1]) experiment at CERN has been recording LHC [2] collisions at 7, 8 and 13 TeV since 2009, collecting around 100 PB of RAW data. In order to exploit this vast amount of data and produce physics knowledge (more than 1000 papers have been published so far), a large computing infrastructure has been built since the early 2000s and included in the Worldwide LHC Computing GRID [3]. The CMS experiment is one of the largest scientific computing globally, with almost 1 Exabyte of storage and 300,000 computing cores operational 24x7. By far, the largest consumer of computing resources in CMS is CMSSW [4], an in-house software realized by the CMS community in the last 15 years. CMSSW is released under the Apache 2.0 license. It consists of ~10 Million lines of code (~1/3 of the Linux Kernel) and covers most of the computing needs of the CMS experiment, from data reconstruction, simulation to analysis.

In the next few years LHC will undergo a profound upgrade, allowing for collisions which are expected to be both at higher energy (14 TeV) and more complex (with ~200 superimposed collisions instead of ~35 as it is today). Meanwhile the collaborations, including CMS, are working on detectors upgrade and this will translate into the increases of the acquisition channels which will select many more events (up to 7.5 kHz instead of 1 kHz).

While the current computing infrastructure is perfectly adequate to accommodate the current needs of CMS, the fulfillment of the future computing demands several major changes in the computing are needed. By simple scaling, the resources needed would be in excess of what we have today but an intense R&D program, ongoing for at least 5 years, is bringing down the requests to a more reasonable level, but still incompatible with expected funds.

One of the expected handles to overcome the excess in needs is to use allocations at Supercomputer (HPC) centers. In particular, ExaScale systems, capable of 10^{18} floating point operations per second, are being deployed in the same time range. For this reason, CMS is keeping investing effort in order to integrate HPCs [5] and, since 2019, this results in the ever increasing capacity used by CMS at those machines.
These systems deploy a variety of computing solutions, usually different from our standard systems based on High Performance Throughput farms with *x86_64* compatible CPUs, as ARM 64 bit (AARCH64) and IBM Power 8/9. CMS has included these architectures in its continuous build infrastructure, while not validated for physics utilization, CMS would be in the position to process on ARM and Power, given a large enough allocation to motivate the human effort.
After a short description of the national context and the related background of the resources integration, the paper describes the major challenges identified and addressed in order to successfully extend the computing systems of CMS to support the Power9 at Marconi 100. Finally the early results and next steps are summarized.

2. CINECA and the INFN computing infrastructure

The Italian National Institute for Nuclear Physics (INFN) [6] is one of the major contributors to the CMS experiment, with ~13% of the physicists and computing resources. In particular, INFN computing is distributed on 10 large centers (1 WLCG Tier-1 and 9 Tier-2s), at national labs or university departments. Bologna hosts the CNAF [7], a WLCG Tier-1, and also the italian node of PRACE/EuroHPC at CINECA [8], responsible for the deployment and management of a Tier-0 HPC system. The two centers are very close (in "network metrics") and are connected via a DCI capable of up to 1.2 Tbps.
Since a few years now, various collaborations have been established allowing to start several integration activities using different systems at CINECA. One of the major activity with considerable success concluded on February 2021 and it was carried on in the context of a PRACE grant[1] [9]. The focus was on the integration of *x86_64* (KNL system Marconi A2) resources in the CMS Distributed Computing Infrastructure. Other systems of interest at CINECA are Marconi A3 (Skylake) and Galileo 100 (Intel plus GPU).
More recently the bulk of the interest focused on the integration of Marconi 100[2], a machine that deploys 980 IBM Power9 nodes with 4 Nvidia V100 each, with 256 GB of RAM per node.

