E-Infrastructures
H2020-EINFRA-2016-2017

EINFRA-11-2016: Support to the next implementation phase of Pan-European High Performance Computing Infrastructure and Services (PRACE)

PRACE-5IP

PRACE Fifth Implementation Phase Project

Grant Agreement Number: EINFRA-730913

D6.4

Development of Prototypal New Services
Final

Version: 1.0
Author(s): Janez Povh, UL FME
Date: 15.04.2019

Project and Deliverable Information Sheet

PRACE Project	**Project Ref. №: EINFRA-730913**	
	Project Title: PRACE Fifth Implementation Phase Project	
	Project Web Site: http://www.prace-project.eu	
	Deliverable ID: < **D6.4**	
	Deliverable Nature: < Report>	
	Dissemination Level: PU *	**Contractual Date of Delivery:** 15 / 04 / 2019
		Actual Date of Delivery: 30 / 04 / 2019
	EC Project Officer: Leonardo Flores Añover	

* - The dissemination level are indicated as follows: **PU** – Public, **CO** – Confidential, only for members of the consortium (including the Commission Services) **CL** – Classified, as referred to in Commission Decision 2005/444/EC.

Document Control Sheet

Document	**Title: Development of Prototypal New Services**	
	ID: D6.4	
	Version: <1.0>	**Status:** Final
	Available at: http://www.prace-project.eu	
	Software Tool: Microsoft Word 2013	
	File(s): D6.4.docx	
Authorship	**Written by:**	Janez Povh, UL FME
	Contributors:	Agnes Ansari, CNRS/IDRIS Abdulrahman Azab, UiO Simone Bna, CINECA Kyriakos Ginis, GRNET Miroslaw Kupczyk, PSNC Frederic Suter, CNRS / CC-IN2P3
	Reviewed by:	Dimitrios Dellis, GRNET Florian Berberich, JUELICH
	Approved by:	MB/TB

Document Status Sheet

Version	Date	Status	Comments
0.1	5/3/2019	draft	
0.2	13/3/2019	First version	Sent to contributors for updates
0.3	26/3/2019	Second version	Sent to contributors for improvement
0.4	2/4/2019	Version for reviewers	Sent for internal review

0.5	15/4/2019	Intermediate version	Sent to contributors for updates
1.0	16/4/2019	Final	Sent to PRACE 5IP MB for approval

Document Keywords

Keywords:	PRACE, HPC, Research Infrastructure, Urgent Computing, Large Scale Scientific Instruments, Containers, Repositories for Scientific Libraries, Big Data Analysis, In-Situ visualisation, Remote Visualisation, Smart Post-processing.

Table of Contents

Document Control Sheet i
Document Status Sheet i
Document Keywords iii
List of Figures vi
List of Tables vii
References and Applicable Documents vii
List of Acronyms and Abbreviations xi
List of Project Partner Acronyms xii
Executive Summary 1
1 Introduction 1
2 Service 1: Urgent Computing 2
2.1 Motivation 2
2.2 PRACE-RI urgent computing service in Solid Earth 4
2.3 The legal tenders considerations 5
3 Service 2: Link with Large Scale Scientific Instruments (LSSI) 6
3.1. Description of the pilot use cases 6
3.1.1. Enabling ESRF computations on PRACE resources 6
3.1.2. Establishing a formal collaboration with CERN 9
3.2. Evaluation of the pilots 10
3.2.1. Enabling ESRF computations on PRACE resources 10
3.2.2. Enabling CERN computations on PRACE resources 11
3.3. Recommendations for PRACE-6IP 12
4 Service 3: Smart post processing tools including in-situ visualization 13
4.1. Description of the prototypical services 14
4.1.1. The SAIO tool (Large data I/O optimization) 15
4.1.2. In-situ Visualization Service 15
4.1.3. Remote Visualization Service 19
4.2. Experiences with the prototypical services 20
4.2.1. SAIO Tool (Large data I/O optimization) 20
4.2.2. The In-Situ Visualization Service 23
4.2.3. The Remote Visualization Service 24
4.3. Recommendations for the next implementation phase 25

4.3.1. SAIO tool *25*
4.3.2. The In-situ Visualization Service *25*
4.3.3. The Remote Visualization Service *25*
5 Service 4: Provision of repositories for European open source scientific libraries and applications **26**
5.1. Description of the service **26**
5.1.1. Service hosting *27*
5.1.2. Code Repository: GitLab PRACE *27*
5.1.3. Project Management & bug tracking: TRAC *28*
5.1.4. Project Management & bug tracking: Redmine *29*
5.1.5. Continuous integration system: Jenkins *30*
5.1.6. Continuous integration system: GitLab PRACE *30*
5.1.7. Account management: LDAP *30*
5.1.8. Single Sign On to all services: CASino *31*
5.1.9. Documentation *31*
5.2. Service Policy **31**
5.2.1. Access Policies *31*
5.2.2. Authentication policies *32*
5.2.3. Usage and Data policies *32*
5.3. Use of the service **35**
5.4. Security review **35**
6 Service 5: The deployment of containers and full virtualized tools into HPC infrastructures **36**
6.1. Introduction **36**
6.2. Pilot description **36**
6.3. Overall Evaluation **37**
6.4. Site contributions **37**
6.5. Prototypes and use cases **37**
6.5.1. Docker (CNRS/IDRIS & UiO/SIGMA2 & CESGA) *37*
6.5.2. Private Cloud On a Compute Cluster PCOCC *42*
6.5.3. Singularity *45*
6.5.4. GYOC2: Get Your Own Containerised Cluster (UiO/Sigma2) *51*
6.6. Overall Conclusions and future directions **52**
6.6.1. Conclusions *52*

6.6.2. Future directions *53*
7 Service 6: Evaluation of new prototypes for Data Analytics services **54**
7.1. Description of the service **54**
7.2. Description of the prototype services and use cases **55**
7.2.1. Deep Learning SDK, libraries and use cases *55*
7.2.2. Spark tools – The IBM Spectrum Scale support to Hadoop HDFS *57*
7.2.3. Advanced features *58*
7.3. Experiences with the prototype services **59**
7.3.1. Deep learning SDK and libraries *59*
7.3.2. Spark tools *65*
7.3.3. Advanced features *67*
7.4. Conclusion and recommendations for the next implementation phase **68**
7.5. First draft of PKIs **69**
7.5.1. Deep Learning SDK *69*
7.5.2. Data Analytics GitLab *69*
8 Conclusions **70**

List of Figures

Figure 1: SAIO architecture 15
Figure 2: in-situ service architecture 17
Figure 3: Example of a catalyst dictionary for fyMeshInput 19
Figure 4: Work with TiledViz and ANATOMIST on neurobiologist's result 20
Figure 5: Results of SAIO optimization on a read heavy process 21
Figure 6: O/I optimization using SAIO tool on Ansys Fluent use case 22
Figure 7: the LSF/Docker platform architecture 38
Figure 8: PCOCC Networking Model: VM networks with multiple physical hosts 42
Figure 9: PCOCC Infiniband benchmark (packet size vs BW) 44
Figure 10: Relative execution time of parallel benchmarks executed in a cluster of PCOCC VMs compared to the same benchmarks launched on the host cluster 44
Figure 11: Tensorflow training (# images/second) with NVIDIA Tesla K80 using 1,2,3, and 4 GPUs 48
Figure 12: Workflow of Singularity images in the e-Infrastructure 49
Figure 13: STREAM best bandwidth rates (MB/s) comparing Singularity with native run using different benchmarks 50
Figure 14: Latency from Singularity using OSU micro-benchmark 50
Figure 15: Bandwidth from Singularity using OSU micro-benchmarks 50
Figure 16: Get Your Own Container Cluster (GYOC2) Architecture: Compute nodes, resource broker, and the front-end portal are all in Docker containers 52

Figure 17: Basic benchmarks – Tensorflow and Caffe with the trained model GoogLeNet (X-axis: number of images per second [log scale], Y-axis: framework and architecture) and for a varying set of batch size with one GPU only 59
Figure 18: CIFAR-10 benchmark 60
Figure 19: The Astrophysics use case 60
Figure 20: ImageNet - Intra-node model bandwidth 61
Figure 21: ImageNet-Multi-nodes model bandwidth 62
Figure 22: ImageNet - Parallel efficiency over GPUs 63
Figure 23: ImageNet - Parallel efficiency over nodes 63
Figure 24: The X-axis describes the TeraGen and TeraSort benchmarks executed on 1 or 2 nodes. The blue colour indicates that the HDFS connector for GPFS is used, whereas the red colour refers to GPFS and it means that there is a direct access to the GPFS file system without any HDFS connector HDFS block size = 32*GPFS block size 66
Figure 25: The X-axis describes the TeraGen and TeraSort benchmarks executed on 1 or 2 nodes. The blue colour indicates that the HDFS connector for GPFS is used, whereas the red colour refers to GPFS and it means that there is a direct access to the GPFS file system without any HDFS connector. HDFS block size = GPFS block size 66

List of Tables

Table 1: SAIO optimized configuration for IOR benchmark on Cray XC-30 platform 22
Table 2: Optimization results on Lenovo NeXtScale platform for various data transfer sizes 23
Table 3: Optimization results on bullx platform 23
Table 4: Singularity Tensorflow Test results (# images/second) on GALILIO using 1, 2, and 4 GPUs 46
Table 5: Frameworks and libraries - Main features 56
Table 6: Benchmarks description 56
Table 7: HPC resources 57
Table 8: TensorFlow - Pro and cons 64
Table 9: Caffe - Pro and cons 64
Table 10: Keras - Pro and cons 65
Table 11: Horovod - Pro and cons 65

References and Applicable Documents

[1] PRACE-5IP, “PRACE-5IP D6.3 Analysis of New Services,” 2017.

[2] PRACE-4IP, “PRACE-4IP D6.4 Deployment of Prototypal New Services,” 2017.

[3] Kupczyk, M.,Kaliszan, D., Stoffers, H., Wilson, N., Moll, F. (2017). Urgent Computing service in the PRACE Research Infrastructure.

[4] Wang, X. (2017). A light weighted semi-automatically I/O-tuning solution for engineering applications.

[5] Klasky, S., Abbasi, H., Logan, J., Parashar, M., Schwan, K., Shoshani, A., ... & Chacon, L. (2011). In situ data processing for extreme-scale computing. *Proceedings of SciDAC*.

[6] Ayachit, U., Bauer, A., Geveci, B., O'Leary, P., Moreland, K., Fabian, N., & Mauldin, J. (2015, November). Paraview catalyst: Enabling in situ data analysis and visualization. In *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization* (pp. 25-29). ACM.

[7] "Paraview," [Online]. Available: http://www.paraview.org.

[8] "Openfoam," 2018. [Online]. Available: https://www.openfoam.com/.

[9] "Catalyst project," [Online]. Available: https://develop.openfoam.com/Community/catalyst.

[10] Bnà, S., Olesen, M., & Bauer, A. In situ data analysis and visualization in OpenFOAM with ParaView Catalyst.

[11] Mancip, M., Spezia, R., Jeanvoine, Y., & Balsier, C. (2018). TileViz: Tile visualization for direct dynamics applied to astrochemical reactions. *Electronic Imaging*, *2018*(16), 286-1.

[12] Simulation of Fluid-Particle Interaction in Turbulent Flows," Gauss Centre for Supercomputing, November 2015. [Online]. Available: http://www.gauss-centre.eu/gauss-centre/EN/Projects/CSE/2015/Meinke_AIA.html?nn=1345710.

[13] Ye, Q., & Tiedje, O. (2016). Investigation on Air Entrapment in Paint Drops Under Impact onto Dry Solid Surfaces. In *High Performance Computing in Science and Engineering´ 16* (pp. 355-374). Springer, Cham.

[14] "GitLab," [Online]. Available: https://about.gitlab.com/.

[15] "The Trac Project," [Online]. Available: https://trac.edgewall.org/.

[16] "Redmine: Overview," [Online]. Available: https://www.redmine.org/.

[17] "Jenkins," [Online]. Available: https://jenkins.io/.

[18] "CASino," [Online]. Available: http://casino.rbcas.com/.

[19] Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, *2014*(239), 2.

[20] Kurtzer, G. M. Singularity 2.1. 2-Linux application and environment containers for science, 2016. *Available from Internet:< https://doi. org/10.5281/zenodo, 60736*.

[21] "Environmentalomics," [Online]. Available: http://environmentalomics.org/bio-linux/ .

[22] "Post-installation steps fopr Linux," [Online]. Available: https://docs.docker.com/engine/installation/linux/linux-postinstall/.

[23] "Docker security," [Online]. Available: https://docs.docker.com/engine/security/security/#docker-daemon-attack-surface.

[24] Yoo, A. B., Jette, M. A., & Grondona, M. (2003, June). Slurm: Simple linux utility for resource management. In *Workshop on Job Scheduling Strategies for Parallel Processing* (pp. 44-60). Springer, Berlin, Heidelberg.

[25] Azab, A. (2017, April). Enabling docker containers for high-performance and many-task computing. In *2017 ieee international conference on cloud engineering (ic2e)* (pp. 279-285). IEEE.

[26] "The Abel computer cluster," [Online]. Available: http://www.uio.no/english/services/it/research/hpc/abel/ .

[27] "Colossus: Sensitive data cluster," [Online]. Available: https://www.uio.no/english/services/it/research/sensitive-data/use-tsd/hpc/.

[28] "Elixir," [Online]. Available: https://www.elixir-europe.org/.

[29] "NeIC/Tryggve," [Online]. Available: https://neic.no/tryggve/ .

[30] "Udocker GitHub," [Online]. Available: https://github.com/indigo-dc/udocker .

[31] "uDocker with MPI:," [Online]. Available: https://github.com/indigo-dc/udocker/blob/master/doc/user_manual.md#4-running-mpi-jobs.

[32] "PCOCC," [Online]. Available: https://pcocc.readthedocs.io.

[33] Kurtzer, G. M., Sochat, V., & Bauer, M. W. (2017). Singularity: Scientific containers for mobility of compute. *PloS one*, *12*(5), e0177459. [Online]. Available: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0177459 .

[34] "Singularity web site," [Online]. Available: https://www.sylabs.io/ .

[35] "PBS web site," [Online]. Available: https://www.pbspro.org/.

[36] "Slurm web site," [Online]. Available: https://slurm.schedmd.com/.

[37] "Docker Hub web site," [Online]. Available: https://hub.docker.com/ .

[38] "Singularity Hub web site," [Online]. Available: http://singularity-hub.org/ .

[39] "GALILEO cluster at CINECA," [Online]. Available: https://wiki.u-gov.it/confluence/display/SCAIUS/UG3.3%3A+GALILEO+UserGuide .

[40] "Tensorflow web site," [Online]. Available: https://www.tensorflow.org/ .

[41] "Official Tensorflow Performance benchmarks," [Online]. Available: https://www.tensorflow.org/guide/performance/benchmarks.

[42] "Official Tensorflow Docker Hub web site," [Online]. Available: https://hub.docker.com/r/tensorflow/tensorflow/.

[43] "User Guide on CINECA," [Online]. Available: https://wiki.u-gov.it/confluence/display/SCAIUS/Container .

[44] "Singularity at UiO" [Online]. Available: https://www.uio.no/english/services/it/research/hpc/abel/help/software/singularity.html

[45] "Caffe web site," [Online]. Available: http://caffe.berkeleyvision.org

[46] "Keras web site," [Online]. Available: https://keras.io

[47] "Uber Engineering web site," [Online]. Available: https://eng.uber.com/horovod

[48] CINECA, Marco Rorro –. *Basic Benchmarks Documentation and Results.*

[49] EPCC, Andreas Vroutsis. *Basic Benchmarks Documentation and Result*

[50] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems* (pp. 1097-1105).

[51] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 1-9).

[52] Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*.

[53] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

[54] B. Rigaud, "https://gitlab.in2p3.fr/brigaud/prace-keras-cifar10-benchmark," [Online].

[55] CNRS/IDRIS, Alberto Garcia Fernandez –. *ILSVRC2012 Use Case Documentation and Results.*

[56] Simonyan, K., & Zisserman, A. Visual Geometry Group, Department of Engineering Science,University of Oxford. Very Deep Convolutional networks for large scale image recognition. ICLR 2015

[57] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 770-778).

[58] CNRS/IDRIS, Agnès Ansari. *Call for Prototypes - https://bscw.zam.kfajuelich.de/bscw/bscw.cgi/d2599979/CallForPrototypes.doc* .

[59] CNRS/IDRIS, Agnès Ansari. *The Use cases Master document - https://bscw.zam.kfajuelich.de/bscw/bscw.cgi/d2640846/UseCasesMasterDocument.docx.*

[60] B. Rigaud, "https://gitlab.in2p3.fr/brigaud/prace-ramp-astro-benchmark/tree/v1.0," [Online].

[61] Ronneberger, O., Fischer, P., & Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.

[62] CNRS/IDRIS, Agnès Ansari. *The DataSet download service specifications - https://bscw.zam.kfa-juelich.de/bscw/bscw.cgi/d2684726/DataSet-Download-ServiceV1.0.pdf* .

[63] CINECA, Marco Rorro –. *The Dataset download service design document for use by the PRACE Data Analytics Servic*

[64] "iRods web site," [Online]. Available: https://irods.org,

[65] "EUDAT," [Online]. Available: https://www.eudat.eu/b2safe

[66] Agnès Ansari - CNRS/IDRIS, Alberto Garcia Fernandez CNRS/IDRIS, Bertrand Rigaud CNRS/CC-IN2P3, Marco Rorro CINECA, Andreas Vroutsis EPCC. *The PRACE Data Analytics service.*

List of Acronyms and Abbreviations

aisbl	Association International Sans But Lucratif (legal form of the PRACE-RI)
BCO	Benchmark Code Owner
CoE	Centre of Excellence
CPU	Central Processing Unit
CUDA	Compute Unified Device Architecture (NVIDIA)
DARPA	Defence Advanced Research Projects Agency
DEISA	Distributed European Infrastructure for Supercomputing Applications EU project by leading national HPC centres
DoA	Description of Action (formerly known as DoW)
EC	European Commission
EESI	European Exascale Software Initiative
EoI	Expression of Interest
ESFRI	European Strategy Forum on Research Infrastructures
GB	Giga (= 2^{30} ~ 10^9) Bytes (= 8 bits), also GByte
Gb/s	Giga (= 10^9) bits per second, also Gbit/s
GB/s	Giga (= 10^9) Bytes (= 8 bits) per second, also GByte/s
GÉANT	Collaboration between National Research and Education Networks to build a multi-gigabit pan-European network. The current EC-funded project as of 2015 is GN4.
GFlop/s	Giga (= 10^9) Floating point operations (usually in 64-bit, i.e. DP) per second, also GF/s
GHz	Giga (= 10^9) Hertz, frequency =10^9 periods or clock cycles per second
GPU	Graphic Processing Unit
HET	High Performance Computing in Europe Taskforce. Taskforce by representatives from European HPC community to shape the European HPC Research Infrastructure. Produced the scientific case and valuable groundwork for the PRACE project.
HMM	Hidden Markov Model
HPC	High Performance Computing; Computing at a high performance level at any given time; often used synonym with Supercomputing
HPL	High Performance LINPACK
ISC	International Supercomputing Conference; European equivalent to the US based SCxx conference. Held annually in Germany.
KB	Kilo (= 2^{10} ~10^3) Bytes (= 8 bits), also KByte
LINPACK	Software library for Linear Algebra
MB	Management Board (highest decision making body of the project)
MB	Mega (= 2^{20} ~ 10^6) Bytes (= 8 bits), also MByte
MB/s	Mega (= 10^6) Bytes (= 8 bits) per second, also MByte/s
MFlop/s	Mega (= 10^6) Floating point operations (usually in 64-bit, i.e. DP) per second, also MF/s
MOOC	Massively open online Course
MoU	Memorandum of Understanding.
MPI	Message Passing Interface

NDA	Non-Disclosure Agreement. Typically signed between vendors and customers working together on products prior to their general availability or announcement.
PA	Preparatory Access (to PRACE resources)
PATC	PRACE Advanced Training Centres
PRACE	Partnership for Advanced Computing in Europe; Project Acronym
PRACE 2	The upcoming next phase of the PRACE Research Infrastructure following the initial five year period.
PRIDE	Project Information and Dissemination Event
RI	Research Infrastructure
TB	Technical Board (group of Work Package leaders)
TB	Tera (= 2^{40} ~ 10^{12}) Bytes (= 8 bits), also TByte
TCO	Total Cost of Ownership. Includes recurring costs (e.g. personnel, power, cooling, maintenance) in addition to the purchase cost.
TDP	Thermal Design Power
TFlop/s	Tera (= 10^{12}) Floating-point operations (usually in 64-bit, i.e. DP) per second, also TF/s
Tier-0	Denotes the apex of a conceptual pyramid of HPC systems. In this context the Supercomputing Research Infrastructure would host the Tier-0 systems; national or topical HPC centres would constitute Tier-1
UNICORE	Uniform Interface to Computing Resources. Grid software for seamless access to distributed resources.

List of Project Partner Acronyms

BADW-LRZ	Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften, Germany (3rd Party to GCS)
BILKENT	Bilkent University, Turkey (3rd Party to UYBHM)
BSC	Barcelona Supercomputing Center - Centro Nacional de Supercomputacion, Spain
CaSToRC	Computation-based Science and Technology Research Center, Cyprus
CCSAS	Computing Centre of the Slovak Academy of Sciences, Slovakia
CC-IN2P3	Computing Centre of French National Institute of Nuclear and Particle Physics
CEA	Commissariat à l'Energie Atomique et aux Energies Alternatives, France (3 rd Party to GENCI)
CESGA	Fundacion Publica Gallega Centro Tecnológico de Supercomputación de Galicia, Spain, (3rd Party to BSC)
CINECA	CINECA Consorzio Interuniversitario, Italy
CINES	Centre Informatique National de l'Enseignement Supérieur, France (3 rd Party to GENCI)
CNRS	Centre National de la Recherche Scientifique, France (3 rd Party to GENCI)
CSC	CSC Scientific Computing Ltd., Finland
CSIC	Spanish Council for Scientific Research (3rd Party to BSC)
CYFRONET	Academic Computing Centre CYFRONET AGH, Poland (3rd party to PNSC)
EPCC	EPCC at The University of Edinburgh, UK
ETHZurich (CSCS)	Eidgenössische Technische Hochschule Zürich – CSCS, Switzerland
GCS	Gauss Centre for Supercomputing e.V.

GENCI	Grand Equipement National de Calcul Intensiv, France
GRNET	Greek Research and Technology Network, Greece
IDRIS	Institut du Developpement et des Ressources en Informatique Scientifique
INRIA	Institut National de Recherche en Informatique et Automatique, France (3rd Party to GENCI)
IST	Instituto Superior Técnico, Portugal (3rd Party to UC-LCA)
IUCC	INTER UNIVERSITY COMPUTATION CENTRE, Israel
JKU	Institut fuer Graphische und Parallele Datenverarbeitung der Johannes Kepler Universitaet Linz, Austria
JUELICH	Forschungszentrum Juelich GmbH, Germany
KTH	Royal Institute of Technology, Sweden (3rd Party to SNIC)
LiU	Linkoping University, Sweden (3rd Party to SNIC)
NCSA	NATIONAL CENTRE FOR SUPERCOMPUTING APPLICATIONS, Bulgaria
NIIF	National Information Infrastructure Development Institute, Hungary
NTNU	The Norwegian University of Science and Technology, Norway (3rd Party to SIGMA)
NUI-Galway	National University of Ireland Galway, Ireland
PRACE	Partnership for Advanced Computing in Europe aisbl, Belgium
PSNC	Poznan Supercomputing and Networking Center, Poland
RISCSW	RISC Software GmbH
RZG	Max Planck Gesellschaft zur Förderung der Wissenschaften e.V., Germany (3rd Party to GCS)
SIGMA2	UNINETT Sigma2 AS, Norway
SNIC	Swedish National Infrastructure for Computing (within the Swedish Science Council), Sweden
STFC	Science and Technology Facilities Council, UK (3rd Party to EPSRC)
SURFsara	Dutch national high-performance computing and e-Science support center, part of the SURF cooperative, Netherlands
UC	Urgent Computing
UC-LCA	Universidade de Coimbra, Labotatório de Computação Avançada, Portugal
UCPH	Københavns Universitet, Denmark
UHEM	Istanbul Technical University, Ayazaga Campus, Turkey
UiO	University of Oslo, Norway (3rd Party to SIGMA)
ULFME	UNIVERZA V LJUBLJANI, Slovenia
UmU	Umea University, Sweden (3rd Party to SNIC)
UnivEvora	Universidade de Évora, Portugal (3rd Party to UC-LCA)
UPC	Universitat Politècnica de Catalunya, Spain (3rd Party to BSC)
UPM/CeSViMa	Madrid Supercomputing and Visualization Center, Spain (3rd Party to BSC)
USTUTT-HLRS	Universitaet Stuttgart – HLRS, Germany (3rd Party to GCS)
VSB-TUO	VYSOKA SKOLA BANSKA - TECHNICKA UNIVERZITA OSTRAVA, Czech Republic
WCNS	Politechnika Wroclawska, Poland (3rd party to PNSC)

Executive Summary

In this deliverable, we present results obtained in Task 6.2 of Work Package 6 of the PRACE-5IP project. This task was focused on six services that had potential to address some of the widely recognised needs in scientific computing. Four services were already investigated within the preceding project PRACE-4IP: (1) the provision of urgent computing services, (2) links to large-scale scientific instruments (i.e. satellites, laser facilities, sequencers, synchrotrons, etc.), (3) smart post processing tools including in-situ visualization, and (4) provisioning of repositories for European open source scientific libraries and applications.

Additionally, we have analysed within PRACE-5IP project two new services: (5) Evaluation of lightweight virtualisation technology and (6) Evaluation of new prototypes for Data Analytics services.

At the beginning of this project we have decided that PRACE-4IP brought service 4 to a level that is ready for migration to a regular PRACE service, so the team working on this service focus to activities needed for migration. For the other three services inherited from PRACE-4IP we have decide to continue evaluation and testing and results are reported in this document.

For the two new services we have created competent teams from partner institutions that followed the operational plans prepared in deliverable D6.3 [1] and have done a long list of activities which have resulted in many test scripts, benchmark results, dissemination and training activities and are described in this document.

The deliverable has therefore six main sections – one for each service. Each section contains description of the service, the list of planned activities, report of the work done within PRACE-5IP and final conclusion based on the results of the activities. This document is therefore also a basis for work in Task 6.2 of PRACE-6IP and will help the partners to decide which services are matured enough to be upgraded into regular services, which will remain in a pilot phase and which will be closed.

1 Introduction

An efficient and state-of-the-art pre-exascale HPC infrastructure at European level should be ready to operate innovative services to address scientific, technological and societal challenges. In Task 6.2 of project PRACE-5IP we have examined six services. Following the interest of project partners with person months (PMs) assigned in WP6 Task 6.2, six groups of partners – one for each new service – were defined. The groups were coordinated by ULFME, which was Task 6.2 coordinator. They were composed of:

- Service 1: The provision of urgent computing services: the coordinator was PSNC and another partner involved in the task was NCSA;
- Service 2: The link with large-scale scientific instruments: the coordinator of this task was CNRS / CC-IN2P3 and other partners involved in the task were: CaSToRC and NCSA.

- Service 3: Smart post processing tools including in-situ visualisation: this task was coordinated by CINECA and the other partners involved in the task were: HLRS and CEA/DRF,
- Service 4: Provisioning of repositories for European open source scientific libraries and applications: this task was coordinated by GRNET and the other partners involved in the task were: CESGA and NIIF.
- Service 5: Evaluation of lightweight virtualisation technology: the service coordinator was UiO, while the contributors were CEA/DAM, CNRS-IDRIS, CINECA, EPCC (observer) and CESGA (observer).
- Service 6: Evaluation of new prototypes for Data Analytics services: this service was coordinated by CNRS/IDRIS and the contributors were CNRS/CC-IN2P3, EPCC, KTH, CINECA, PSNC.

In Deliverable D6.3 we have described operational plans for each service, related work and different related challenges. In this deliverable, we present for each service the prototype(s) that was (were) developed for each service by Month 28. More precisely, for each service we:

- Describe the service;
- Describe the pilots (experiments): we first describe the common point of all pilots (experiments) within the service (for those services where there is more than 1 pilot) and then specifics of each of them;
- Evaluate the pilot(s)/experiments and present the results. We point out positive and challenging experiences;
- Based on evaluation we also make propositions:
  - How shall the service in PRACE-6IP be continued;
  - We propose key points of the policy, if a PRACE policy is needed;
  - We propose the first draft of KPI's;

During the finalization of Task 6.2 we have collected important technical details also in several white papers, which are in the process of internal quality control and will be available on PRACE web page.

2 Service 1: Urgent Computing

2.1 Motivation

The work on Urgent Computing service was conducted due to the prospective and possible usage of PRACE Infrastructure in the future. It is foreseen to include this service into PRACE core service list in case of the agreement with the PRACE external users. The detailed description of the service was included into deliverable D6.4 [2] of PRACE-4IP and the policy design document – Whitepaper [3].

We have investigated several possible scenarios and made the trade-off proposal on the PRACE-RI usage in Urgent Computing (UC) scenario. The ordinary use case must include the corresponding parties: Public Service Authority as the ordering party, PRACE Authority as the

supplier of the service; UC User, AAA Management in PRACE, Urgent Application and the data source. The minimal agreement can state that the data staging is assured by the UC user itself directly on selected PRACE machine (without Instrument / Storage engagement at urgent time).

The operational platform supporting UC case implies that the systems are each in a state of "warm standby" for the real event. To be, and remain, in a state of warm standby:

- All needed application software must be pre-installed;
- One or more data sets suitable for validation must be pre-installed;
- There must be a validation protocol using pre-installed software and pre-installed data;
- Runs to execute the validation protocol must be scheduled regularly to verify that the applications keep performing as they should, especially after system changes, software upgrades on general purpose and computational libraries, etc;
- Validation runs can be regular batch jobs. A budget of sufficient core hours to perform these jobs regularly must be allocated;
- Since pre-installed software and data may be damaged by human error, or hardware malfunctioning, or even the above noted updating of other software components, there should also be regularly tested procedure to quickly restore – reinstall, relink, recompile, reconfigure, whatever applies – the pre-installed components;

While validation of the software can blend in with the regular production environment, the workflow of a real event will inevitably need more:

- Platforms supporting a UC case must have a mechanism to raise the emergency flag, which can be executed by a preselected class of users: UC users, UC operators;
- The raising of the emergency flag must enable the availability of the required compute and storage resources in due time;
- What the exact amount of required compute and storage resources actually is, and what exactly is due time, is of course project dependent, but should be agreed upon in advance by the involved parties supplying and using the platform;
- How the freeing of adequate resources in due time is implemented should be at the discretion of the platform supplying side. Some sides may want to use pre-emption of already running jobs, others may e.g. have a large enough dedicated partition for jobs with a fairly short wall clocks time that they can drain from regular job usage;
- Complete workflow scenarios, including the actual making available of resources in due time, must be regularly practiced as "dry runs" as well;
- The regular testing of workflow scenarios is potentially much more disruptive of normal production work than the above mentioned regular software validation and their frequency must be agreed upon in advance, at the intake of a UC project. The budget in core hours allocated for a UC project should be sufficient to cover the actual loss of economic capacity resulting from the agreed upon level of workflow scenario dry runs.

The aforementioned precautions and statement will be investigated when put into the legal tender between PRACE and external Urgent User (Institution). There is no such tender nor the template of such tender available. Due to the merit of this work which is non-technical at this stage, PRACE-5IP decided to wait for the EC review outcome in 2017 indeed. It was discussed during Work

Package technical meeting and the official statement was presented on the PRACE-5IP All-hands meeting in 2017.

The PRACE-4IP reviewers suggested to seek political support which has been elaborated in PRACE-5IP. The future steps include the seeking of political support from PRACE managerial bodies which will address the reviewers' comment:

"Particularly the urgent computing is recognized as important for society. Although technically feasible, it faces a number of challenges which are beyond the current scope and power of decision of the PRACE network (e.g. urgent cases definition, tackling financial and personnel resources for system availability and software maintenance, requirement of institutional access, etc.). Therefore, respective efforts to seek political support may be worth to be continued and expanded towards a global leadership"

Till the end of project duration, the Urgent Computing Tier-0 case has not been identified. However, PRACE aims to co-operate with external bodies, like Centre of Excellences in terms of the tightening the HPC landscape and scientific environment. In PRACE-5IP we have started the joint collaboration with CoE 'ChEESE' in the area of UC.

2.2 PRACE-RI urgent computing service in Solid Earth

PRACE-5IP has started the piloting co-operation with CoE: Center of Excellence for Exascale in Solid Earth (ChEESE), ID: 823844, started in 1. November 2019. The development of UC services should be based on the European open science and FAIR data policy with matured and operational open codes and use cases including all the data and logistics required up to the distilling of the results in forms than can be used for expert opinions and decision making.

This should involve co-design and co-development between ChEESE, PRACE-IP projects and the end users. The ChEESE CoE would consider "urgent" simulations as possible use cases within a testing phase to identify and assess the required services, resource management and policy access in relation with different workflow patterns and data logistics, while checking the feasibility of effectively contributing to future emergencies.

The Center of Excellence for Exascale in Solid Earth (ChEESE) has a special focus on developing pilots and services that will make use of urgent supercomputing with disaster resilience purposes. The main use case pilots are the following:

Near real-time seismic scenarios

By including full 3D physical models and topography, the level of details that simulations can attain and the quantities of interest that can be inferred (e.g. shaking time, peak ground acceleration, and response spectral values at each point on the surface) are highly valuable to analyse the outcome of earthquakes with high resolution. If a solution can be attained at time spans that are sufficient for disaster management (i.e. hours), urgent computing seismic scenario simulations might become useful seismic resilience tools.

High-resolution volcanic forecasting

Model data assimilation has the potential to improve ash dispersal forecasts by an efficient joint estimation of the (uncertain) volcanic source parameters and the state of the ash cloud. ASHEE 3d simulation of multiphase multicomponent volcanic plumes requires of 1024 CPU cores on a 10^7 grid cell. The computations will be considered to deliver hourly solutions when expected.

Faster than real-time tsunami simulations

FTRT tsunami computations are crucial in the context of Tsunami Early Warning Systems (TEWS) and in the context of post-disaster management. Greatly improved and highly efficient computational methods are the first raw ingredient to achieve extremely fast and effective calculations. HPC facilities have the role to bring this efficiency to a maximum while drastically reducing computational times. Possibly and in addition, inputs from an urgent seismic simulation can be exploited for physics-based urgent tsunami simulations. A typical case of probabilistic tsunami forecasting would use from thousands to tens of thousands tsunami simulations for different realizations of the parameters describing the causative source.

2.3 The legal tenders considerations

It is noticeable, that we have not been able to identify any existing legal tender of Tier-0 size usage in Urgent Computing mode. Since there has not been defined the legal tender template of Urgent Computing co-operation, we decided to start with the piloting use case implementation basing on the current available computational PRACE resources.

LRZ investigated the topic deeply and they haven't found a good example of use-case strictly required to be run on Tier-0 by now. Indeed, so far, there has not been identified a convincing use case for the necessity of strict usage Tier-0 for Urgent Computing scenario. In most cases coarse grain scenarios are fully sufficient for emergency services and these can be computed on much smaller computers. In seismology emergency, services use 2D shake maps, in weather prediction there are precomputing scenarios. One would rather apply Model Order Reduction methods than starting a full 3D coupled simulation in an emergency case.

GENCI reported that on their Tier-0 machine there is deployed an Urgent Computing/on Demand Computing mechanism but the detailed information is not available. There is no evidence that any other PRACE (neither hosting nor general) partner has got the legal tender for Urgent Computing service with external consumer of such service.

The co-operation with ChEESE has been already started and it is becoming more realistic that the real UC application will run on Tier-0.

Due to the mature state of the project (ChEESE) development, the thorough specification of the UC application requirements is not available yet. The project-domestic site is BSC and some of the pilots' setup will be done on MareNostrum4 system in BSC. They will also share PRACE hours devoted to CoEs. PSNC has offered the pilot environment on EAGLE (Tier-1) system as well. The scientific ambition of ChEESE is to prepare 10 flagship codes to address Exascale Computing Challenging problems on computational seismology, magnetohydrodynamics, physical volcanology, tsunamis, and data analysis and predictive techniques for earthquake and volcano monitoring. The codes will be audited and optimized at both intranode level (including

heterogeneous computing nodes) and internode level on Exascale architecture hardware prototypes.

The end of date of ChEESE is 31 October 2021 overlapping PRACE-6IP timeline makes the UC scenario deployment highly possible on Tier-0 and Tier-1.

3 Service 2: Link with Large Scale Scientific Instruments (LSSI)

Natural science discovery is currently driven by the analysis of tremendous amounts of data produced by Large Scale Scientific Instruments (LSSI). These instruments, and the produced data, are usually shared among large international scientific groups or collaborations. Easing the access to HPC resources and services to these partnerships calls for direct formal agreements between PRACE and the institutions hosting the instruments. In order to be useful the experimental results require post-processing, analysis, and visualization, all of which may require large computational power and thus cannot be analysed in practice without the use of large HPC facilities and advanced software tools. Moreover, experiments are often coupled with numerical simulations, which also require substantial computational power, which can also generate a comparable amount of data. In this sense HPC facilities and efficient numerical software together can be viewed as an instrument of their own, and are of fundamental importance in the process of validation and refinement of physical models.

The analysis of the experimental results requires access to compute power, which is often beyond what is hosted on the instrumental site itself. Therefore, there is a need for access to external HPC facilities that could offer some CPU time. Operators and related scientists may have to regularly perform certain computational tasks, which are large in volume, but routine in nature. Thus, they do not qualify for resource allocation under PRACE Project Access calls, but the computational work is nonetheless decisive for the research. Service 2 aims at improving the link between large-scale scientific instruments and the PRACE HPC infrastructure with additional involvement of GEANT by addressing the question of a formal institutional access granted to the institutions hosting such LSSI to HPC resources they do not own using best possible internet network. The involved partners, CNRS/CC-IN2P3, NCSA, and CASTORC, have a history of collaboration with a different scientific community with different goals and using a different LSSI. This provides a unique opportunity to draw a more generic partnership framework.

3.1. Description of the pilot use cases

3.1.1. Enabling ESRF computations on PRACE resources

Synchrotrons are circular particle accelerators, which accelerate electrons or positrons (rare) to produce synchrotron radiation. The European Synchrotron Radiation Facility (ESRF), is the world's most intense X-ray source. It is located in Grenoble, France, and it is supported and shared by 21 countries. The ESRF acts as a "super-microscope" to reveal the structure of matter in a very wide range of fields, such as chemistry, material physics, palaeontology, archaeology and cultural heritage, structural biology and medical applications, environmental sciences, information science, and nanotechnologies. With 6,000 users from all over Europe each year, the ESRF produces around

a TB of data per hour, almost 24 hours a day. Three main types of computations are associated to the produced data. Before data acquisition, simulations are run by users to define the set of parameters of the experiment.

While the experiment is running, close to real-time analysis are needed to control the quality of the experiment and steer it as it is happening. Once the experiment is over, raw data is stored for 50 days on an externally accessible storage facility and available for post-processing. For some experiments this post-processing is composed of two parts: a 2D or 3D reconstruction of the sample and then a domain specific analysis of the observations. For other experiments, only the latter is necessary. About 30% of the computing at ESRF is related to simulations performed before data acquisition. It encompasses a set of codes depending on the type of experiments (e.g., spectroscopy, radiation, quantum mechanics) whose executions can be grouped either by scientific project as a single user usually launches several simulations with different parameter sets or by code, as in a service offer. These codes only show limited scalability but may require a large amount of memory.

After a joint PRACE-GEANT-ESRF meeting held in Grenoble on 11 September 2017, Service 2 sent a form about application requirements to ESRF in order to identify which application(s) could be candidate for the implementation of the pilot. This questionnaire asked to provide information about programming languages and dependencies on external libraries, the typical duration and scale of a production, and the size of input and output data for a run. We received five answers to this request on 15 October 2017 which were classified as follows:

- Two « HPC-friendly » codes
  - Written in Fortran + OpenMP/MPI
  - Rely on external libraries (linear algebra, graph partitioning, ...)
  - Execution times are problem dependent: from a few hours to several weeks
  - Production runs use up to 128 cores.
- Two « parallel python » codes
  - Written in Python + MPI
  - Rely on third-party HPC library (petsc, ...) or python modules
  - One application takes no input and runs for 72h on 84 cores
  - The other has a 100GB input and the size of the production can be adapted
- One « GPU » code
  - Written in Python + CUDA/OpenCL
  - Requires several python third-party modules
  - Current production runs length for 40h on 32 P40 GPUs
  - Takes thousands of HDF5 files as input for a total amount of 300MB

After approval the Management Board (PRACE-5IP-MB-2017-03-d06), Service 2 issued a call for volunteering Tier-1 centres to get access to resources for the implementation of the pilot, based on the description above of the candidate applications to run. Volunteering centres were asked to provide access to a maximum of 256 cores for up to two months from January to March 2018, which roughly amounts to a global estimate of half a million core-hours. Four Tier-1 centres

answered to this call and CASTORC offered to serve as a backup, if needed. The characteristics of the offered resources are the following:

- PNSC (Poland), EAGLE cluster:
  - 1226 Intel Xeon E5-2697 v3, E5-2682 v4
  - RAM: 64GB - 256GB per node.
  - Interconnect: Infiniband FDR
  - Storage: Lustre, GPFS.
  - OS: Scientific Linux CERN SLC release 6.9
  - SLURM queueing system.
- MPCDF (Germany), Hydra cluster:
  - 338 Intel Xeon Ivy Bridge 676 NVIDIA K20X GPGPUs
  - RAM: 64GB per node
- Cyfronet (Poland), Prometheus cluster:
  - Intel Xeon (Haswell) -- 2.4 PFlops
  - RAM: 279 TB
  - Storage: 10PB
  - OS: CentOS 7
- CINES (France), Occigen cluster:
  - Intel Xeon E5-2690 V4@2.6GHz
  - RAM: 2.6GB per core
  - OS: BullX SCS6 (Redhat 7.3)
  - Interconnect: Infiniband FDR
  - Storage: Lustre
  - SLURM queueing system

Scientists from ESRF chose to ask for account openings on the Eagle (PSNC) and Hydra (MPCDF) clusters in February 2018. In addition to these resources available through the call for volunteering Tier-1 centres, ESRF also conducted some tests on the Ouessant OpenPower prototype at IDRIS, France. This machine is composed 12 Power8+ nodes with 4 Nvidia P100 GPUaccelerators each. At PSNC, the objective was to run a code that reconstructs near-field ptychography (NFP) projections from the data that acquired at the ID16A line at ERSF. This code relies on the PtyPy Python library, in which the ESRF provides upstream contributions. The ideal result of a production run of this code would be to compute about 1,000 projections per dataset, which would allow scientists to reconstruct a 3D volume. However, the time to compute a single projection is about 24 hours using 10 cores and MPI, making the use for HPC resources crucial. At MPCDF, ESRF aimed at running the PyNX code, developed at ESRF, for Coherent X-ray Imaging techniques. It is written using Python, OpenCL and CUDA, with all calculations occurring on a GPU. It is used to reconstruct real-world objects from diffraction data. Some reconstructions require combining thousands of images, and are naturally parallel. 3D reconstructions can further take advantage of clusters by reconstructing independently a few hundreds of 2D projections, thus using a consequent number of GPU in parallel.