3. Integration challenges

In order to efficiently use the Marconi 100 resources offering nodes based on Power9 architecture several challenges, spanning over several layers of the software and computing system of CMS, have been identified and addressed in order to build a testbed with the ultimate goals of allowing the validation of the technical integration as well as the validation of the physics output.
In particular four main areas have been identified:

- **Software**: a mandatory prerequisite is that the application software is compiled for *ppc64le* architecture. Technically this is already done at CMS, which produces nightly builds for all the recent CMSSW releases for *x86_64* and *ppc64le* architectures. For each architecture CVMFS is used to provide access to the software. CMS is quite advanced in this respect, however events processed with architectures different from *x86_64* require a physics validation.
- **Workload Management System**: CMS has two different central WMS: WMAgent [10] for "central processing", CRAB [11] for end user "analysis". They are designed to work with *x86_64* workflows and nodes. In order to dispatch payloads to Marconi 100 we need a central Workload Management System architecture-aware in order to build payloads that properly

[1] PRACE Grant n° 2018194658, P.I. Dr. Tommaso Boccali
[2] Marconi 100 was in July 2020 the HPC # 9 in Top500.org.

configure the runtime environment at site and to define the HTCondor [12] requirements according to the expected architecture. So, the central agents required patches in order not to assume *x86_64*, but to leave the architecture as an external parameter to be defined at workflow injection time and to be configured via auto discovery mechanism on the resource itself.

- **Experiment runtime environment:** The CMS workload management distributes libraries and executables in order to process CMS jobs on the remote nodes. They are almost all Python based, with the exception of the HTCondor Startd daemon that needs to join the Global Pool of the experiment; it is also responsible to declare the GPU availability, if any. In order to have a system able to run on Power8/9, a few changes have been implemented to the WMA system aiming to allow the architecture auto-discovery and thus to set the correct version of the Python scripts. Regarding HTCondor, initially we compiled one on our own, while for the future it should be maintained directly by HTCondor developers.
- **Site runtime environment:** The site needs to offer a WLCG compliant environment. In order to achieve that, we needed to find a way to prepare on Marconi100 a proper execution environment, including the support for GPUs (a key to avoid wasting the majority of the node performances). The solution has been built using a combination of tools such as: singularity [13] which allows the inclusion of GPU support, Nvidia drivers, libraries and tools (--nv option); CVMFS [14] grants access to the VOClient for VOMS authentication. To produce a singularity image for PPC we used QEMU to fully virtualize a PowerPC system on a *x86_64* box. From a CentOS7 docker image for *ppc64le* we built our customized image adding the gfal dependency, and finally converted to the singularity image.

4. Testbed setup

As explained, the Marconi 100 integration activity comes after a series of successful experiences and this allowed us to reuse the previous handshakes that granted the presence of CVMFS as well as the usage of singularity (see Fig. 2). Regarding the networking initially we enabled the outgoing connectivity via TSOCKS mechanism [15-16], but in a later stage the routing both to CNAF and CERN was fully available following the very same pattern already available during Marconi A2 integration. Resource provisioning was performed without an instance of a HTCondor computing