3.1.2. Establishing a formal collaboration with CERN

The Large Hadron Collider (LHC) – being the world's largest and most powerful particle collider, and the one of the most complex experimental facility ever built – at CERN produces data used by a variety of applications ranging from high energy physics to hadron therapy for cancer treatment. An essential part of the data analysis in all particle-matter interaction considerations are Monte Carlo simulations. Physical events are generated by numerical software using a theoretical model, with a complete set of detector parameters as an input and a set of final-state particles as an output. Results of the extremely time and memory consuming numerical simulations are compared with the data gathered during experiments in order to validate and refine physical models.

The upcoming upgrade of the LHC (High Luminosity LHC) will dramatically increase the demand related to the processing and storage of data. This requires a drastic change in the computing models to go beyond the used of commodity clusters accessed through the Worldwide LHC Computing Grid (WLCG). The simulation, reconstruction, and analysis codes have to evolve towards HPC to be able to fully exploit modern CPUs and accelerators on the available resources and harness HPC facilities such as those provided by PRACE. The main experiments already use HPC resources as part of their computing model but usually follows an opportunistic approach by scavenging cycles through the backfilling mechanism. However, most of the LHC codes are not HPC-ready due to their complexity. This calls for a training of developers within the physics collaborations that PRACE could offer on demand. This transition towards HPC has to be incremental to ensure that trust is progressively built between PRACE and CERN. Finally, all the LHC experiments are data-intensive and most of the codes access data in a just-in-time fashion. This raises interesting challenges in terms of data transfers that could lay the foundations for a joint effort to build a European data infrastructure.

In September 2017, Service 2 organized a first joint PRACE-CERN meeting that highlighted important differences between CERN and PRACE computing models. Indeed, most of the LHC experiments are data-intensive and access date in a just-in-time fashion. These collaborations are also able to exploit or scavenge cycles on various computing infrastructures, include some HPC centres to process a long-lasting stream of computations. Conversely, scientific projects supported by PRACE correspond to highly specific and optimized parallel codes and are bounded in time and resource usage. However, the conclusions of this first meeting were that while most of the LHC codes are not HPC-ready, there exists a will to go for more HPC, but will require training that PRACE can provide. Such a transition towards HPC has to be incremental to ensure that trust is progressively built between the two communities. Moreover, the data-related challenges raised by CERN create an opportunity for a concrete collaboration that could be used to target future EU calls with a joint answer. The objective of this pilot was to keep the momentum and build on existing initiatives.

3.2. Evaluation of the pilots

3.2.1. Enabling ESRF computations on PRACE resources

Several unexpected and unfortunate events that happened to the members of the ESRF involved in the pilot have prevented its full implementation. With the help of the support team from PSNC, they were able to install Ptypy and run a few test jobs, for a total amount of 15 core.hours but the experiments could not go further. They also tried to install and test another candidate application, COMSYL which performs numerically coherent mode decomposition of the undulator radiation. However, the complex software dependencies of this code, including some private libraries, prevented the success of this installation.

On the GPU resources provided by the MPCDF, the execution of the Pynx code was more challenging than expected. Indeed, during the test phase of the pilot, the only available version of Pynx was coded in OpenCL and relied on pyOpenCL to access GPUs. pyOpenCL compiles kernels and selects a GPU device on-the-fly, which was not working on the MPCDF resources. Despite the assistance of the support team, it seems that the OpenCL ICD loader, which is used to automatically test which platform and devices are available, is not compatible with the module structure used by Tier-1 centres. Then a second code for Coherent Diffraction Imaging (CDI), based on CUDA (through pyCUDA), was also tested. Test runs that optimize the same dataset over up 32 GPUs in parallel showed very good performance and scaled as expected. Offloading this kind of computations to the HPC resources of PRACE could be very useful to the ESRF as a full optimisation in CDI usually requires more than a hundred of such optimisations to get statistically significant results and a better confidence on the final reconstructed object.

Unfortunately, events made impossible the further investigation of specific applications (i.e., ptychography-tomography) which would have used a few hundreds and up to thousands of GPU hours. This would however be the typical application ESRF would be interested to execute on PRACE resources.

The main benefit for ESRF of using PRACE GPU resources lies in the number of available resources. The GPUs at MCPDF are similar to those owned by ESRF but while ESRF has only 22 GPUs, the Hydra cluster has 676.

Assuming that 20 times more GPUs could be reserved at MPCDF than at ESRF, the estimated gain would to reduce the duration of a typical run from 11 days at ESRF to 13 hours at MPCDF. Note that if ESRF had to run this code on their CPU resources only, it would take more than 1,000 days.

A more specific application

(i.e., ptychography-tomography) has been tested at IDRIS. 1,200 projections with each 17 frames of 4k x 4k has been run with different datasets.While such experiments would have last for 11 days on 5 hosts with 2 K20 on the ESRF cluster, they only took 7.5 hours on 8 hosts with 4 P100 GPU each at IDRIS, outlining the benefit of leveraging PRACE resources. However, this gain does not include the cost of transferring data (a few hours) from ESRF to IDRIS.

3.2.2. Enabling CERN computations on PRACE resources

On 22 October 2018, a second joint PRACE-CERN meeting was held whose main objective was to discuss collaboration opportunities between the two institutions. More precisely, this meeting was the occasion for CERN to learn how PRACE works and how to interact with PRACE, to learn about the training activities offered by PRACE, to understand the mechanisms for resource allocation, scheduling, I/O, and authentication and authorization, and firewalls, to present PRACE plans for the next generation of hardware deployments. Conversely, it was the occasion for PRACE representatives to understand High Energy Physics software and determine how to best exploit HPC resources, present the current experience of the LHC experiments with HPC centre, and discuss what should be changed on the infrastructure and application sides to enable a more efficient usage of HPC resources.

This meeting confirmed that there are significant differences in scheduling, access, and services between the PRACE HPC community and large scientific communities such as the LHC. While it does not seem reasonable to expect that either community will entirely bridge the existing gap, it is worth recognizing that improving the scientific applications at CERN to be better tuned for HPC environments and make the PRACE HPC centres evolve to be better suited for data intensive sciences would bring the two communities much closer.

The following challenges were identified:

- The LHC experiments are open-ended multi-year projects that need predictable computing resources, which cannot be easily accommodated by the PRACE annual proposal-driven allocations.

- Most High Energy Physics applications currently make very little use of high-performance communication between processes due to the nature of their workflows. In turn, HPC centres typically significantly invest in the optimisation of the interconnection network. Therefore, one of the most expensive components of the infrastructure of PRACE Tier-0 and Tier-1 would not be leveraged by the applications submitted by CERN.

- A common set of interfaces for authorization, resource allocation and data management would be needed in PRACE centres to reduce the cost of adoption by CERN users.

None of the challenges above is seen as insurmountable and represent interesting R&D opportunities instead. Moreover, they are also shared by other data intensive science project, such as the forthcoming Square Kilometre Array (SKA) telescope.

A few early joint strategic and technical activities were proposed during this meeting, some of them being also relevant to SKA. The proposed strategic activities are:

1. Develop a four-way agreement between CERN, SKA, GEANT and PRACE to explore long-term cooperation to support the LHC and the SKA science programmes.

2. Write a survey of the current LHC experiment activities using PRACE resources that documents the existing successes and challenges. This document should also develop a list of priorities for a pilot use case making an efficient use of PRACE sites and resources for as many LHC workflows as possible.

3. Develop services and tools for both LHC and SKA workflows.

4. Develop a software-driven initiative to tackle the LHC and SKA needs in terms of software optimization and performance on HPC resources.

5. Create training programs tailored to favour the adoption of HPC architectures by the LHC experiments and the performance optimization of their codes on high-end accelerators such as GPUs and FPGAs.

And the proposed technical activities are:

1. Definition and execution of a pilot project for an HTCondor overlay for PRACE resources based on a similar work done at CERN. The allocation of PRACE resources could then be performed generically for the LHC community via a HTCondor pool.

2. Definition and execution of a data federation demonstrator: the goal is to demonstrate data delivery at run-time through the HPC firewalls at an incoming rate that is sufficient to efficiently operate LHC data intensive workflows on PRACE resources at scale.

3. Definition of a program to demonstrate that local storage at HPC sites can record data at production scale from high-output workflows (reconstruction and simulation). This demonstrator should also verify that export from the HPC facilities to remote custodial storage can be sustained at a level which permits a continuous operation.

3.3. Recommendations for PRACE-6IP

Despite the events that limited the implementation of the pilot, ESRF renewed its declaration of interest for a formal collaboration with PRACE. Getting access to additional community shared resources for data analysis that ESRF cannot provide in-house computing power would be a real game changer with the forthcoming upgrade of the X-ray source. ESRF has been testing several external resources beyond PRACE (e.g., Amazon EC2, HNSciCloud, EOSC, or loaned NVidia machines) over the last year. The preliminary tests with PRACE were very positive and show important benefits in terms of execution time that greatly reduce the time to solution. Moreover, the measures made on PRACE resources are consistent with what was observed with other resource providers. A noticeable difference though is that PRACE can give access to cutting edge hardware (e.g, the OpenPower prototype at IDRIS), where most cloud providers are limited to more mainstream resources. However, some leads to improve the links with large scale scientific instruments have also been proposed. First, standardised packaging deployment methods (e.g., using containers docker images) would be very helpful to make the resources more easily usable for HPC-experts scientists. Second, the lack of an interactive machine with access to the same GPU setup as the other machines at MPCDF increased complexity of the experimentation made by

ESRF. Third, the large volumes of data needed as input of several ESRF applications restricted this first pilot to a limited set of simulation applications. To be able to leverage PRACE resources for all the applications, a guaranteed high bandwidth end to end connection from ESRF to PRACE would be needed, which implies the need for a formal agreement with GEANT. Moreover, a long-term commitment from PRACE to grant access to resources to ESRF would be a great incentive to allocate more human resources on the ESRF side to port more applications to HPC resources.

PRACE has an in-depth experience in building and operating large computing facilities, optimizing software, and migrating applications to new accelerated hardware while CERN, and the High energy Physics community at large, have expertise in data management, data access, and data-intensive applications. Thanks to this uniqueness of competencies, a close collaboration between PRACE and CERN will be beneficial not only for the both parties but also to others large scale scientific instruments such as the Square Kilometer Array telescope. Developing a common set of tools and interfaces to reduce the cost of accessing individual HPC centres and to make more efficient use of them will be the next milestone. Moreover, the exploitation of Exascale computing to process Exabytes of data calls for the next generation of systems for data management and data access. The proposed joint activities between PRACE, CERN, and SKA and the accumulated knowledge from each party will allow to solve the challenge of data intensive science on HPC centres by streamlining data delivery and data access in large scale facilities. They will also act as a lobby to encourage investments in enhanced software skills in HPC and better career recognition for developers.

4 Service 3: Smart post processing tools including in-situ visualization

The main goal of service 3 within Task 6.2 was to define which services, policies, tools could be deployed to support post-processing activities related to HPC simulations running on PRACE HPC facilities.

The need for the development of these tools originates from the fact that more and more time is spent during the simulation for I/O operations and post-processing. I/O is recognized to be the main bottleneck to achieve the Exascale computing, which is up to 1000x faster than current Petascale systems.

We decided to focus on three topics, all related to reduce the time spent in an HPC simulation to extract useful information from the results on our HPC centres.

These three aspects correspond to different “pilots” that are relatively decoupled but nevertheless could come into play within the same HPC and even be used within the same community or the same application in different stages:

- **Large data I/O optimisation:** The first pilot, called SAIO, aims to optimise the parallel I/O operations semi-automatically. SAIO is a light weighted and intelligent framework that utilises the machine learning concept to optimise parallel MPI-IO operations. It frees the users from struggling with different I/O strategies or I/O configurations for their applications by setting the MPI info objects transparently. All applications, which are built

upon MPI-IO, parallel HDF5 and parallel NetCDF, might benefit from SAIO, especially if the I/O has not been manually optimized by the user. In this work a new version of SAIO has been released, starting from the code developed in PRACE-4IP. This new release comes with many performance improvements, demonstrated in a real case dealing with analysis and parameter optimization within the industrial fluid dynamics field and within the IOR benchmark.

- **In-situ Visualization Service**: starting from the work done in PRACE-4IP where catalyst has been coupled with an in-house academic code and tested only with rectilinear grids, in this second pilot we explored the feasibility of applying in-situ visualisation techniques to a widely adopted Open Source CFD simulation code and for many kinds of grids. OpenFoam has been selected as a good candidate for this experiment because it has a very strong community of users, power users and developers. As far as we know, no in-situ instrumentation has been done yet in OpenFoam. Catalyst has been used as the in-situ use case library which allows the coupling between OpenFoam and Paraview, a well know post-processing tool in the CFD world.
  Exploration of in-situ techniques for OpenFoam could have different purposes and goals, for example to provide OpenFoam users early visual feedback of their current simulation jobs and to reduce the amount of data stored during the simulation; in fact, an early and interactive feedback could help in tuning the frequency, resolution and format of saved data.
- **Remote Visualization Service**: For the third pilot, a tiled visualisation tool named TileViz, has been developed within the MANDELBROT platform (large high resolution display wall, cluster and interactive devices), at CEA Saclay. Instead of a single huge screen as in the RCM tool developed by CINECA in PRACE-4IP, the TileViz software allows users to display many visualisation tool outputs side by side. Each tile is connected to a remotely deployed docker container running a VNC server. A work session can be planned on a desktop computer, with a browser, and then brought in the high-resolution room for collaborative meetings to analyse multiple simulations at the same time and to manipulate the tiles on the wall with our 64 inches tactile table, or any connected tablet.

All pilots have been deployed on different clusters and use a deployment environment tailored to HPC environment.

4.1. Description of the prototypical services

All pilots have in common the goals of reducing the turnaround time from simulation to insight, and rely on open source technologies and try to be as much as possible architecture neutral.

The first pilot has the goal to reduce the time spent during the parallel I/O operations of a simulation. It is a framework that utilizes a machine learning approach to optimize the parallel MPI/IO operations.

The last two pilots deal with visual interactive post-processing applications and both aim at simplify their deployment and adoption; they aim at providing an environment where the computational scientist find easy to use visual tools to explore and analyse the data.

Both pilots have been deployed on different clusters and use a deployment environment tailored to HPC environment. Specifically, this includes in-situ visualization (pilot 2) which requires an environment similar to the one provided by remote visualization (pilot 3): in-situ experiments rely on ParaView application being available and usable within a remote visualization service, they could be viewed as scaled up visual applications, leveraging on application neutral remote visualization services.

4.1.1. The SAIO tool (Large data I/O optimization)

SAIO, a **S**emi-**A**utomatically **I**/**O** Tuning Framework, uses machine learning to determine the optimum configuration of Lustre stripe counts, stripe size and MPI collective read and write [4]. This tool is designed to be portable across multiple HPC platforms and requires little knowledge of parallel I/O optimization. As a wrapper for MPI-IO library, it is compatible with the parallel HDF5 and parallel NetCDF libraries. SAIO has proved its scalability by being successfully tested on 3000 compute nodes. Figure 1 shows the architecture of the tool which is composed of two functional modules. Core module performs the run-time I/O tracing and optimizing whereas learning module parses log data and generates optimum configurations. By saving the I/O request information into log files and thus generating the log file pool, optimal configuration can be determined. Various configuration setups are stored in the configuration pool. SAIO tool can be run in different modes which include tracing, optimizing and recording data transfer size of each I/O process. Each mode can be selected by setting the SAIO_MODE variable before running the application.

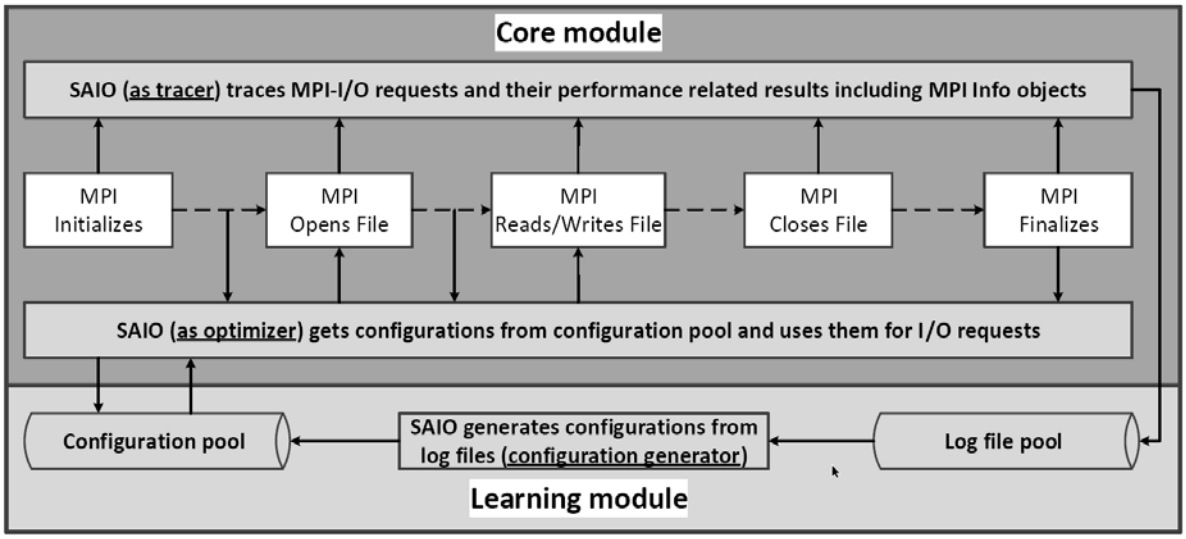


Figure 1: SAIO architecture

4.1.2. In-situ Visualization Service

It is well known that more and more time is spent during the simulation for I/O operations and postprocessing. I/O is recognized to be the main bottleneck to achieve the Exascale computing, which is up to 1000x faster than current Petascale systems [5]. The main cause can be identified in

the disproportion of the rate of change between memory and external storage and CPUs. With such a limited I/O bandwidth and capacity, it is acknowledged that traditional 3-step workflows (pre-processing, simulation, post-processing) are not sustainable in the exascale era. One approach to overcome the data transfer bottleneck is through an in-situ approach: the in-situ approach moves some of the post-processing tasks in line with the simulation code [6].

ParaView Catalyst [7] is an open-source data processing and visualization library that enables in-situ data analysis and visualization. Built on top of and designed to interoperate with the standard visualization toolkit VTK, Catalyst enables simulations to perform analysis, produce output data and visualize intermediate results during a running simulation concurrently [6].

OpenFOAM [8] is a well-known open-source CFD software package used by engineers and scientists from both the commercial and academic industries. The desire to apply in-situ techniques to OpenFOAM using open-source software led to the development of a **plugin** based on the ParaView Catalyst library. The software has been released in the 1806 release of OpenFOAM (see https://www.openfoam.com/releases/openfoam-v1806/post-processing.php#post-processing-paraview-catalyst). The code can be freely downloaded from the following link (https://develop.openfoam.com/Community/catalyst) and is licensed under the GNU GPL license. A python installation recipe for the Spack HPC package manager is also provided (https://github.com/spack/spack/blob/develop/var/spack/repos/builtin/packages/of-catalyst/package.py).

Exploration of in-situ techniques for OpenFOAM could have different purposes and goals:

- provide OpenFOAM users early visual feedback of their current simulation jobs from the earliest stages of the computation;
- reduce the amount of data stored as early and interactive feedback could help in tuning the frequency, resolution and format of saved data;
- computational steering connections that let the scientist to analyse the results on the fly and changing of the analysis pipelines interactively, through user feedback;
- allow for scaling of heavy postprocessing operation that would benefit of the same scaling available for the simulation;
- develop analysis pipelines using C++ or Python that are executed along side the simulation run, in the same address space;
- promote adoption of web based presentation of simulation results for sharing amongst work groups by embedding production of visual artefacts within the batch simulation jobs (ParaView Cinema).

4.1.2.1. Architectural design

The in-situ service relies on ParaView in order to render the output of an OpenFOAM simulation instrumented with Catalyst during the run concurrently. The rendering is performed remotely, on a compute node of the HPC cluster, and nothing is stored on the hard drive.

For these reasons, this service needs a remote visualization tool to allow the user to login from his laptop to the HPC cluster, connect to a VNC server, start a GUI session using one of the available

window managers installed on the HPC machine, and operate with the ParaView software. The RCM tool, developed in the previous PRACE projects, has been adopted within the in-situ service for this scope.

This tool, called Remote Connection Manager (RCM), is a lightweight python3 software, which wraps underlying VNC component for simplifying VNC client installation, remote session activation and bookkeeping. It is based on the following software stack:

- **TurboVNC:** derivative of VNC (Virtual Network Computing) tuned to provide peak performance for 3D and video workloads
- **Fluxbox:** lightweight window manager
- **VirtualGL:** OpenGL interposing and GPU remotization library
- **Qt:** graphical user interface

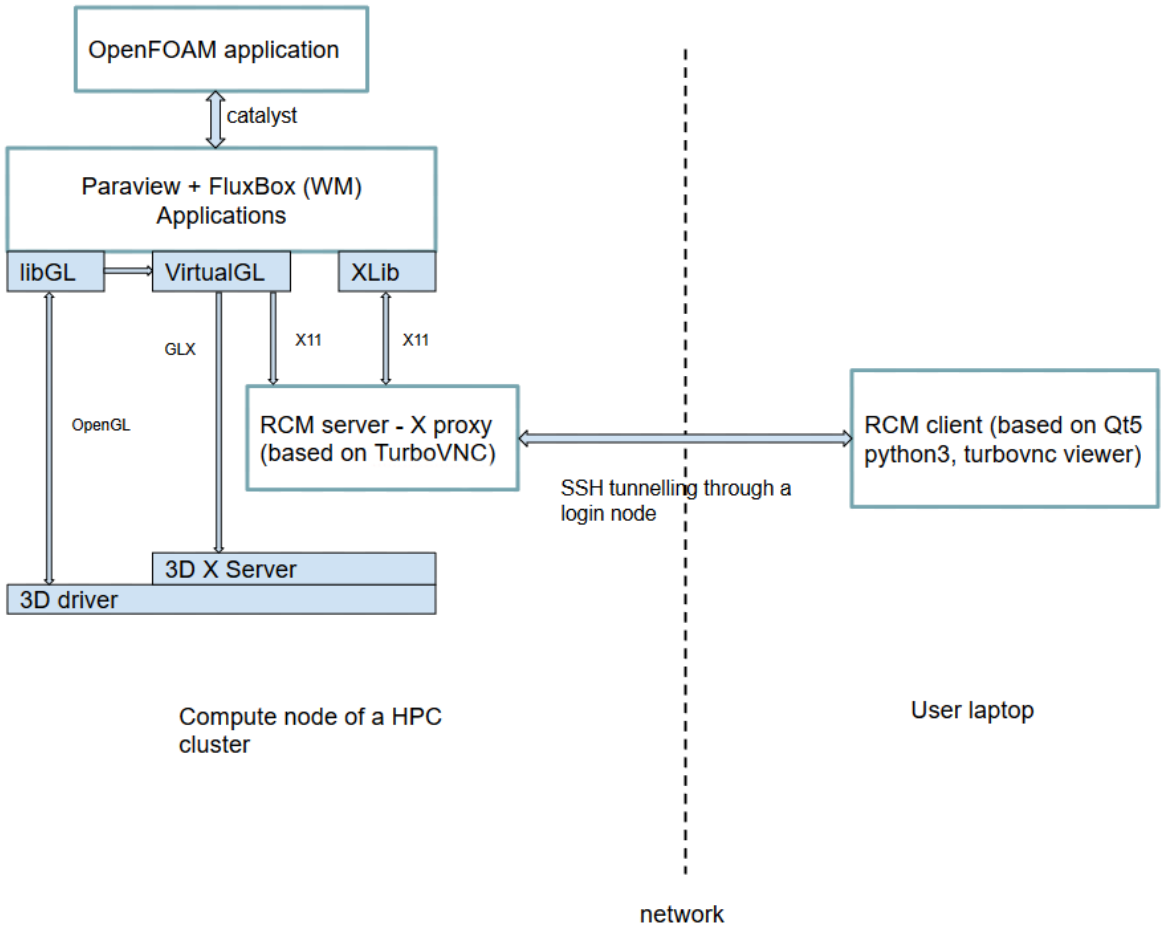


Figure 2: in-situ service architecture

4.1.2.2. Plugin Architectural design

The Catalyst plugin [9], [10] is implemented in OpenFOAM as a FunctionObject named c*atalystFunctionObject*. This implies that the functionalities can be simply plugged in by adding it in the c*ontrolDict* file. This function object is called many times according to the frequency specified by the user in the catalyst dictionary.

The *catalystFunctionObject* owns a ***coprocessor*** and a list of ***inputs***. The *coprocessor* is the interface between the simulation and Catalyst and it is implemented in the c*atalystCoprocess* class. The roles of the ***coprocessor*** are:

- initialize the Catalyst library
- load/reset the Catalyst scripts (e.g. Python pipelines)
- query Catalyst to determine if any analysis is required to be run for the current timestep
- an adaptor backend that maps the OpenFOAM data structures to Catalyst, which uses a VTK data model
- finalize the Catalyst library

Inputs are instead the sources of the VTK pipelines to be processed by ParaView Catalyst. The available *inputs* in the plugin are (see Figure 2):

- ***fvMeshInput:*** an input (source) from fvMesh regions
- ***faMeshInput:*** an input (source) from faMesh regions
- ***cloudInput:*** an input (source) from clouds (lagrangian)

Since each input works with different data types, a specific adaptor backend for the coprocessor has been developed:

- ***fvMeshAdaptor***: the backend for *fvMeshInput*; the output is a multi-block dataset with one block per region. Each region block contains up to two blocks corresponding to the internal (volume) mesh (block 0) and the boundary (block 1), which are further divided into sub-blocks for each patch.
- ***faMeshAdaptor***: the backend for *faMeshInput*. The output is a multi-block dataset with one block per area mesh. Each block is further divided in pieces for each processor.
- ***CloudAdaptor***: the backend for *cloudInput* for converting an OpenFOAM cloud to vtkPolyData. The output is a multi-block dataset with one block per cloud with pieces from each processor.

The *inputs* are specified by the user in the Catalyst dictionary. For each input the user can also specify its options, e.g. the fields and the regions for *fvMeshInput*, see the snippet code in Figure 3.

The *Catalyst Python scripts* are specified in the catalyst dictionary too. The Catalyst Python scripts can be generated automatically using the ParaView GUI with the Catalyst Script Generator plugin enabled Figure 3.

```
// ParaView Catalyst function object for OpenFOAM (-*- C++ -*-)

catalyst
{
    #includeEtc "caseDicts/insitu/catalyst/catalyst.cfg"
    scripts
    (
        "<system>/scripts/slice.py"
    );
    inputs
    {
        region
        {
            // All regions
            regions     (".*");

            internal    true;
            boundary    false;

            // Selected fields (words or regex)
            fields      (T U p);
        }


        // Solid walls only
        walls
        {
            internal    false;

            regions     ( heater "(?i).*solid" );
            patches     ( "(?i).*solid_to.*" "heater.*(Air|Water)" );

            fields      (T);
        }
    }
}
```

Figure 3: Example of a catalyst dictionary for fyMeshInput

4.1.3. Remote Visualization Service

A tiled visualization tool, TileViz, with a client/server python/Flask management and PostgreSQL database, is being developed within the MANDELBROT platform (large high-resolution display wall, cluster and interactive devices) hosted at La Maison de la Simulation, at CEA Saclay [11]. Instead of a single huge screen, the TileViz software allows to display many visualization tools outputs side by side over multiple screens (see Figure 4): personal desktop, our tactile table with touch interaction and our high-resolution display wall.

Figure 4: Work with TiledViz and ANATOMIST on neurobiologist's result

Each tile is connected to a Docker container deployed on an HPC computer (where the data are computed and stored). Within each container, an X graphical server is running with graphical acceleration plus a VNC server (a remote access tool) and the graphical applications needed by the researchers.

A user can create a project and a session and invites other remote users to share his connection to a supercomputer and build their own connections to other machines or data. Those connections give some tiles sets and a session. A special anonymous user may be invited with a special link from the session GUI to enable a special set of clients on a display wall, with no interaction on them. Session and tile sets can be saved, copied and modified to come back in another meeting with new password on containers.

A public version of the code is available on Github (https://github.com/mmancip/TiledViz).

4.2. Experiences with the prototypical services

The pilots presented in this section are very different, regarding scope, diffusion and code maturity. In this paragraph we will report the main results and the main considerations of each pilot.

4.2.1. SAIO Tool (Large data I/O optimization)

Experiments conducted on different platforms showed that SAIO tool is able to improve the performance of the scientific simulations and IOR benchmark significantly. Its memory and time

consumption remain small when compared to the overall application runtime and the time which is capable to save. End user doesn't need to be skilled in parallel I/O optimization to use this tool. It's enough to define a couple of environment variables in the job submission script before the application starts. Furthermore, since it's implemented as a dynamic library, recompiling the application's source code is not necessary. Other advantage is that SAIO is portable across multiple HPC platforms as well as with different I/O libraries. The only issue was with the bullx HPC platform where SAIO didn't improve the performance as much as on other infrastructures. Additional research on the platform in question could improve the performance. In the next paragraphs, some results on performance improvement are shown.

4.2.1.1. Engineering use cases on Cray XC-40 platform

4.2.1.1.1. Read heavy CFD code written using Fortran and HDF5

This code is part of the GCS-JEAN project [12] which studies turbulence and focuses on creating quieter, safer and more fuel-efficient jet engines. The part which is converting the result files of large eddy simulations to the source files of acoustic perturbation equations is considered a read heavy process since it performs read operation on more than 500 HDF5 files. Only 250 seconds is spent on the SAIO training process. Figure 5 shows the results of the I/O optimization for the case of 1200 processes. Best result was obtained with SAIO-3 configuration where 4634 core hours has been saved when compared to the original configuration.

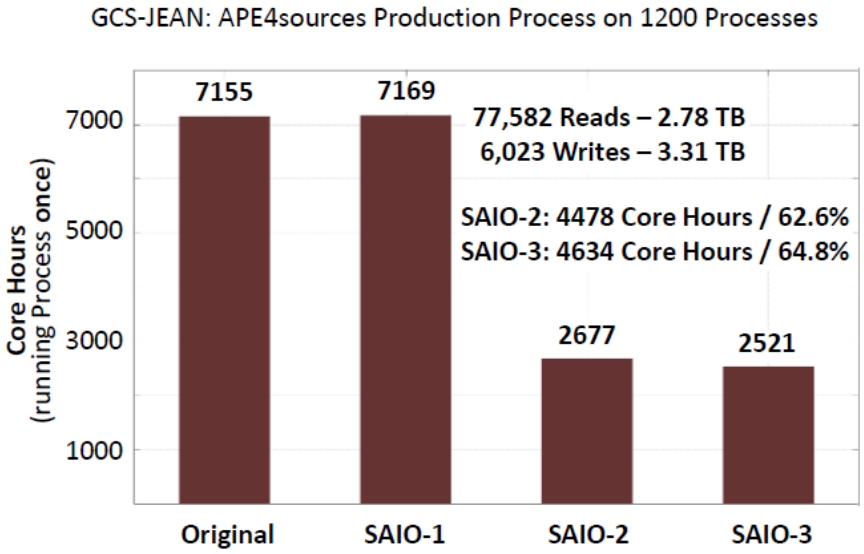


Figure 5: Results of SAIO optimization on a read heavy process

4.2.1.1.2. Write heavy CFD case using Ansys Fluent

This project investigates an air entrapment in paint drops under impact onto dry solid surface [13]. Ansys Fluent provides proprietary independent and collective HDF5 I/O operations. Figure 6 shows the result of I/O optimization using SAIO. As a basis for the optimization, serial I/O module without MPI-IO library was used.

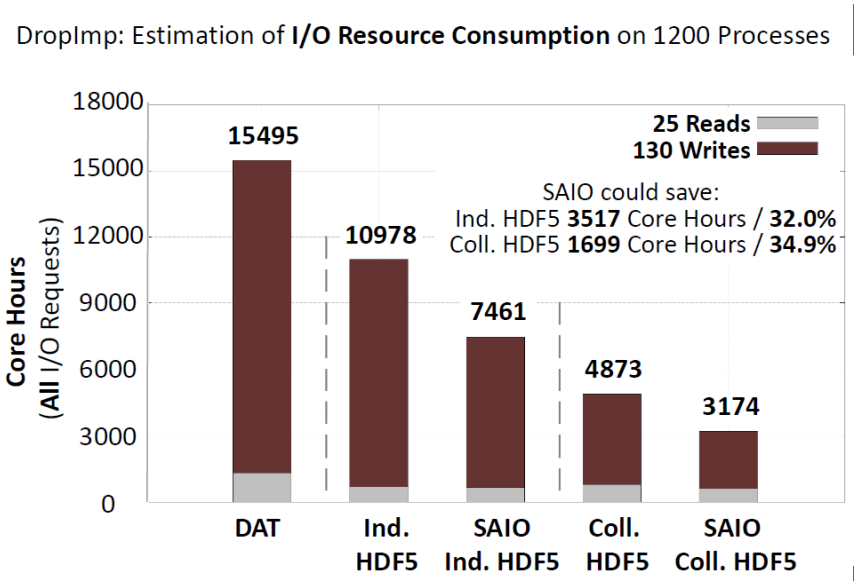


Figure 6: O/I optimization using SAIO tool on Ansys Fluent use case

Parallel HDF5 module with independent or collective I/O operations and with default setup already saved 4517 and 10622 core hours respectively. SAIO however, was able to save additional 3517 and 1699 core hours.

4.2.1.2. IOR Benchmark

Interleaved Or Random benchmark is used for evaluating the performance parallel file systems and it provides the I/O bandwidth as a performance indicator.

4.2.1.2.1. IOR benchmark on Cray XC-30 cluster

With SAIO, bandwidth was improved for 468 %. Table 1 shows the comparison between default and optimized setup. Lustre filesystem is based on Seagate Sonexion controllers.

	Default	**Optimized**
romio_cb_write	Automatic	Automatic
romio_no_indep_rw	False	False
striping_factor	1	16
striping_unit	1048576	2097152
Aggregate file size [MB]	960	
Bandwidth [MB/s]	466.85	2651.05
Improvement	468 %	

Table 1: SAIO optimized configuration for IOR benchmark on Cray XC-30 platform

4.2.1.2.2. IOR benchmark on Lenovo NeXtScale platform

SAIO can improve the bandwidth up to 969 % on GPFS file system. Optimization was performed for various data transfer sizes as presented in Table 2.

Data Transfer Size	64 kB	128 kB	256 kB	512 kB	1 MB	2 MB
cb_buffer_size	33554432	33554432	33554432	8388608	8388608	8388608
romio_cb_write	enable	enable	enable	enable	enable	Enable
romio_no_indep_rw	true	true	true	false	false	True
Agregate file size [MB]	17	34	68	102	272	544
Bandwidth (Default) [MB/s]	19.76	25.14	57.22	80.15	262.5	634.3
Bandwidth (Optimized) [MB/s]	146.72	268.73	494.8	559.46	1510.67	2258.97
Improvement	643 %	969 %	765 %	598 %	475 %	256 %

Table 2: Optimization results on Lenovo NeXtScale platform for various data transfer sizes

4.2.1.2.3. IOR benchmark on bullx platorm

In comparison with the previous results, SAIO improved the performance on the bullx platform with Lustre filesystem only for some of the data transfer sizes. As presented in table 3, the best improvement achieved is a 57 % for a 64 kB transfer size. For the case of 512 kB and 1 MB, decrease in performance was recorded.

Data Transfer Size	64 kB	128 kB	256 kB	512 kB	1 MB
striping_factor	15	30	30	15	15
striping_unit	1048576	1048576	1048576	1048576	33554432
romio_cb_write	enable	automatic	automatic	enable	enable
Agregate file size [MB]	15	30	60	120	240
Bandwidth (Default) [MB/s]	49.61	95.88	185.35	355.84	567.58
Bandwidth (Optimized) [MB/s]	78.05	125.5	199.17	313.94	371.69
Improvement	57 %	31 %	7 %	-12 %	-35 %

Table 3: Optimization results on bullx platform

4.2.2. The In-Situ Visualization Service

During the project, we spent about one third of the time to install the software stack (ParaView, Catalyst, OpenFOAM, of-catalyst plugin) and automatize the installation process using the package manager Spack, one third for the development of the plugin and one third for the refactoring of the RCM software.

The development of installation recipes allows the reproducibility of the software stack on any machine in any HPC centres. The installation process of the huge number of dependencies of

ParaView and catalyst has been considerably simplified by the adoption of a package manager. This aspect is more relevant in a supercomputer, where the OS software stack is light.

The development of the plugins has required strong programming skills and a good knowledge of the architecture of ParaView and OpenFOAM. For these reasons, a fruitful collaboration with Kitware, in the name of Andrew Bauer, and ESI-OpenCFD, in the name of Mark Olesen, has been established in order to develop a catalyst adaptor for OpenFOAM to be released in the official repository of OpenFOAM.

The architecture of the previous RCM software did not allow the development of new services. Only a TurboVNC server running on a compute node can be requested by the user and the customization of the job scripts parameters was very poor. It was agreed to overcome these limitations by a massive refactoring of the RCM software. The main changes are reported below:

- moving from python2 to python3
- moving from tk to Qt
- support Linux, Windows and Mac OS platforms
- allows the possibility to select different ssh and VNC client implementations
- allows the possibility to support many schedulers and many services at the same time
- give the user the possibility to customize the job script parameters and the kind of service

Thanks to the refactoring, a new service, named in-situ service, was plugged-in. In future, many in-situ services based on different in-situ libraries and CFD codes, can be developed and plugged-in, according to the user requests.

4.2.3. The Remote Visualization Service

We have worked a lot on the continuous integration using our internal Gitlab server to ensure our developments are reliable for the customers (researchers and engineers) who come in Mandelbrot room to analyse their data sets locally with their own visualization tool. For example, in Figure 4, you can see many sessions with neurobiologists, from CATI team in Neurospin Institute, working on their brain data sets issued from Brainvisa, their machine learning application for RMI post-treatments, with ANATOMIST visualization tool and TiledViz application.

We have worked on the GUI and the touch interaction to improve the experience in front of the wall. In this first version, tiles are shown in a grid view. However, many researchers want to have some statistical algorithms to sort the tiles, so we have tested some D3js views to sort or move them. It will be available in a future version.

We have improved a lot the use of docker swarm network on our graphical cluster Mandelbrot. We have worked on an ANATOMIST server with CATI team and a specific ingress network between tiles to save all data from one element displays on the tactile table. The server is a special tile.

Docker is a powerful technology but it has the drawback that it cannot be deployed easily on every HPC machine. Indeed, for security reasons, we were not able to install any containers on the IDRIS and Poincaré clusters.

We began some tests using the Singularity technology on the fusion machine in Centrale-Supelec school, but no real tests with TiledViz yet.

Some preliminary investigations on the use of the PCOCC software in order to deploy a docker swarm network on a graphical node of a HPC cluster have been started.

4.3. Recommendations for the next implementation phase

We report for the three pilots, a set of suggestions for the next implementation phase. These recommendations reflect the status of the pilots at the end of the current project.

4.3.1. SAIO tool

Regarding future activities within PRACE-6IP, SAIO is considered to become a regular service as a part of the WP6. The tool will be licensed and available for download, together with the user documentation. Considered are the following KPIs:

- Number of downloads
- Database with configurations for different HPC platforms

Database will be an expandable list of various configurations for different systems which will help decrease the necessary training time and to improve the performance even more. It is not excluded that more advanced machine learning algorithms will be integrated for that purpose.

4.3.2. The In-situ Visualization Service

Within PRACE-5IP, we have decided to offer to the user an in-situ service for the OpenFOAM software. In principle, a similar service can be provided for whatever CFD code coupled with Catalyst. To be more general, a code that can be instrumented with a library that implements the in-situ visualization can be adopted for the in-situ visualization service. A new library, named SENSEI and presented at the Supercomputing 2018 conference, can be a good candidate to replace Catalyst.

Some years ago, Kitware released ParaView-Web, a web framework which allow to bring the power of ParaView and VTK into the WEB. RCM has been refactored in order to allow to plug-in many services at the same time. A new service, based on ParaView-WEB, Catalyst and OpenFOAM can be developed in order to provide an in-situ web service in the HPC world.

4.3.3. The Remote Visualization Service

TiledViz has a lot of developments to complete to have a real application ready to be installed on all supercomputers in Europe. Priority is to develop the launch and management of remote containers from the GUI of TiledViz, using some of the most used containers tools, like docker swarm, Charly Cloud, Singularity and Shifter backends, and to build an interface to manage the database directly in the GUI, for example for the lost tiles. In a second step, we will add more grids shape for the view of a session using the D3js library, use PCA algorithms from researchers simulations and post-treatment, and move the tiles with sub-ensemble movements.