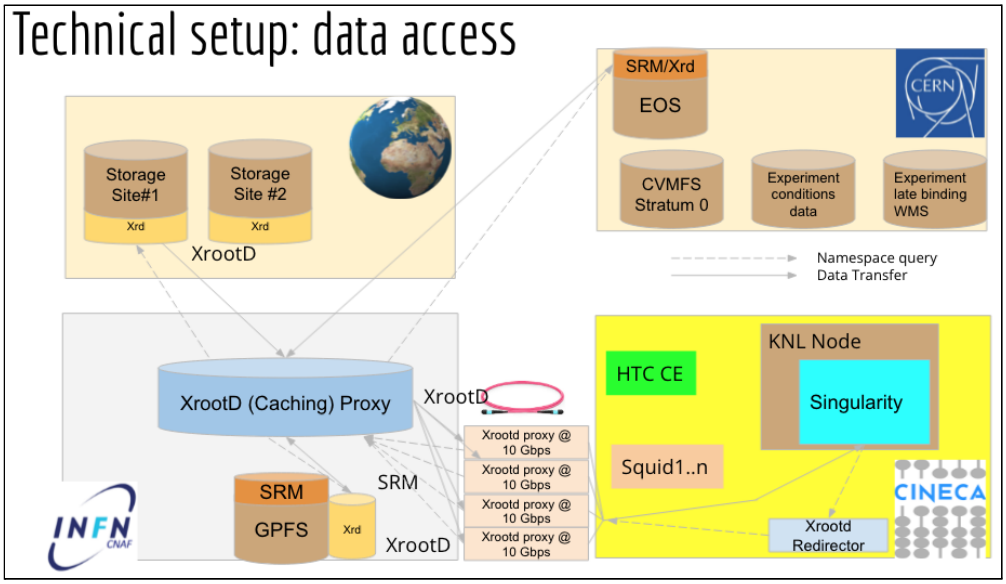

Figure 1. Diagram representing the integration schema summarizing the flow enabled during the Marconi A2 activity.

element, by using manual glidens. The strategy is to use the latter, creating a lightweight mechanism to inject and monitor glideins in order maintain a pressure on CINECA's SLURM queues. This is possible thanks to the manual pilot functionality officially supported by CMS. There was no need for special configurations for the Marconi100 entry in the pilot but the enforcing of *Singularity = NEVER*. This is because the runtime is provided in our singularity image to provide the WLCG compliant environment. Once we prepared all this, in order to commission the system we followed two distinct patterns: the first was performed with analysis jobs and the second via centrally managed release validation workflows. The CMS Release Validation are complex workflows composed by 55 distinct and chained tasks.

4.1. Marconi100 integration with CMS Analysis Infrastructure (CRAB)

The reason we started with the CRAB system is just because the setup of the workflow is simpler with respect to WMAgent, but good enough to implement the very first Proof of Concept aiming at verifying the goodness of the prepared setup at Marconi100. CRAB is not natively ready for cross platform so we prepared CRAB tasks directly on a PowerPC UI and then we performed the related

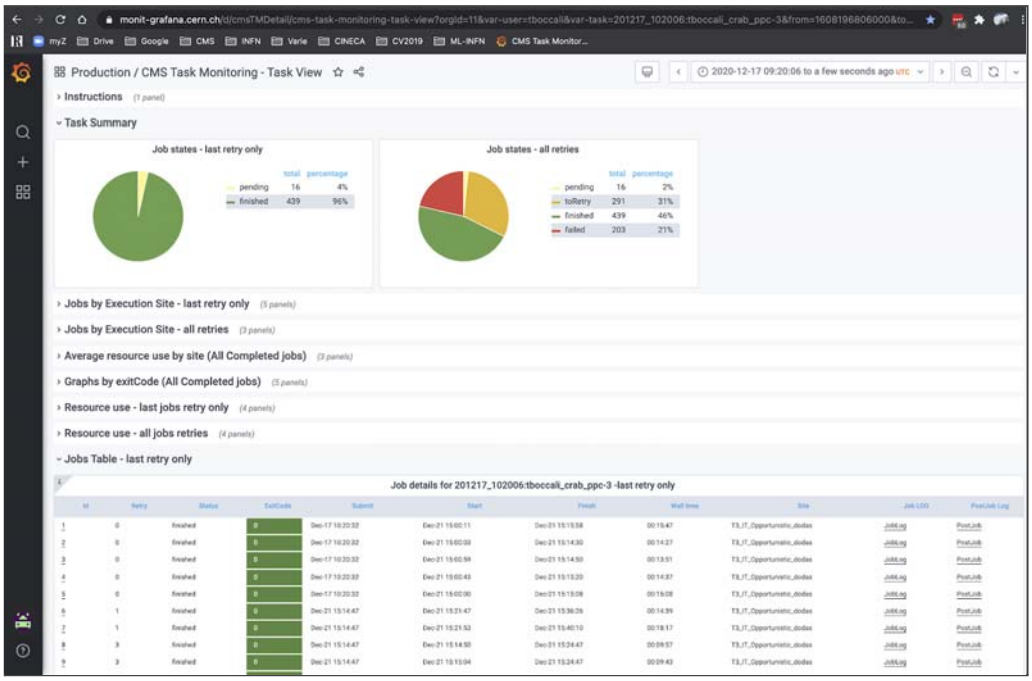

Figure 2. Monitoring Taskview showing the results of two distinct CRAB based run. Right pie chart shows the results before the stageout fix. Left is how the system behaves after the fix.