5 Service 4: Provision of repositories for European open source scientific libraries and applications

5.1. Description of the service

Service 4 of Task 6.2 aims to provide a series of repositories for European open source scientific libraries and applications, and focuses in the wide adoption, uniformity and consolidation of European products. The service was introduced during PRACE-4IP as a pilot, in order to get an overview of how the solution would work and to satisfy partner's needs with a real structure and with different accounts for different kinds of actors. At the beginning of PRACE-5IP, the service was deemed mature enough to become a regular PRACE service, and the transition from the pilot stage to production began.

This service provides enough tools to satisfy a wide range of needs and requirements for different projects and interests but at the same time, it must help to consolidate European products providing uniformity and consistency.

The implemented solution was to deploy a modern, useful and featured tool for code repository that will serve as the core for the solution. Around this core, different complements have been deployed and serve as key elements to achieve the required wide adoption and uniformity. These components consist of a project management tool, a bug tracker and a continuous integration system.

The core of the service has been decided to be based on GitLab given the analysis of current technologies and possible features studied. Other components are based also on open source software and are TRAC, Redmine, Jenkins and CASino.

The online open source PRACE repository can be accessed from the next URL: https://repository.prace-ri.eu/

The credentials to get access can be retrieved upon request on email prace-repo-access@lists.grnet.gr.

The following tools form the repository:

- Gitlab: Code repository and Continuous integration tool. Main tool of the open repository services.
- Trac: Project management & bug tracking
- RedMine: Project management & bug tracking
- Jenkins: Continuous integration system

The service uses internally an LDAP for authentication and the Login page is based in a CAS Central Authentication Service) technology, so with the same credentials you can access any of the repository's tools. Once you have been validated correctly, you will be redirected to the tool selected.

Moreover, a Service Use Policy document has been defined and approved, which will be described in the next paragraphs.

5.1.1. Service hosting

The service is a Virtual Machine (VM), hosted on GRNET ViMa IaaS service. The ViMa service aims to provide to the educational and academic community access to shared computing and network resources that can be used for production purposes. In particular, the service is addressed to the NOCs of GRNET's constituency or laboratories involved in national and European funded projects and have needs for computing resources. In order to be able to ensure high availability, the service is hosted on multiple computing clusters on more than on Datacenters, while for the safe storage of data an external SAN unit is used. The network infrastructure ensures seamless connection to the backbone network and the commercial internet at very high speeds. The service supports a variety of benefits tailored to the needs of potential users. Users are able to use the IP addresses of their institutions to virtual machines using L2VPN, as well as using IPv6.

The particular VM that is hosting the PRACE repository consists of 2 virtual CPUs, 8 GB RAM and 200GB hard disk storage. It is possible to increase the available resources if the need arises in the future. There is also a development clone of the production VM, which is used for testing updates and other changes to the service. Both VMs, production and development, are maintained by GRNET and NIIF.

5.1.1.1. Backup

There is a daily backup of the VM contents (PostgreSQL Database, GitLab storage) to the GRNET ARIS HPC storage. This ensures that in case of a storage or other failure the user data stored in the PRACE repository will not be lost.

5.1.1.2. Security updates

A cronjob has been configured that notifies via email the administrators for available updates. In this way it is safeguarded that security updates are installed in a timely manner.

5.1.2. Code Repository: GitLab PRACE

GitLab [14] is the core of this service and a web-based Git repository manager with wiki and issue tracking features. GitLab offers hosted accounts similar to GitHub, but also allows its software to be used on third party servers.

- Features
  - Git repository management
  - Code reviewer tool
  - Bug/Issue tracking tool
  - Activity feeds
  - Integrated Wiki
  - GitLab CI for continuous integration and delivery.
  - Open Source: MIT licensed, community driven, 700+ contributors, inspect and modify the source, easy to integrate into your infrastructure
  - Scalable: support 25,000 users on one server or a highly available active/active cluster

- Roles and Permissions
  - Types of permissions
  - Breakdown of permissions are accessible in GitLab documentation. Structure is the same than the one presented for Owner-Admin-Write-Read permission for GitHub.
  - Groups
  - Groups can have different members with different permissions. When multiple projects are assigned to the same group, the members will have the same permission for all the projects. One can promote specific members of the group for specific projects by adding them also as a member of a project.

- Interface and Access: the interface and access methods are exactly the same than the specified for GitHub.
- GitLab Pros, based on requirements of PRACE:
  - Not depending on an external enterprise service
  - Can really say it is a separate PRACE service/repository
  - Can be PRACE branded with look and layout changes for each subpage (CSS, HTML5), other PRACE services can be linked from / integrated
  - Absolute freedom of configuration, installation of application addons and freedom of group management and private repositories
  - Mobile apps
  - Option to integrate with LDAP / connect with PRACE LDAP and use PRACE userbase
  - Integration of data into other sites (e.g. PRACE web, training portal, etc.) is possible and customizable
  - Built-in advanced wiki features that can be updated with git
  - Powerful import features from GitLab
  - Gravatar integration allows using the same avatar used on github
  - Unlimited public/private repos without the need of upgrading plans
  - Integration option with GitLab ci to test, build and deploy code snippets
  - UI is very similar to GitHub, users are familiar with it.
  - Possibility to use federated login, like eduGAIN (using SSO method), auth eduGAIN members seamlessly
  - Advanced Jira Support, Jenkins support

- GitLab Cons:
  - One-time effort of deployment and configuration
  - Requires operation effort to run (these two however might be covered as a WP6 effort)
  - Requires hosting (there are numerous PRACE services hosted by PRACE partners independently from IPs)

5.1.3. Project Management & bug tracking: TRAC

Trac [15] is an enhanced wiki and issue tracking system for software development projects. Trac uses a minimalistic approach to web-based software project management. It helps developers write great software while staying out of the way. Trac should impose as little as possible on a team's established development process and policies.

It provides an interface to git, an integrated Wiki and convenient reporting facilities.

Trac allows wiki markup in issue descriptions and commit messages, creating links and seamless references between bugs, tasks, changesets, files and wiki pages. A timeline shows all current and past project events in order, making the acquisition of an overview of the project and tracking progress very easy. The roadmap shows the road ahead, listing the upcoming milestones.

Main components are:

- Wiki subsystem: TracWiki: built-in Wiki
- Version Control subsystem:
  - TracBrowser: Browsing source code with Trac
  - TracChangeset: Viewing changes to source code
  - TracRevisionLog: Viewing change history
- Ticket subsystem:
  - TracTickets: Issue tracker
  - TracRoadmap: Tracking project progress
  - TracReports: Writing and using reports
  - TracQuery: Executing custom ticket queries
  - TracBatchModify: Modifying several tickets in one request
- Other modules:
  - TracSearch: Full text search in all content
  - TracTimeline: Historic perspective on a project
  - TracRss: RSS content syndication
  - TracAccessibility: Accessibility keys

5.1.4. Project Management & bug tracking: Redmine

Redmine [16] is a flexible project management web application. This can accomplish similar points of TRAC features. Both software products have been included in the service since different partners could be using different tools.

Some of the main features of Redmine are:

- Multiple projects support
- Flexible role-based access control
- Flexible issue tracking system
- Gantt chart and calendar
- News, documents & files management
- Feeds & email notifications
- Per project wiki
- Per project forums
- Time tracking
- Custom fields for issues, time-entries, projects and users
- Git integration

- Issue creation via email
- Multiple LDAP authentication support
- User self-registration support
- Multilanguage support
- Multiple databases support

5.1.5. Continuous integration system: Jenkins

Jenkins [17] is a self-contained, open source automation server, which can be used to automate all sorts of tasks such as building, testing, and deploying software. In this aspect is similar to the features provided by GitLab integrated CI, but it is a dedicated and very powerful tool widely used by many projects.

5.1.6. Continuous integration system: GitLab PRACE

GitLab has integrated CI (continuous integration) and CD (continuous deliver) to test, build and deploy code.

- Multi-platform: builds can be executed on Unix, Windows, OSX, and any other platform that supports Go.
- Multi-language: build scripts are command line driven and work with Java, PHP, Ruby, C, and any other language.
- Stable: your builds run on a different machine than GitLab.
- Parallel builds: GitLab CI splits builds over multiple machines, for fast execution.
- Realtime logging: a link in the merge request takes you to the current build log that updates dynamically.
- Versioned tests: a gitlab-ci.yml file that contains your tests, allowing everyone to contribute changes and ensuring every branch gets the tests it needs.
- Pipeline: you can define multiple jobs per stage and you can trigger other builds.
- Autoscaling: you can automatically spin up and down VM's to make sure your builds get processed immediately and minimize costs.
- Build artefacts: you can upload binaries and other build artefacts to GitLab and browse and download them.
- Test locally there are multiple executors and you can reproduce tests locally.
- Docker support: you can use custom Docker images, spin up services as part of testing, build new Docker images, even run on Kubernetes.

5.1.7. Account management: LDAP

In this case we use LDAP in order to provide authentication and account management. The deployment has been based on a standard OpenLDAP installation with default schemas. In order to ease the management of the LDAP tree, LDAP Account Manager (LAM) has been also deployed in the server. LAM provides a nice web interface from where you can add or remove users, manage

groups, etc. This tool is very convenient since it is also able to execute particular scripts when users or groups are created or modified, thus creating base directories, projects, etc.

5.1.8. Single Sign On to all services: CASino

One of the main concerns of having multiple components is the fact that the user must log in on each service if a special system is not configured. This is the purpose of the Single Sign On (and sign-out) software. This software allows the user to enter his username and password only once, and then be automatically logged on each of PRACE repository services. When logged out same thing must happen, the user is logged out of all PRACE repository services.

The elected solution has been one of the easiest to install available nowadays, it is CASino [18].

5.1.9. Documentation

The PRACE repository documentation has been uploaded to the PRACE web site, in the User Documentation section:

http://www.prace-ri.eu/prace-repository-services/

5.2. Service Policy

In this chapter, the PRACE repository service policies are described, divided in the next items:

- **Access policies**: Eligibility of use and procedure to request access
- **Authentication policies**: Authentication of repository user policies
- **Usage and Data Policies**: Terms and conditions of the service

5.2.1. Access Policies

Here we describe who and what projects can access the repository system and from what countries/institutions. Take in mind that different privacy policies can be applicable coming from different countries.

- All PRACE projects users (PRACE users with PRACE allocation hours assigned) have access to the repository by default
- All PRACE researchers/staff (users registered in PRACE LDAP as staff) can ask for creation of new projects inside the repository and have access to it

External users or entities that have some type of collaboration (a signed MoU) with PRACE are eligible to ask for an account in the repository. PRACE BoD can provide access to external entities or projects which are considered strategic.

An access committee formed by the Repository service coordinator and Operations work package representative will evaluate the feasibility for the inclusion of new projects and its content into the repository.

The repository service coordinator must provide the steps to consolidate the PRACE repository as a valuable resource for the institution and for the research in general. That means that she has to manage the service and align it with the needs addressed from the PMO and PRACE infrastructure.

5.2.2. Authentication policies

The authentication method is password based to provide an easier way of usage for everybody.

The passwords are recommended to have an expiration of 2 years, this will be controlled in a standard LDAP field and a warning script could be deployed to send a warning to the user.

Accounts must be personal and projects will have r/o and r/w users, with r/w users being responsible for the repository content.

5.2.3. Usage and Data policies

Any user of the repository will be responsible for the content hosted in PRACE repository, to be aligned with applicable laws, including copyright or trademark laws, export control laws, or other laws in your jurisdiction.

The uploaded contents must support research, especially European research. It is not permitted to use the repository and services for non-research and personal or company profit only.

The service is intended to share the knowledge within research world and to provide value to the community. This means that the authors, owners, licenses, copyrights, etc. will be respected but the idea is to share and open the contents.

Private projects can happen under the supervision of the access committee.

5.2.3.1. Terms and Conditions

The following Terms and Conditions of Access (in English) are presented to each user when they register. We anticipate that they will change rarely, if at all. It also includes an agreement to the Service processing the user's personal data; in return, the Service undertakes to follow the Personal Data and Privacy Policy, adopted by PRACE.

> *"You agree:*
>
> *only to use the service for the purposes for which you were given access, for example as specified in your project award*
>
> *not to disrupt the working of the service, for example by knowingly introducing malicious software into it, nor to try to breach its security or use resources which are not assigned to you*
>
> *not to interfere with other users' work, corrupt their data or invade their privacy*
>
> *not to infringe copyright or other intellectual property rights*
>
> *not to take data from any database or dataset without the explicit or implied permission of its owner*

not to use another person's account, nor to let other people use your accounts, except by agreement with us;

to keep your passwords confidential, and to inform us if someone else learns any of them or if you become aware that the security of our systems is compromised in any way

not to misuse the Internet, for example by sending spam or malicious software, by pretending to be someone else or by doing anything that might hinder or prevent someone else from using the Internet legitimately

not to use the service for illegal or immoral purposes, such as theft, fraud, drug-trafficking, money-laundering, terrorism, pornography, violence, cruelty, incitement to racial hatred, prostitution, paedophilia, or for offensive, obscene, abusive, menacing, defamatory or insulting behaviour

to comply with any special conditions that may apply to particular software packages

We agree:

as far as we reasonably can, to provide a 24-hour service as described on this website, it being understood that there will be times when the service is unavailable, for example as a result of unexpected failures, maintenance work or upgrades

to take reasonable steps to protect your data from being lost or corrupted.

to protect the security and privacy of the data we hold about you, as described in our Personal Data and Privacy Policy

that we will acquire no intellectual property rights over your software and data

to respond promptly to any complaints or suggestions you make about the service

You accept:

that the service is provided "as is" and we can't guarantee 100% perfection. In legal terms, this means that we are excluding all warranties and conditions applying to the service, including those implied by law.

that you are responsible for your use of any advice or information we may give you. We will take reasonable steps to ensure that it's true and useful, but we cannot guarantee this. In legal terms, this means that we expressly disclaim any and all liability for all representations, statements, conditions or warranties to that or any other effect except to the extent that such liability may not be lawfully excluded.

that we may make changes in the service

that we will use the personal details which you supply to us, together with records of your use of the service, as described in our Personal Data and Privacy Policy

that we may suspend your access to the service and discuss this with your project leader if it seems to us that you are breaking these Terms and Conditions; that if it is necessary to protect the service or other users' work or data, we can halt the execution of any program which you start; and that we have the right to close your accounts

that you alone are responsible for what you do when using the service. If you break the law you alone must answer for it, and if you cause damage to anyone else, you alone are liable, not us

that you will acquire no intellectual property rights over the software or any information we provide

that the use of the service by nationals of certain countries is controlled by special regulations laid down by many European Governments in connection with the Wassenaar Arrangement

that we may make reasonable changes to these Terms and Conditions at any time, and, once we have posted those changes on our website, the new version will then apply to you

If you have any questions about these Terms and Conditions, please contact the PRACE Helpdesk.

This policy applies to personal data about users of the PRACE Repository service:

We will store in the service's database the personal data you supply to us when you register as a user of the service and later.

We will also store in the database details of your use of all aspects of the service, including, for example, the amount of storage space you use and details of accesses to your repository.

We may use this information to help us manage and administer the service, to review, analyse and audit its performance and its patterns of use, and to plan for the future.

This information will be available to those members of the staff who are working on the PRACE project. They may also be available to the leader of your research project and to anyone whom the project leader designates as a manager of the project.

Information about the use made of the service by projects and the sub-groups within them may be made available periodically to PRACE management. Individual users will not be identified. This information may also be placed on the service's public web pages.

We reserve the right to monitor your use of the service, including anything you transmit over the Internet, and any data or software you store on our systems, in order to ensure that you and all the other users are complying with the Terms and Conditions of Access and not breaking the law. We must allow any court or other competent authority to inspect our records of your use of the system, or your data, and to take copies of it, if this is legally required; and we must report your activities to the competent authorities if we know or suspect that you are breaking the law. These are legal obligations for us.

We would not be able to administer the service properly, nor adequately account to our funding bodies for our conduct of this project, without processing your personal data in our database in this way. For this reason, we have to ask you to consent to this policy. This consent is included in the Terms and Conditions of Access.

If you have any questions about the treatment of your personal data, please contact The PRACE Helpdesk."

5.3. Use of the service

Up to now the component that has seen the most use is the GitLab service. As of this writing, 29 users have been registered to the service, 14 from WP7, 8 from WP4, and the rest from WP3/WP5/WP6. This is not very good number, but can be justified with the fact that the service is disseminated mostly internally for the PRACE users. When the migration to a regular service will be completely done, we expect more users.

The following groups have been defined in GitLab:

- CodeVault (WP4, WP5, WP7) with the following subgroups:
  - Training material/PGAS programming
  - Training material/Parallel programming
  - Training material/GPU programming
  - Training material/Generic
  - HPC-kernels
- Data Analytics (WP6 - T6.2 Service 6)
- UEABS (WP7)

Each group has its own manager(s), who can manage the users belonging to the group, their rights, and the projects belonging to the group. This way the various groups or WPs can manage their own projects and users without the intervention of the repository admins, except for the user account creation.

Up to now, about 100 MB of data have been uploaded to the CodeVault projects, and about 80 MB to the UEABS projects.

5.4. Security review

In order to complete the transition of the PRACE repository service to production, a security review is in progress by the PRACE Security Forum. In this security review, the following service aspects are examined:

- Service architecture
- Security components (firewall etc.)
- Authentication and authorization
- Confidentiality (use of encryption)
- Security policy (updates, auditing)

This security review is the last step required to consider the PRACE repository a production service.

6 Service 5: The deployment of containers and full virtualized tools into HPC infrastructures

6.1. Introduction

Linux containerisation is an operating system level virtualisation technology that offers lightweight virtualisation. An application that runs as a container has its own root filesystem, but shares the kernel with the host operating system. This has many advantages over virtual machines. First, containers are much less resource consuming since there are no guest OS. Second, a container process is visible on the host operating system, which gives the opportunity to system administrators for monitoring and controlling the behaviour of container processes. Linux containers are monitored and managed by a container engine which is responsible for initiating, managing, and allocating containers. Docker [19] is the most popular platform among users and IT centres for Linux containerisation. A software tool can be packaged as a Docker image and pushed to the Docker public repository, Docker hub, for sharing. A Docker image can run as a container on any system that has a valid Linux kernel. HPC targeted platforms, e.g. Singularity [20] and uDocker, make it possible to use Docker containers in production for HPC systems.

Virtual machines (VMs) are widely adopted as a software packaging method for sharing collections of tools, e.g. BioLinux [21]. Each VM contains its own operating system. A VM monitor, also known as hypervisor, is the platform for managing and monitoring VMs. VM technology is suitable for packaging collections of tools that run independently or dependently on the top of a specific OS platform, e.g. a GUI that runs python and R tools on the top of Ubuntu Linux. VMs are also effective in cases where a specific Linux kernel or Windows OS is needed, the cases in which, Linux containers cannot be used as a solution.

6.2. Pilot description

This service (Service 5) has targeted evaluation and benchmarking of containerized and fully virtualized workloads on both bare-metal and cloud-based HPC clusters.

The service included the following pilots:

- Container workloads. Prototypes:
  - Docker: IBM LSF, Socker, and uDocker
  - Singularity (with MPI and GPU workloads)
- Fully virtualized workloads on Slurm. Prototype: PCOCC
- Containerized cluster with web-based front-end. Prototype:
  - Containerized Galaxy-HTCondor
- Containerized service platform. Prototypes: NIRD Kubernetes platform,
- Container enabled meta-Scheduler. Prototype: ARC Control Tower (aCT).

6.3. Overall Evaluation

The use of containers and VM tools has proven to offer more usability, portability, and reduce complexity. Multiple prototypes have been tested and multiple use cases have been supported. In all sites, at least one container/VM platform is already in production. It can be concluded that the use of containers and VMs for HPC workloads is useful and applicable. The following have been investigated:

Security: Security issues has been investigated and are resolved in most container platforms.

Scalability: Scalability tests, performed using singularity with MPI, proved that container applications are scalable. Unprivileged container platforms, uDocker and Singularity, proved to be scalable in terms of deployment.

Performance: There are almost no performance issues (especially for containers).

Deployment overhead: for most container applications, the build-once-run-everywhere rule can be applied. There is some deployment overhead:

- in case of MPI applications that the container needs to have the same host MPI installed internally in its filesystem.
- For applications that need HW drivers, e.g. GPU and IB, the drivers need to be installed in the container filesystem. Singularity resolves this issue for NVIDIA GPUs by detecting and mounting the host drivers in the container.

6.4. Site contributions

The following is a list of the PRACE sites have contributed to Service 5: **UiO/SIGMA2** (Coordinator), **CEA, CINECA**, **CNRS/IDRIS**, **EPCC** (observer), and **CESGA** (observer).

6.5. Prototypes and use cases

6.5.1. Docker (CNRS/IDRIS & UiO/SIGMA2 & CESGA)

Docker is the most popular platform among users and IT centres for Linux containerisation. The main drawback of Docker is that it is more suitable for service containers rather than job containers that are useful for HPC applications. This is mainly due to the following reasons: Docker does not support sharing of containers between multiple computers, and the Docker installation cannot be shared between multiple computers. Docker containers run by default as the system root, which has a strong impact on the security of the system and is described as the “Docker Daemon Attack Surface” [22] [23]. The two prototype implementations for the Docker pilot experiment has mainly focused on the security issue

6.5.1.1. The LSF/Docker platform

Status: Production

Docker is an interesting technology as it offers an easy way for users to run applications on various operating system. However, the deployment of Docker at IDRIS was slowed down due to the security risk inherent in this technology. The LSF/Docker platform that is currently installed on a Power8 prototype machine at IDRIS is described in Figure 7. It allows to run docker containers in a restricted environment for the users but offers a better security.

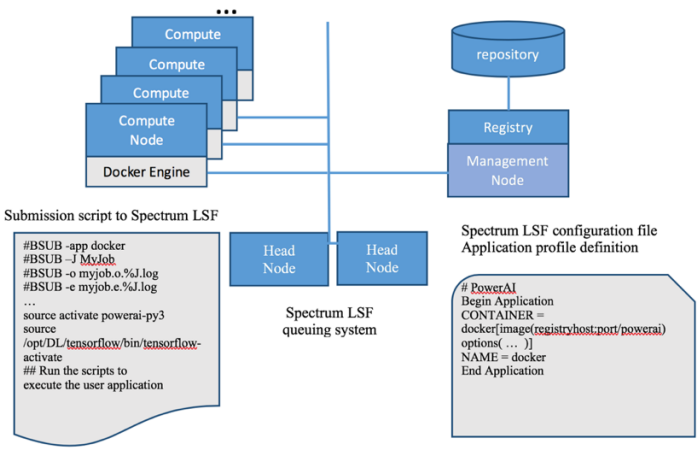


Figure 7: the LSF/Docker platform architecture

This software infrastructure or platform runs in a distributed environment where the various components are spread over the head nodes, compute nodes and management nodes. Each type of node has a well-defined role regarding the platform. The head nodes allow users to submit jobs to the Spectrum LSF batch queuing system. The batch queuing system has been configured to allow users to execute programs wrapped in docker containers as LSF jobs thanks to specific LSF directives. The compute nodes run the docker engines. It uses the registry to get the images to be executed within docker. The registry maintains the database of “authorized” docker images. The registry has been set up on a management node.

When a user submits a powerai job, LSF allocates compute nodes where the related docker image is deployed and the container runs. Given the file system mappings configuration, then powerai can be executed inside the running container.

During the prototype phase, we focus on securing the docker installation and configuration. We run a set of tests, split in 3 different domains:

- The first one is related to the docker installation on the Linux hosts. It includes the actions that have to be taken at the Linux hosts level to secure the docker installation in addition to the security best practices that are usually applied in an HPC site.
- The second is related to the docker configuration and operation. It includes the docker daemon configuration, the registry configuration and management, the images management and how docker operates
- At last, the LSF and docker interface which will focus on how the LSF specifications are forwarded to the docker environment.

The first two points can be checked using the docker security documentation which includes a detailed security related section that describes how to get a secure engine [24]. The third point, the LSF and docker interface doesn't rely on any benchmark and requires a detailed analysis in order to identify the tests to be performed. We carefully checked the Docker security vulnerabilities and threats and we observed that even we were running Docker, the custom design of our platform made that most of the time we were not exposed to these vulnerabilities and threats, making this platform very secure.

Use case

The use case that IDRIS had for using docker containers, was related to the Machine Learning/Deep Learning (ML/DL) tasks that are expected to take place on the IBM Power8 prototype machine running under RHEL7. The ML/DL software stack is made available on this platform mainly through the IBM PowerAI package which brings the following software packages: **Caffe**, **Chainer**, **DIGITS**, **Google Tensorflow**, **Theano** and **Torch**.

PowerAI has been designed to take advantage of the OpenPower supercomputers and is able to fully exploit the hardware available including the accelerators. However, PowerAI was initially available only on Ubuntu16, so the only way to run it was to use containers. In that way, the ML/DL workload executed on the allocated compute node is performed inside a Docker container.

To overcome the unavailability of the PowerAI package under RHEL deployed at IDRIS, a custom solution integrating LSF and docker was developed [23].

6.5.1.2. Socker: running Docker containers on Slurm

Status: Pilot

Socker [25] described in deliverable D6.3 [1], is a wrapper for running Docker containers securely inside HPC jobs. Socker uses the underlying Docker engine to manage and run containers. It is mainly designed to enable the users of our Slurm clusters (Abel and Colossus) to run Docker enabled jobs. There are mainly two functions provided by socker: First, run each container process as the user who submitted the job in order to make the container bounded by the user's capabilities. Second, bound the resource usage of any container, called inside a job, by the limits set by the queuing system for the job.

A performance benchmark for Socker has been already described in D6.3. During the security evaluation of the pilot experiment, the following attack surface is found:

- Socker is a wrapper which runs with SUID privilege. Bugs in the code might give root privileges to the user.
- Socker forces the container to run as the user, by enforcing the `--user` option of the `docker run` command. This option will not prevent a vulnerable image from attacking the system. A typical use case is a Docker image with a section in the sudoers file to grant sudo rights to a specific UID

On the other hand, Socker does its job well forcing the container to consume only the CPU and memory resources assigned by Slurm to the container job. To avoid the attack service, the

following changes has been made to Socker (and approved by the IT security group at the University of Oslo):

- Socker is no longer a SUID binary. Instead of running docker commands as root, Socker uses a system user that has only one privilege, that is to run the docker command.
- To avoid the danger of vulnerable images, Socker drops all Linux privileges in the docker run command, which makes the root user unable to use all root privileges.
- As an additional layer of protection, Socker enforces the use of user namespaces. In this way, the container root is no longer the host root, i.e. even if a user manages to become root inside the container s/he doesn't get any root privileges on the host

Implementing the above, running a container application with Socker on the Docker engine becomes as secure as running a native application with user privileges.

MPI support

Socker doesn't have implicit MPI support so far. It can be used with MPI by doing the following:

- Install MPI (and IB drivers if any) on the host
- Use a Docker container with MPI supporting application (and the IB driver installed if any)
- Run uDocker with MPI as:

```
mpiexec -np <N> socker run -e LD_LIBRARY_PATH=/usr/lib <app> <args>
```

Use cases

The following use cases has been operated with Socker on Abel [26] and Colossus [27] clusters:

- *Data analytics:* **mriqc**, **freesurfer**, **heudiconv**
- *Physics:* **GAMBIT**
- *Robotics:* **Gazebo**

The following genomic tools has been supported for the ELIXIR [28] and Tryggve [29] projects:

cite-seq-count, **BWA-MEM**, **GATK**, **htsec**, **subread**

6.5.1.3. uDocker

Status: Production

uDocker [30] is a basic user tool to execute simple Docker containers in user space without requiring root privileges. Enables basic download and execution of Docker containers by non-privileged users in Linux systems where Docker is not available. uDocker has its own container platform and supports multiple container runtimes. It is open source and written in Python.

MPI support

Like Socker, uDocker doesn't have implicit MPI support, and using uDocker with MPI is similar to Socker:

```
mpiexec -np <N> udocker run -e LD_LIBRARY_PATH=/usr/lib --hostenv --hostauth \
--workdir=<app-main> <app> <args>
```

Ref [31] includes an example for running uDocker with OpenMPI and openQCD.

Use cases

Since uDocker does not require any admin privileges to install, it has been offered to be used directly by the users in UiO (both Abel and Colossus) and CESGA FinisTerrae-II

6.5.1.4. Pros and Cons

LSF/Docker & Socker

Pros:

- Usability and Flexibility: Docker is the most popular and user-friendly container platform
- Portability and popularity: Almost all other container platforms have support for converting from Docker images, and the vast majority of containerized software packages are built with Docker
- Docker has the best support and richest set of features among all container platforms
- Docker with IBM LSF and Socker is “currently” secure
- Based on the docker engine, not a standalone runtime. Docker engine is maintained by Docker Inc. It also has support for multiple runtimes including NVIDIA, i.e. both prototypes can run NVIDIA GPU jobs.
- Security is continuously evolving and enhancing in Docker

Cons:

- The Docker engine is not designed for large scale distributed systems, e.g. Tier-0 and Tier-1. The problem of impossibility to share image and container repositories among multiple nodes remains.
- The Docker security documentation has to be carefully checked for each new docker release. The two illustrated pilots need to be revised against security before usage with new Docker releases, because they are both based on the Docker engine.

uDocker

Pros:

- Used for Docker containers, taking the advantage of the docker advanced support for image building.
- It can be installed and used without any admin privileges
- Supports multiple container runtimes
- Founded and maintained by INDIGO and is part of EOSC-hub. A collaboration has already started which means that PRACE could get advanced formal support

Cons:

- No implicit support for MPI
- Not as powerful as the Docker engine in terms of features

6.5.2. Private Cloud On a Compute Cluster PCOCC

Status: Production

PCOCC [32] (pronounced like "peacock": Private Cloud on a Compute Cluster) allows users of HPC cluster to host their own clusters of VMs on compute nodes alongside regular jobs. Users are thus able to fully customize their software environments for development, testing or facilitating application deployment. Compute nodes remain managed by the batch scheduler as usual, since the clusters of VMs are seen as regular jobs. From the point of view of the batch scheduler, each VM is a task for which it allocates the requested CPUs and memory and the resource usage is billed to the user, just as for any other job. For each virtual cluster, PCOCC instantiates private networks isolated from the host networks, creates temporary disk images from the selected templates (using Copy-on-Write) and instantiates the requested VMs. PCOCC is able to run virtual clusters of several thousands of VMs and has enabled varied new uses of CEA's compute clusters, from running complex software stacks packaged in an image to reproducing Lustre issues happening at large scale without impacting production servers. The networking model of PCOCC is described in Figure 8

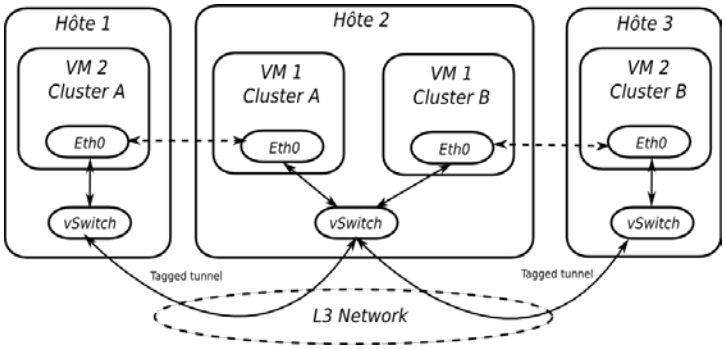


Figure 8: PCOCC Networking Model: VM networks with multiple physical hosts

PCOCC checkpointing support

Importation of images directly made by the users on his laptop is also possible: PCOCC support different types of image. Example “providing automatic checkpoint restart”:

PCOCC can also take a checkpoint restart of the entire virtual cluster:

```
pcocc ckpt [-j <jobid>] $CCSCRATCH/mycheckpoint
```

It is not as efficient as a real checkpoint restart directly integrated in the application but if nothing is done at an application level, this can be a solution: it is used in production on some genomics application at CEA. A full integration has been done using SLURM prolog/epilog with automatic checkpoint at the end of the job and automatic re-submission.

Work in progress

Containers are more and more used and are provided by ISV (Independent Software Vendor) to launch complex environment and products which is not the case for VMs. Because containers are

not really far away from VMs, PCOCC can provide a single interface to our users for launching either VMs or containers. Container support would be developed by mid-2019.

6.5.2.1. Use cases

PCOCC is deployed on CEA R&D cluster since 2014 and on our production computing centres since 2016.

It is really used by our users for:

- checkpoint restart job (for Gaussian application and Genomics applications)
- deep learning tool which needs the latest GPU driver or special system software stack
- remote visualization by creating VMs with 1 GPU, on a multi-GPU nodes (more node, secure solution, …)

More over PCOCC is used for internal (administrator) use for:

- **debugging at large scale:** PCOCC was used for debugging Lustre at large scale. It is more convenient to debug a VM rather than directly the kernel of bare-metal machine.
- **homemade system software non-regression tests**. For NFS-Ganesha opensource project, PCOCC was coupled to Gerrit/Jenkins product (automatic non-regression tests launching product), to instantiate client and servers to test it automatically.
- **Inter-shipment:** Because the virtual cluster instanced by PCOCC is secure (VMs run as user and confined network using VLAN or equivalent), inter-shipments can have administrator privileges in this virtual cluster. Moreover simulating large scale cluster is very convenient: 512 VMs take only 1024 cores.
- **Teaching:** All our lessons are prepared on VMs and than are launched directly on the university cluster.

At CEA, different use-cases ask for flexibility for the computing centre and especially for more services:

Genomics population: This population needs a lot of brand-new specific products with always the latest version. For a computing centre, it is very difficult to provide such up-to-date environment. One solution is to let users providing and running such environment in a container or in a virtual machine.

Industrial population: Industrial cases with certified codes which are validated on a specific OS (like Windows for example).

Admin population: Testing in advance new OS version before putting it in real production is very convenient for administrators. More flexible than dedicated hardware tests machine, virtual cluster is a very good solution.

Deep learning: Using specific software stack on specific hardware in a production environment: one example is to use Nvidia deep learning specific stack (Ubuntu based) on RedHat cluster.

6.5.2.2. Benchmarking Results

- **Performance:** Performances are very important for the users: to promote such a tool, performances are critical. So, several tests using VMs instead of bare-metal hardware have been done.
  - **First Infiniband tests: IMB tests.** The figure below shows negligible impact on the bandwidth. Some overhead have been seen on the latency: less than 500ns
  - **Parallel benchmarks:** In general, these benchmarks show less than 5% overhead using VM compare to bare-metal hardware. In some cases, VMs are faster than traditional compute nodes: thanks to huge pages. For an I/O point of view, benchmarks are very sensitive to the size of the I/O and the result depend really of how I/O are done.
  - **Other tests:** PCOCC is also very efficient to launch VM: more than 1024 VMs CentOS7 is less 1 minute has been reached (1 VM / core). Moreover, checkpoint of 4032 cores (1.7GB/core) linpack benchmark, has been written in less than 5 min for 6.8TB of checkpoint.

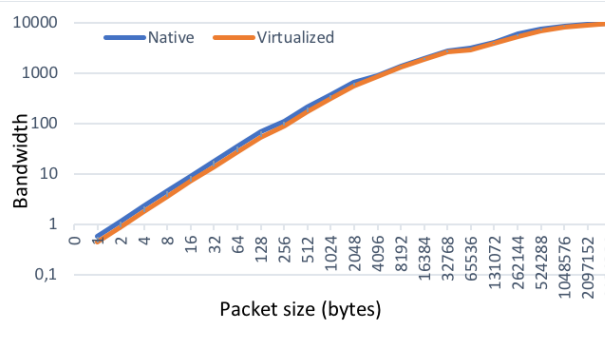


Figure 9: PCOCC Infiniband benchmark (packet size vs BW)

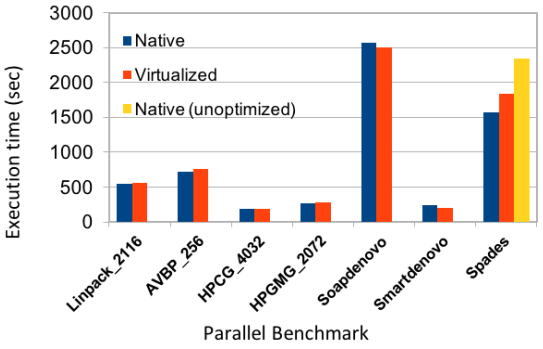


Figure 10: Relative execution time of parallel benchmarks executed in a cluster of PCOCC VMs compared to the same benchmarks launched on the host cluster

6.5.2.3. Pros and Cons

Pros:

- proving that using VMs can be very efficient even in HPC
- providing lot of flexibility for the computing centre gives it more attractivity: even on the HPC part
- providing (or will provide) a single interface for users launching VMs or containers
- providing through PCOCC features that today container technology is not able to provide (like real security)

Cons:

- need SRIOV (or equivalent) for being efficient

- At IDRIS computing centre where OPA is used, it was not possible to use PCOCC like it is use at CEA (on Infiniband) because OPA does not support SRIOV feature or equivalent.
- based on a complex HPC infrastructure (SLURM, …)
- PCOCC is a tool that allows to use HPC cluster like a cloud cluster. So it needs to have pre-installed HPC cluster to be installed on the top.
- container technology changes very quickly and improves itself. So more and more features providing by PCOCC in this domain (like security for example) is less and less needed.

6.5.3. Singularity

Singularity [19] is a container paradigm, designed for HPC platforms, that aims to reach three specific goals: reproducibility of results, mobility of compute and user freedom. As promised since the first official release in 2017, Singularity software have been shown to be able to bring containers and reproducibility to scientific computing [33] [34]. Singularity have been developed to meet HPC needs including 1) security, because the container is executed in user space and no escalations to root permissions are allowed inside it, and 2) parallelism, because both MPI, OpenMP and hybrid codes run in a Singularity container. Moreover, a Singularity container can be executed in a HPC cluster using a batch system scheduler, like PBS [35] or Slurm [36], as a common HPC software: any special packages or custom configurations have to be done. There are two main methods building Singularity containers: bootstrapping both from other Docker or Singularity containers already available in the official Docker HUB [37] and Singularity Hub [38], or bootstrapping from Ubuntu or CentOS official distributions. In the first case, few modifications have been done as add a directory for test purposes or install additional packages. In all cases, the simplicity of usage has to be noted, both in building and run a container from scratch and in use a pre-built one.

6.5.3.1. Singularity with GPUs (CINECA & UiO/SIGMA2)

Status: production

Within a Singularity container it is also possible to run NVIDIA GPU software only by adding a specific flag "--nv" in the execution command line. Any driver installation in required in the container, or library binding among the host and the container. The CUDA Toolkit has to be available in the container, with a version compatible with the driver version installed on the GPUs in the host. CINECA activity has been concerned in the analysis of the Singularity usage in two different HPC architectures: GALILEO at CINECA [39] and ABEL at UiO [26]. More in details, the GPU part of GALILEO has been considered, a pool of 15 nodes, each equipped with 2 x 8-cores Intel Haswell 2.40 Ghz + 2 nVidia K80 GPUs, 128 GB/node RAM, and an Infiniband with 4x QDR switches as Internal Network. Regarding to ABEL cluster, also in this case the GPU part has been used, a pool of 17 nodes, each equipped with dual Intel Xeon(R) CPU E5-2609 @ 2.40GHz, aka Sandy Bridge with 64 GiB memory nodes each with two NVIDIA Tesla K20 (Kepler architecture) cards. A special focus has been done on testing GPU Singularity container performance respect to bare metal.

Use cases and benchmark results at CINECA

Here we describe the use cases in CINECA, more results will be included in the forthcoming white paper.

The performed tests aimed at comparing the performance of a software executed in a singularity container and on the bare metal. The software chosen has been Tensorflow [40], the widely used machine learning framework.

On GALILEO, two different containers have been built, one with Tensorflow 1.10.1 and one with version 1.12.1. In the first case, the performance results obtained have been compared with the official Tensorflow performance results available on the web site [41]. In the second case, the results obtained have been compared with those obtained by a Tensorflow installed on the host from scratch. All the tests have been executed on a single node. In both cases, no performance losing appears by executing Tensorflow within a container. On ABEL, only the containerized version of Tensorflow 1.10.1 has been executed. Some tips for building the container have been also provided. On the container, it has been decided to use the Docker container available from the official Tensorflow Docker Hub [42].