submission; no show stoppers were identified. The major issue we hit was about stageout step because the stage-out script enforces the use of python2 from a SLC6 setup while our system at M100 is executed on a SLC7 (see Fig. 3).

4.2. Marconi100 integration with CMS Central Processing Infrastructure (WMAgent)

The next step was more complex, and led us to the final integration with CMS WMAgent targeting the implementation of the capability to dispatch production workflows. To this end we deployed a dedicated instance of the WMAgent service which had been configured to send jobs to the Tier-1

CNAF only. The CINECA resources have been made available to CMS as an elastic and transparent extension; indeed Marconi 100 appeared as a Site Extension of Tier-1 CNAF. The added value of this model is twofold: it is a completely transparent mechanism to the CMS Computing Operations; it is a lightweight and dynamic integration of resources from the site perspectives. In both cases this translated into saving operational costs. As explained in paragraph 3, the major modifications needed to enable WMA to dispatch jobs to CINECA were about removing assumptions on the architecture, and instead fetching it from the Workflow configuration.
We used Release Validation [17] workflow to commission the system as well as to provide useful data to carry on with the physics validation, the ultimate goal of this work. We could run those workflows succeeding in all the tasks. Several tests up to a scale of 25k cores were performed (see Fig. 4).

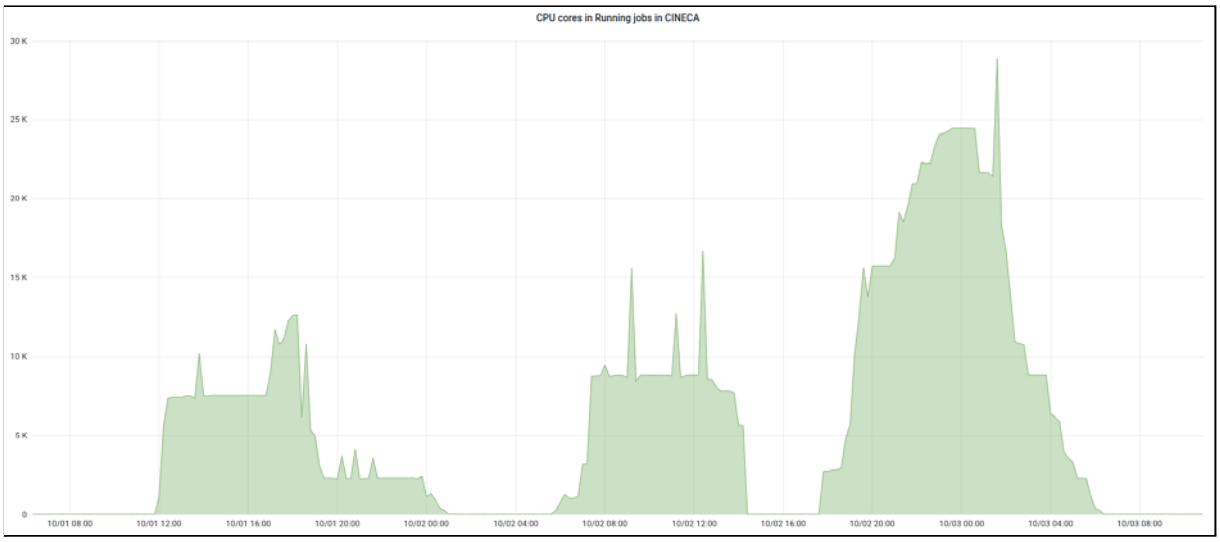

Figure 3. Number of cores exploited on Marconi100 during three processing run. The peak on the right shows the 25kCores reached running WMAgent jobs.