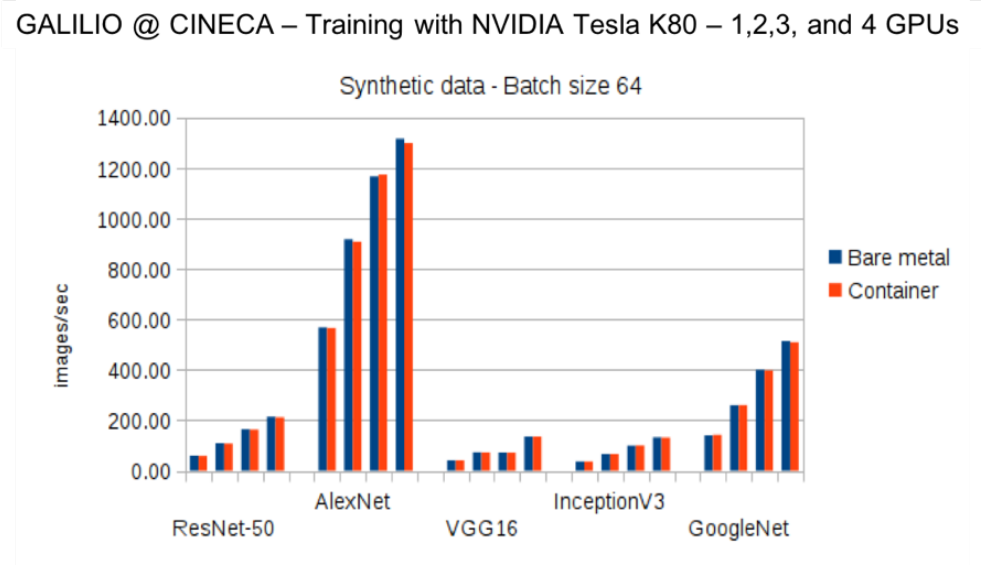
GALILEO

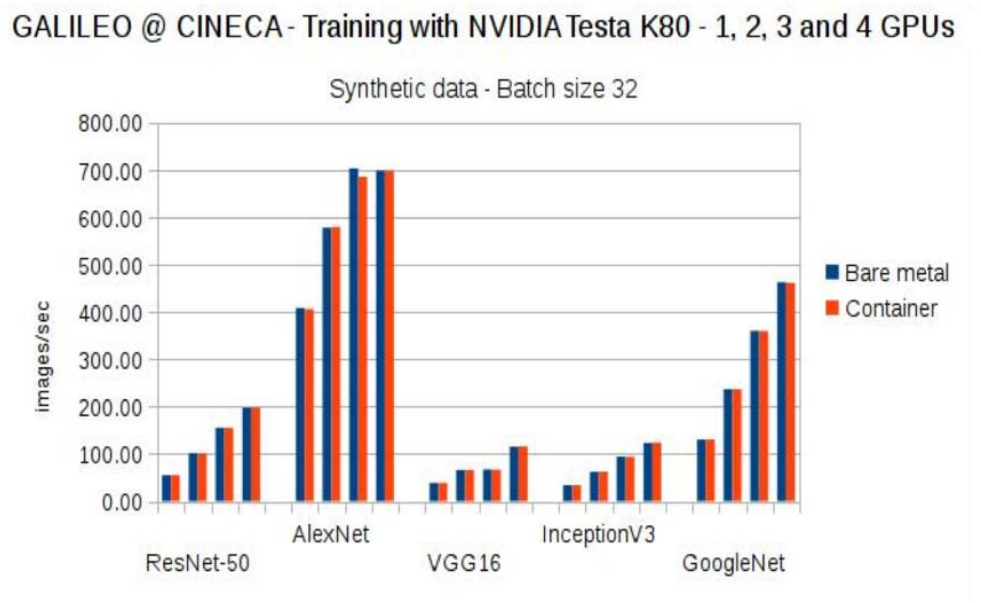
- **Case 1. Tensorflow version 1.10.1:** The Singularity version in the host was 2.5.2. The case considered is the training over NVIDIA Tesla K80, synthetic data. The network is ResNet-50 with a batch size of 32, over the data set ImageNet. The number of images per second is reported in Table 4

	GALILEO nVidia Tesla k80	Official Tensorflow nVidia Tesla K80
1 gpu	54.968	52
2 gpu	107.916	99
4 gpu	194.15	195

Table 4: Singularity Tensorflow Test results (# images/second) on GALILIO using 1, 2, and 4 GPUs

- **Case 2. Tensorflow version 1.12.0:** The Singularity version in the host was 3.0.2. Both for bare metal and container tests, five different neural networks have been considered: AlexNet, GoogleNet, InceptionV3, ResNet-50 and VGG16. The dataset was ImageNet (syntetic), and three different batch sizes was analysed: 32, 64 and 128 per device, where the device is the GPU. All the run have been execute on a single node with 2*8-cores Intel Xeon E5-2630 v3 @ 2.40GHz and 2 NVIDIA K80 GPUs.
  The number of images per second are reported in Figure 11. Fixed the batch size, each picture shows the number of images per second computed in each model and for 1, 2, 3 and 4 K80 NVIDIA GPU on a single GALILEO node. Note that the Model VGG16 and Inception V3 with a batch size of 128 are not shown because the run was out of memory.


GALILIO @ CINECA – Training with NVIDIA Tesla K80 – 1,2,3, and 4 GPUs
Synthetic data - Batch size 64
images/sec
1400.00
1200.00
1000.00
800.00
600.00
400.00
200.00
0.00
ResNet-50
AlexNet
VGG16
InceptionV3
GoogleNet
Bare metal
Container


GALILEO @ CINECA - Training with NVIDIA Testa K80 - 1, 2, 3 and 4 GPUs
Synthetic data - Batch size 32
images/sec
800.00
700.00
600.00
500.00
400.00
300.00
200.00
100.00
0.00
ResNet-50
AlexNet
VGG16
InceptionV3
GoogleNet
Bare metal
Container

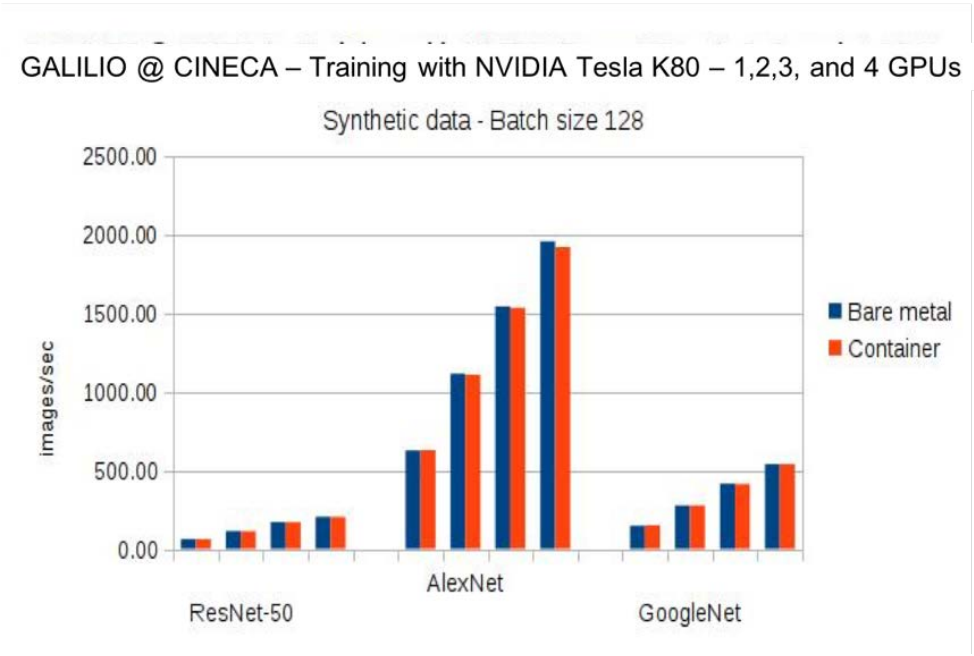


Figure 11: Tensorflow training (# images/second) with NVIDIA Tesla K80 using 1,2,3, and 4 GPUs

6.5.3.2. Singularity with MPI (CINECA & UiO/SIGMA2 & EPCC & CESGA)

Status: production

The Singularity team has created registries; SingularityHub, a public registry like DockerHub, and SRegistry. Both are tools used to remotely store and transfer Singularity images from the Cloud. While SingularityHub is hosted and maintained by Singularity team, SRegistry can be deployed and managed in our own cloud. In addition, the Singularity tool has included a new pull command for downloading or using remote images stored by means of these services. We can take advantage of this improvements to enrich application workflows in two ways; delivery automation and workflow portability. The need to be superuser to create Singularity containers is still present, and it will remain as an inherent requirement in the implemented containerization model. In multi user system as HPC, a normal user will usually not have superuser permissions. Taking all this into account, the Singularity usage workflow has been updated to be adapted to its new features.

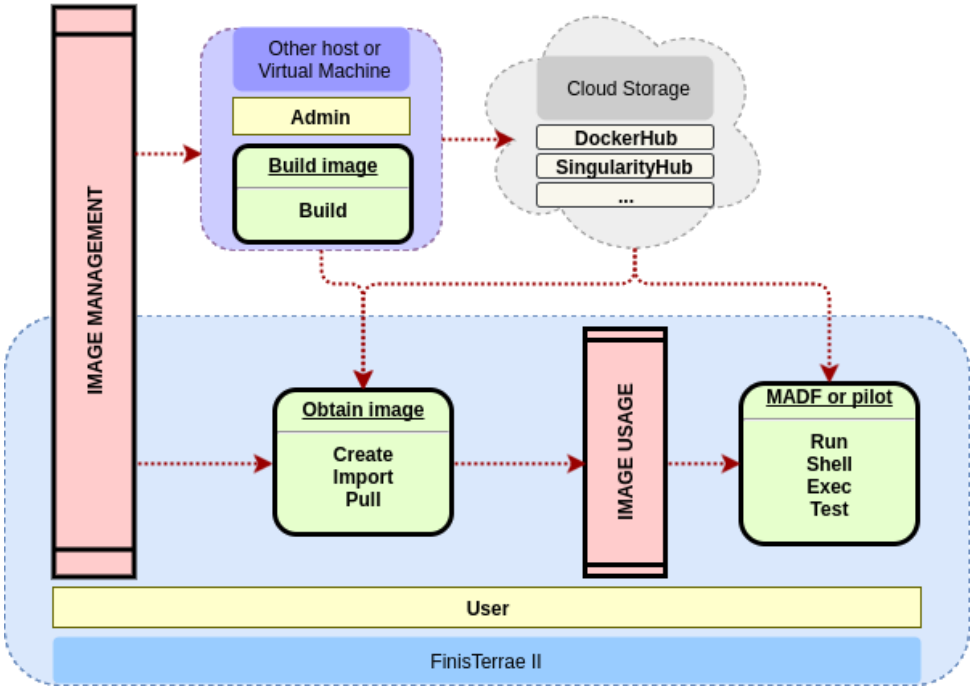


Figure 12: Workflow of Singularity images in the e-Infrastructure

Figure 12 describes the installation in FinisTerrae II, users can pull images or execute containers in FT2 from public registries, and also import images from tar pipes. Once the image is created, Singularity allows executing the container in interactive mode, and test or running any contained application using batch systems. All the work-flow can be managed by a normal user at FinisTerrae II, except the build process that needs to be called by a superuser. We can use a virtual machine with superuser privileges to modify or adapt an image to the infrastructure using the Singularity build command. At EPCC the tests included the ARCHER benchmark suite (https://github.com/hpc-uk/archer-benchmarks)

Use cases and benchmarking results at FinisTerrae II

Here we describe the use cases at FT2, more results will be included in the white paper. The benchmarks were performed in order to demonstrate that Singularity is able to take advantage of the HPC resources, in particular Infiniband networks and RAM memory. For these benchmarks we used a base Singularity image with an Ubuntu 16.04 (Xenial) OS and several OpenMPI versions. For these benchmarks we took into account the MPI cross-version compatibility issue exposed in the previous section.

The STREAM benchmark is de facto industry standards for measuring sustained RAM memory bandwidth and the corresponding computation rate for simple vector kernels. The MPI version of STREAM is able to measure the employed RAM under a multi node environment. The fact of using several nodes with exactly the same configuration helps us to check results consistency. In this case, two FinisTerrae II nodes, 48 cores, were utilized for running 10 repetitions of this benchmark natively and within a Singularity container with a global array size of 7.6×10^8, which is a big enough size to not be cacheable.

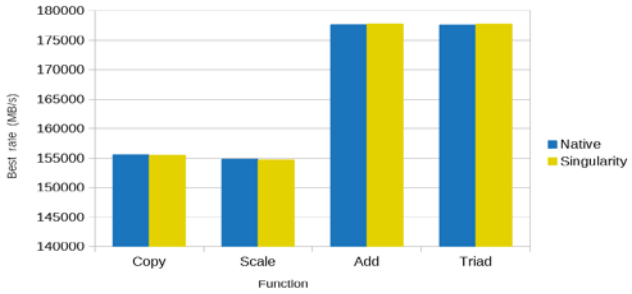


Figure 13: STREAM best bandwidth rates (MB/s) comparing Singularity with native run using different benchmarks

As we can see in the above figure, obtained bandwidth rates are really close between the native execution and the execution performed from a Singularity container, differences are negligible.

Infiniband networks also have decisive impact on parallel applications performance and we have also benchmarked it from Singularity containers. We used the base Singularity container with three different OpenMPI versions (1.10.2, 2.0.0 and 2.0.1) together with OSU micro-benchmarks. OSU is a suite of synthetic standard tests for Infiniband networks developed by MVAPICH. In particular, among the bunch of tests included we have performed those related with point-to-point communications in order to get results about typical properties like latency and bandwidth. Only two cores in different nodes were used for this benchmark.

Latency tests are carried out in a ping-pong fashion. Many iterations of message sending and receiving cycles were performed modifying the size of the interchanged messages (window size) and the OpenMPI version used.

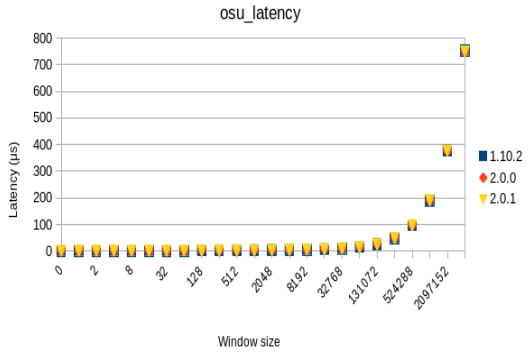


Figure 14: Latency from Singularity using OSU micro-benchmark

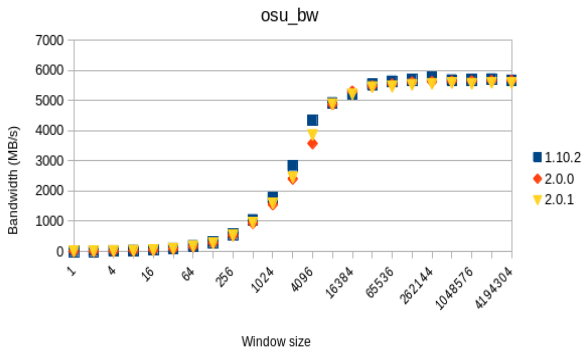


Figure 15: Bandwidth from Singularity using OSU micro-benchmarks

We can see in the above figure, unidirectional latency measurements are strongly related to the message size. For window sizes up to 8192 bytes we obtain less than 6 microseconds of latency, which are correct values for Infiniband networks. In this case the OpenMPI version does not have influence on the results.

For the measurement of the bandwidth, we increase the windows size to saturate the network interfaces in order to obtain the best sustained bandwidth rates. In the figure below, we can observe that the general behaviour is as expected. The maximum bandwidth reached is close to 6GB/s,

which are again in a correct value ranges for Infiniband. Although getting slightly different values depending on the OpenMPI version, we obtain similar results with critical values.

From these benchmark results, we can conclude that Singularity containers running parallel applications are taking advantage of these HPC resources under the specified conditions.

6.5.3.3. Pros and Cons

Pros:

- Singularity has implicit support for Open MPI, and automatically mounts the GPU drivers using the `--nv` , i.e. no need to install NVIDIA drivers inside the container
- Support for Docker to Singularity container conversion
- Runs as a binary not as engine. Runs by default as unprivileged

Cons:

- Singularity requires SUID binaries for full features. This introduces a security attach surface. It has been announced multiple times by the development team that the old releases must be removed due to newly discovered security vulnerabilities
- Still it is required for MPI applications that the host MPI and the container MPI are exactly the same[1]

6.5.3.4. Conclusions and future directions

At the moment, it is available in version 3.0.2 in MARCONI A3 cluster, Skylake partitions, in GALILEO both Broadwell and Haswell with GPU partitions, and in D.A.V.I.D.E cluster. The User documentation for CINECA is available at [43], for UiO at [44].

We plan to provide some Singularity containers for CINECA users, with deep learning, physics, chemistry frameworks. For licensed software in a container, a usage policy isn't designed and provided yet.

6.5.4. GYOC2: Get Your Own Containerised Cluster (UiO/Sigma2)

Status: Pilot

Galaxy web portal [2](depicted as a front-end in Figure 16) for wrapping different tools and enabling users to access those tools via a friendly web interface. Galaxy supports job submission to different HPC platforms including Slurm, PBS, and HTCondor, in addition to running container jobs on those clusters. Those container jobs can pull either Docker containers from Docker hub, or singularity containers from Galaxy singularity repository. In this pilot we implemented a fully containerized HTCondor cluster with Galaxy portal front-end. All compute nodes and the

[1] This is valid up to version 3 of OpenMPI

[2] Enis Afgan, et al. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update, Nucleic Acids Research, Volume 46, Issue W1, 2 July 2018, Pages W537–W544, doi:10.1093/nar/gky379

HTCondor manager run as containers on the top of a Docker swarm cluster. This solution is made mainly for cloud-based infrastructures. More about GYOC2 will be described in a forthcoming white paper.

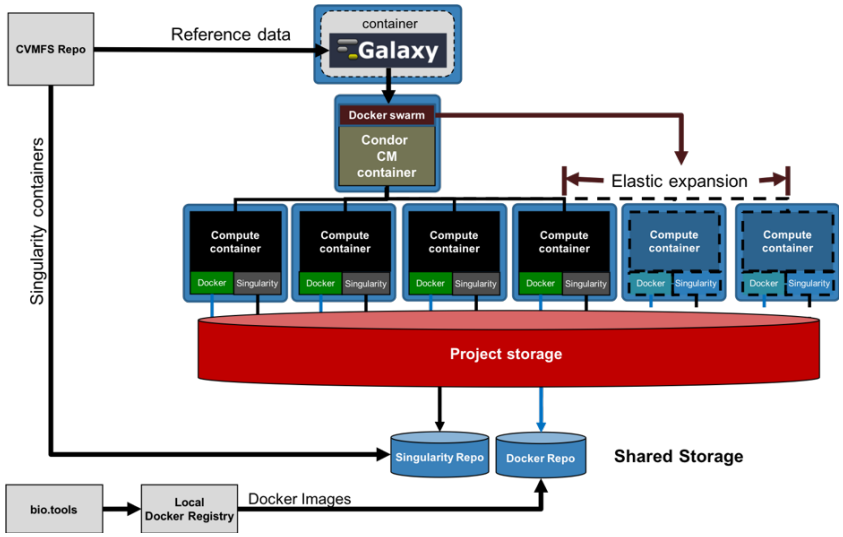


Figure 16: Get Your Own Container Cluster (GYOC2) Architecture: Compute nodes, resource broker, and the front-end portal are all in Docker containers

6.5.4.1. Pros and Cons

Pros:

- Flexibility: The number of compute nodes can be increased/decreased with a simple `docker service` command
- Portability: The whole cluster is packaged as Docker images. All what it needs to run is a collection of nodes/VMs with Docker installed

Cons:

- Some compute containers need to run in privileged mode
- Trouble shooting is within the container filesystem

6.6. Overall Conclusions and future directions

6.6.1. Conclusions

Socker: The solution is based on the docker engine which doesn't have implicit support for MPI. Advanced support for MPI to be added

LSF/Docker: The solution which is currently deployed on a prototype machine, has proved to satisfy fully the IDRIS security requirements. However, for now, we don't know about the future of the machine prototype and its related LSF/Docker solution. Indeed, the IDRIS machines are going to be renewed, the LSF batch scheduler is going to be abandoned in favor of SLURM. A migration of this platform to a SLURM/Docker platform can be of course considered but other technical solutions are also under investigation.

uDocker: The solution lacks maturity for MPI applications. Collaboration to be initiated with the development team in a PRACE EOSC context

PCOCC: PCOCC is used in production at CEA by different communities, and is now a real production-ready tool. Some parts of this tool need more developments like container support, but container technology evolves very fast pushing by Singularity, Redhat and other companies. So future works on PCOCC need first to take care of containers evolutions and try to find a correct positioning in this emergent ecosystem.

Singularity: Since 2017 Singularity software is in production on the main CINECA, UiO, EPCC, and CESGA HPC clusters. The following are instructions and good practices for building Singularity containers. These instructions provide the keys to build valid containers and avoid issues, making the containers completely transparent for users and also for the e-Infrastructure:

- Provide a valid Singularity container. Within the container, an entire distribution of Linux or a very lightweight tuned set of packages can be included, preserving the usual Linux directories hierarchy. It's also recommended to set up the required environment variables within the container in order to expose a consistent environment.
- To get transparent access to the host e-infrastructure storage from the containers, the shared filesystem root directories, e.g. /shared must exist within the container to be shared with the host.
- To run parallel applications using multiple nodes with MPI, the container must have installed support for MPI and PMI. Due to the Singularity hybrid MPI approach, it's mandatory to use the same implementation and version of MPI installed at the host and inside the container to run MPI parallel applications. Also for taking advantage of some HPC resources like Infiniband networks or GPUs, the container must support them. This means that the container must have installed the proper libraries to communicate with the hardware and also to perform inter-process communications.
- In the case of Infiniband support, there is not any known restrictions about the infiniband libraries installed inside the container. In the particular case of using GPUs from a Singularity container, the contained NVidia driver must exactly match the driver installed at the host. Singularity provides GPU containers portability through the experimental NVidia support option to allow containers to automatically use the host drivers.

6.6.2. Future directions

In PRACE-6IP, it is recommended that the service is moved to WP 6.1.

How to proceed: The service to follow the standard PRACE procedures to move to production. All PRACE sites which are supporting containers in production, e.g. CSCS, to join the service. PRACE sites that are planning to support containers, to get experience and support from the service. More container platforms should be also evaluated, e.g. Charlicloud, shifter, and runC.