5. Future work

The described work was extremely successful from several perspectives: it allowed to generalize several assumptions in the computing services (such as CRAB and WMAgent) and this resulted in an official release 1.5.4 of WMAgent which includes the complete support to *ppc64le* architecture. This is a key element for CMS to produce official samples and to perform, for the first time, the Physics Validation of results on PPC. The latter is the important objective in our plan, and is already ongoing at the time of writing. The ultimate goal is to eventually enable the use of the samples produced on Marconi100 for CMS approved Physics papers. In order to achieve this goal, 10s of workflows, for a total of some million events, need to be produced and compared with the same results on *x86_64*.
On a more technical side, CINECA is going to be seen as a storageless site which uses T1 CNAF storage resources. Our plan is not to use CINECA for I/O intensive processing steps, since it would result in an inefficient resource usage (while technically working). A further optimization (already tested in private code repositories) is to allow for workflows to be submitted on a list of architectures, [slc7_amd64_gcc9, slc7_ppc64le_gcc9], and then to select (i.e. cherry-picking) at runtime where to execute what, depending on the various I/O capabilities. In the case under test with CINECA/CNAF, all the CPU-intensive workflows would run at CINECA (using *ppc64le* binaries), while the I/O intensive within the CNAF nodes (using *x86_64* binaries). This last step, more in general, will finally enable the multi-architectures support in the Workload Management System of CMS.

References

[1] S. Chatrchyan et al. “The CMS Experiment at the CERN LHC”. In: JINST 3 (2008), S08004. DOI: 10.1088/1748-0221/3/08/S08004
[2] Lyndon Evans and Philip Bryant. “LHC Machine”. In: Journal of Instrumentation 3.08 (2008), S08001
[3] WLCG Homepage: https://wlcg.web.cern.ch/
[4] https://cms-sw.github.io/
[5] CMS-NOTE-2020-002/CMS-NOTE-2020-003
[6] INFN Homepage: https://home.infn.it/en/
[7] CNAF Homepage: https://www.cnaf.infn.it/en/
[8] CINECA Homepage: https://www.cineca.it/en
[9] T.Boccali, S.Dal Pra, D.Spiga et al.”Extension of the INFN Tier-1 on a HPC system” In: EPJ Web Conf. 245 (2020) 09009 DOI: 10.1051/epjconf/202024509009
[10] “G Boudoul” et.al “Monte Carlo Production Management at CMS”, Journal of Physics: Conference Series, Volume 664
[11] D.Spiga.,et al. “A CMS application for distributed analysis”, (2009) IEEE Transactions on Nuclear Science, 56 (5), art. no. 5280527, pp. 2850-2858. DOI: 10.1109/TNS.2009.2028076
[12] HTCondor: https://research.cs.wisc.edu/htcondor
[13] Gregory M. Kurtzer,Vanessa Sochat ,Michael W. Bauer. “Singularity: Scientific containers for mobility of compute”
[14] Jakob Blomer et al. “New directions in the CernVM file system”, 2017 J. Phys.: Conf. Ser. 898 062031, https://doi.org/10.1371/journal.pone.0177459
[15] http://tsocks.sourceforge.net/
[16] Mariotti, M., Spiga, D., Boccali, T. "A possible solution for HEP processing on network secluded computing nodes". In Proceedings of Science 378,002
[17] The CMS Collaboration. Validation of software releases for CMS. J. Phys. Conf. Ser, 219, 2010.

Acknowledgments

The activities were partially supported via the EU Projects ESCAPE (Grant agreement 824064). The integration was possible also thanks to the support made available by INFN-CNAF, and in particular by S. Dal Pra, S. Zani and L. Morganti. The authors and the CMS collaboration as a whole would like to thank CINECA for the opportunity.