6.6.2.1 Service Policy

A policy is needed with the following as the main bullet points:

- For bare-metal Tier-0 and Tier-1 clusters, Singularity and uDocker should be used;
- For security, Singularity should be deployed only to cluster with root-squash enabled on the shared filesystem;
- The best practices for GPU and MPI support, described above, should be followed;
- Socker should be used in sites that have the Docker engine features required. LSF/Docker to be used for LSF clusters;
- Galaxy should be optional to deploy as a front-end for container jobs;
- Container CVMFS repository for singularity containers to be deployed as part of the PRACE VPN;
- PCOCC and HTCondor are the PRACE supported platforms for VM workloads;

6.6.2.2 Draft KPIs

- **Deployment simplicity:** The ratio of container use cases that needed major modification to the original container file-system (from Docker hub) to all supported container use cases (should be minimum.);
- **Portability:** Containers that are built/used by one site, and needs modification to run on other sites to all PRACE containers (should be minimum);
- **Performance:** There should be no noticeable average performance degradation for containerized workloads compared to native workloads for similar applications;
- **Stability:** Failure incidents of container engines/platforms (should be minimum);
- **Reproducibility:** Results produced by using one or more containerized tool should be reproducible using the same container(s)
- **Security:** Security related incidents (e.g. users manage to get root privileges) should be minimum.

7 Service 6: Evaluation of new prototypes for Data Analytics services

7.1. Description of the service

The goal of Service 6 in Task 6.2, was to define which frameworks, libraries, tools and advanced features could be deployed to support, facilitate and promote data analytics activities in PRACE. Among the technical domains covered by the “Data analytics” term, we decided to focus on the current trends that show a growing interest in the community of data scientists in machine learning and deep learning technics that use automated algorithms, as they can offer faster data set analysis than more conventional methods. Thus, we evaluated how these technics can benefit from HPC or cluster environments that offer powerful CPUs and GPUs to manage the models’ complexity and accelerate the training, as well as to manage the increased amount of training data.
To achieve these objectives, we focused on the following pilots to develop the prototypes and think about a global data analytics service for the purpose of a future deployment and production service:

1. **Deep Learning SDK**
   The goal of this pilot was to identify and evaluate a set of deep learning frameworks and libraries that could be deployed over the PRACE infrastructure, which is composed of systems running different architectures.

We initially browse through a large promising set of frameworks and libraries whose popularity has changed in a different way since the beginning of PRACE-5IP. We finally consider a reduced set of them, composed of the most popular ones. Most of these components were already available at each partner site, but in order to offer a data analytics service, it was important to test the feasibility, efficiency, reliability, ease of use and scalability of these tools by analysing their behaviours from small to large systems.
To reach this goal, we based our evaluation on two main objectives:

- the capability to run standard benchmarks as a preliminary validation phase
- the identification of real use cases that we got through the launch of a call for prototypes, in order to complete the preliminary phase

2. Spark tools

The goal of this pilot was to investigate solutions for improving the performance of Spark in HPC environments. These solutions can be interesting in the scope of running HTC applications in HPC environments. We focused on an I/O solution that allows Spark clusters that frequently use the Hadoop Distributed File System (HDFS) to process data, to interface GPFS. Note that unlike the deep learning frameworks whose popularity has changed a lot during PRACE-5IP, Spark remains the only machine learning framework with still no major competitor for now.

3. Advanced features

The main goal of this action was to identify any additional services that could provide advanced features to users with the goal to enhance their productivity. Following the experiences we got from running the prototypes, we identified two advanced features: a dataset download service that can make standard datasets easily available to users and the Data Analytics GitLab project available within the PRACE GitLab to quickly gain expertise.

7.2. Description of the prototype services and use cases

7.2.1. Deep Learning SDK, libraries and use cases

7.2.1.1. Deep Learning SDK and libraries

The following table shows the set of components that we have considered for our prototype services with their main features:

Component	Main Features
Tensorflow [40]	- Open source DL/ML framework under Apache licence - Deep learning models expressed as data flow graphs - Pre-trained models available - Models integrated with the applications - Support for CPUs, GPUs, TPUs, mobile devices - GPU technology: CUDA, OpenCL (fork) - Distributed capability (multi-nodes) with gRPC (Verbs or MPI), Horovod - Pipelining capability for parallel trainings - A lot of tutorials and examples available - High level APIs: python, java, C, Go - Tensorboard data visualization toolkit

	- Tensorboard event logger - tfdbg TensorFlow Debugging - Very good performance - Frequent releases with new features
Caffe/Caffe2 [45]	- Open source DL framework under BSD licence (forks: IBM Caffe, Intel Caffe) - Model description in external files (i.e configuration without hard-coding) - Pre-trained models available - High level APIs: C++, python - Support for CPU, GPU, mobile devices - GPU technology: CUDA, OpenCL (fork), Intel (fork) - Support for multi GPUs and distributed (multi-node) capability through Intel Caffe, IBM Caffe and Caffe2 (not base Caffe) - Scalability enhanced in Caffe2
Keras [46]	- Open source deep learning library - Support for multiple backends (Tensorflow, CNTK and Theano) - High level API: python - Pre-trained models available - Support for CPUs, Nvidia GPUs and Google TPUs - Support for multi GPUs and distributed training with Horovod - Tutorials and examples available
Horovod [47]	- Open source distributed deep learning framework under Apache licence - Support for Tensorflow, Keras, PyTorch and MXNet - High level API: python - Support for multi GPUs and distributed training with MPI - Timeline profiling tool

Table 5: Frameworks and libraries - Main features

In the first phase, we used these frameworks and libraries to run standard benchmarks ranging from basic, to small, and then larger, varying the dataset size and the model size.

Table 6 and Table 7 show the main characteristics of the different benchmarks we run and the HPC resources we used:

	Model	**Dataset**	**Framework**
Basic benchmark [48] [49]	AlexNet [50] , GoogleLeNet [51] , Overfeat [52], VGG11 [53]	No (image created in memory)	Tensorflow - Caffe PowerAI
CIFAR-10 benchmark [54]	Simple CNN, 12 layers	CIFAR-10 163 MiB (60000*32*32 RGB pictures) - 10 classes	Tensorflow - PowerAI
ImageNet benchmark [55]	VGG-19 [56] , ResNet50 [57]	ImageNet-138GB (training) (1250000*400*350 RGB pictures) – 1000 classes	Tensorflow- Caffe – PowerAI

Table 6: Benchmarks description

Site	**Architecture/CPU**	**Accelerators**
CNRS/IDRIS – Ouessant	OpenPower 8 – 2 Power 8/node	4 GPUs/node – Nvidia Tesla P100
CINECA - Davide	OpenPower 8 – 2 Power 8/node	4 GPUs/node – Nvidia Tesla P100

CNRS/CC-IN2P3	Intel – 2 Xeon E5-2640v3/node	4 GPUs/node – Nvidia K80
EPCC – Urika-GX	Intel – 2 Xeon E5-2695	-
EPCC - Cirrus	Intel – 2 Xeon E5-2695	-
CINECA – Marconi	Intel – Xeon Phi 7250	-
CINECA – Galileo	Intel – 2 Xeon E5-2630/node	2 GPUs/node – Nvidia K80

Table 7: HPC resources

7.2.1.2. Use cases

In order to extrapolate the execution of standard benchmarks to real use cases, we launched a "Call for prototypes" [1] in May 2018. We got two answers to this call from the scientific communities that are described hereafter. We decided to gather this information in a use cases master document [59] which gives an overview of each use case with its major characteristics.

7.2.1.2.1. The Astrophysics use case [60]

This benchmark comes from a deep learning challenge. It uses 20 000 images (128x128 pixels, 32 bits, 2 GB) made of two galaxies overlapping. These galaxies have been extracted from real images from the Hubble Space Telescope and combined manually to create the blends. The goal of the challenge is to train a model that can automatically detect the contiguous region where the light of the two galaxies overlap. The model predicts a probability for each pixel to belong to such region, and the probabilistic image is then thresholded to obtain an actual prediction. The dataset is divided into 12 000 images for the training, 4 000 for the validation (during training) and 4 000 images for the final evaluation of the model. It uses the UNet model [61]. The results are shown in in Figure 19.

7.2.1.2.2. The CERN use case

The goal of the CERN use case was to use a machine learning tool for data certification for the RPC system of the CMS experiment. CMS data quality monitoring and data certification ensure that only consistent data is used for physical analysis and help hardware experts to spot anomalies in detector operation. Currently, the certification is done by human experts and is extremely expensive in terms of human resources and required expertise. The purpose of the project is to aid data quality monitoring and data certification by means of a neural network tool. The plan is to train the network using the PRACE infrastructure. Then, the network will be used at CERN to classify the data.The study of this use case has not been finished and is still on-going at IDRIS. The IDRIS Data Analytics experts do not run this use case but provide support by offering users an appropriate environment. Once the project is over, we are expecting a feedback from the users describing how they have benefited of the PRACE infrastructure.

7.2.2. Spark tools – The IBM Spectrum Scale support to Hadoop HDFS

The goal of this study was to identify if in an HPC environment already running GPFS, the HDFS connector, known as "HDFS Transparency" can bring some benefits to users and how the I/O performances vary compared to the original GPFS. Indeed, it can offer a solution to users to facilitate the software deployment by handling HDFS files which is not usual in HPC environments. Given that HDFS is optimized for big data processing, we decided to conduct this

evaluation using Spark benchmarks. Also, as our study focuses on comparing the HDFS connector with GPFS, we were interested in finding Spark I/O benchmarks rather than Spark compute benchmarks. The challenge was to find some standard benchmarks similar to TestDFSIO but for Spark applications instead of Hadoop ones. Unfortunately, we didn't find any Spark version of DFSIO able to run Spark 2.2. So, finally, we selected 2 benchmarks to evaluate the I/O performance of Spark applications: TeraGen and TeraSort applications. TeraGen performs a parallel creation of a large dataset so is write-intensive. TeraSort performs a parallel merge and a key-based sort, so, it presents a mixed I/O workload to the file system with both read and write intensive phases.

7.2.3. Advanced features

7.2.3.1. The GitLab Data Analytics Projects Group

During our prototyping phases, we identified tools that can help users to gain expertise and to share experience and results obtained on the PRACE AI infrastructure. The PRACE AI infrastructure gathers the HPC resources we used as described in Table 7. We started to gather the material that we used in the Data Analytics working group for the evaluation of the DL/ML tools over the different HPC architectures including standard benchmarks and use cases. This material can be reused by researchers that often start their first AI evaluation steps in a similar way we have done, namely by running these benchmarks or use cases. So, we created a Data Analytics projects group at the PRACE GitLab service which contains three projects:

- Benchmarks
- Use-cases
- Datasets

7.2.3.2. The dataset download service

During the development of our prototype services, we identified the need to offer users an easy access to datasets that are commonly used in the AI domain or to any other specific datasets that can be defined as reference datasets for the PRACE AI infrastructure. The specifications of this service are described in [62]. Indeed, whereas small data files can be downloaded quickly from the internet, dataset downloads of large files can become tedious, lasting for several hours for a complete download. Then, the solution can be to share these datasets in a dedicated storage space somewhere in the PRACE infrastructure in order to benefit from the fast, reliable and secure PRACE VPN network and enhance the user productivity. Also, in order, to offer an efficient dataset download service, the best solution is to prevent users from deploying software tools by themselves, opening user account at the PRACE site(s) providing this service, as all these steps make the process much more complex.
The dataset download prototype has been set up at CINECA [63] with one TB of storage disk. It is implemented through iRODS (Integrated Rule-Oriented Data System) [64] which offers a high transfer protocol and is currently used by the B2SAFE EUDAT [65] service for the federation of data nodes. This service provides both datasets download and upload capabilities to users using an anonymous access. Further developments are in progress such as an http/https API that will allow users to download the files by clicking on a url defined in a html page or use commands such as curl or wget using the http/https protocols.

7.3. Experiences with the prototype services

7.3.1. Deep learning SDK and libraries

The results of all prototype services we described in the previous sections are described in details in a white paper [66]. We present hereafter a subset of these results.

Except for the Basic benchmarks below, the plots show the Tensorflow results. We did not plot the Caffe results as they are very poor compared to the Tensorflow ones. Indeed and although it depends on the benchmarks, the Caffe performance can appear to be between 4 and 5 times worse for 1 and 2 GPUs and between 10 and 12 times worse for 4 GPUs.

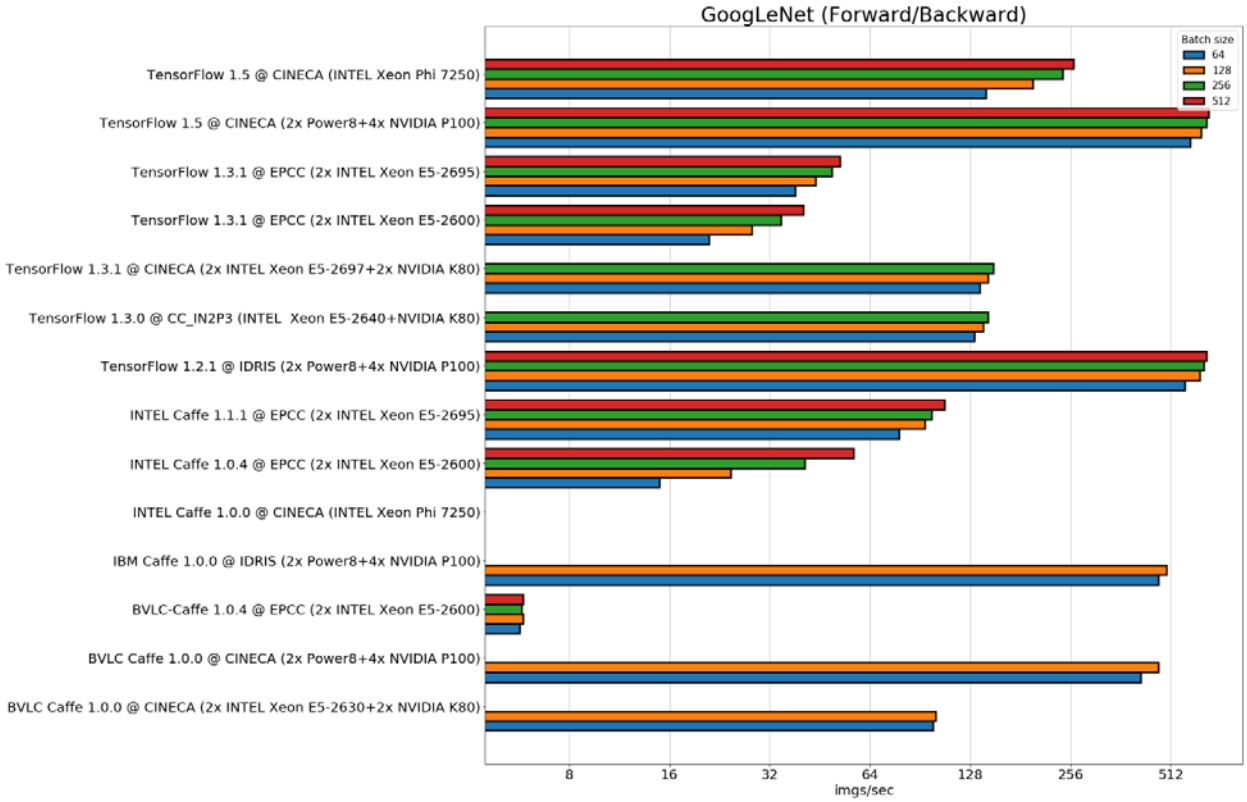


Figure 17: Basic benchmarks – Tensorflow and Caffe with the trained model GoogLeNet (X-axis: number of images per second [log scale], Y-axis: framework and architecture) and for a varying set of batch size with one GPU only

The results reported in Figure 17 above show that the performance levels depend mostly on the hardware configuration: the GPU NVIDIA P100 gets the best result with TensorFlow, followed by the performance obtained with Caffe with either the BVLC and IBM release. The INTEL Xeon Phi 7250 with TensorFlow follows, doing better than the GPU NVIDIA K80. In this case, TensorFlow outperforms Caffe.

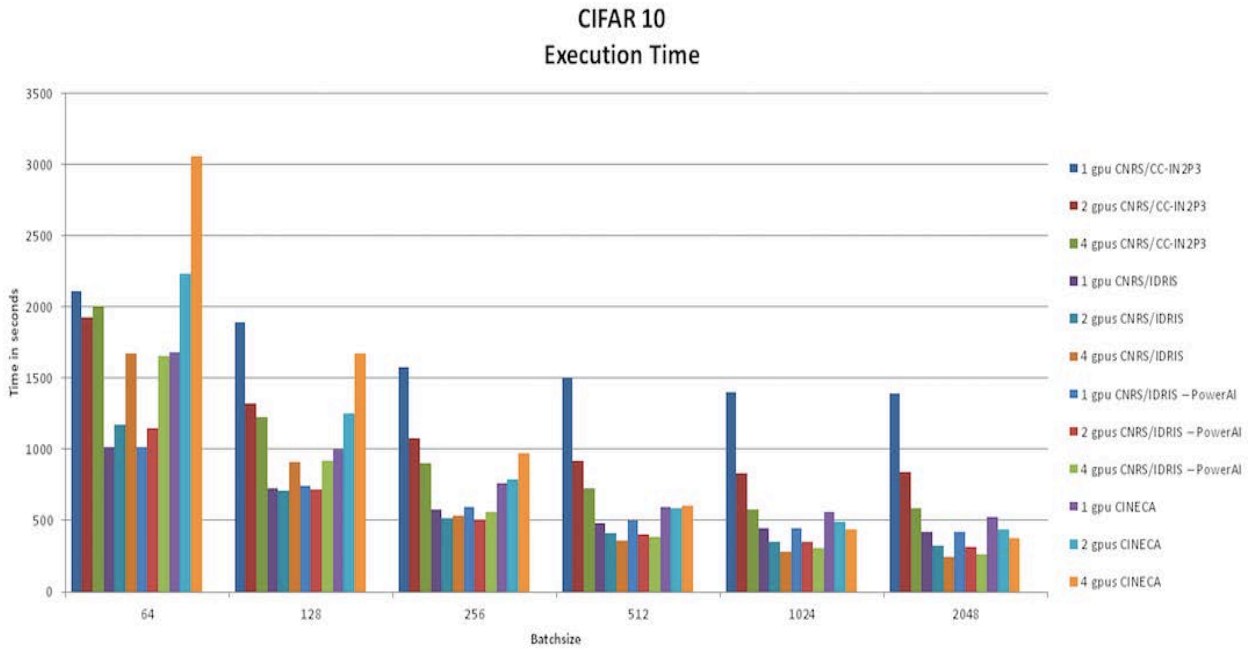


Figure 18: CIFAR-10 benchmark

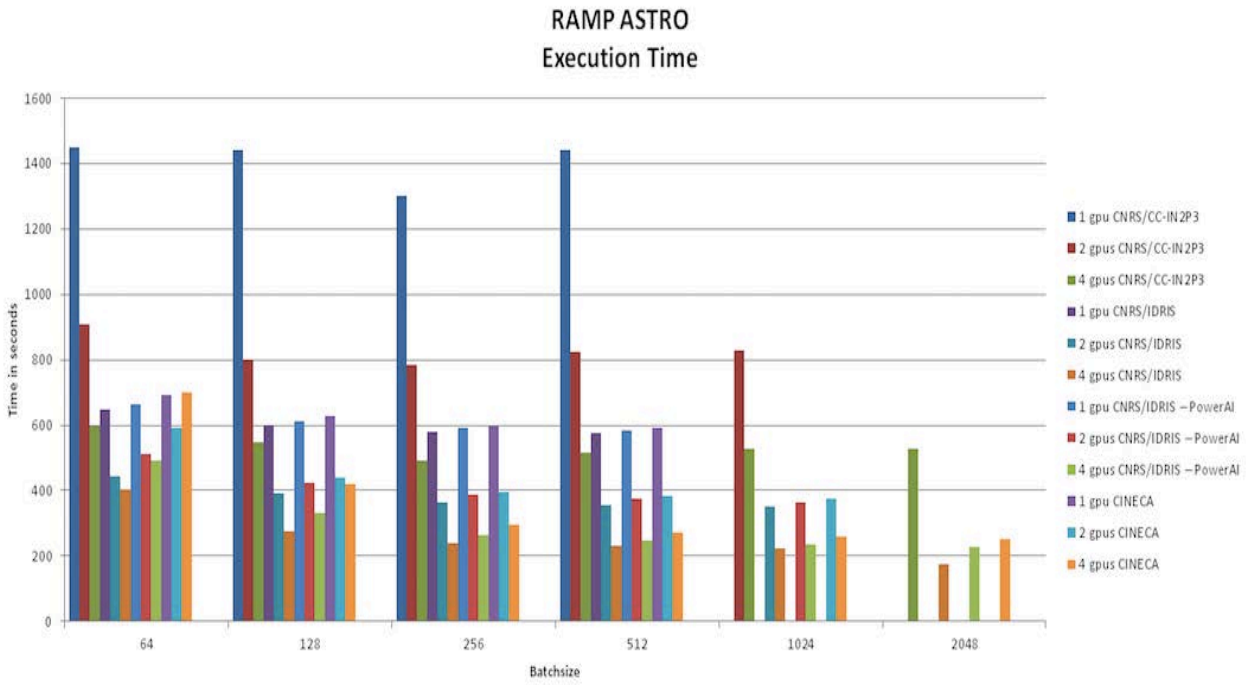


Figure 19: The Astrophysics use case

Figure 18 and Figure 19 show the performance of Tensorflow (X-axis: batch size, Y-axis: execution time (seconds)) and for different architectures for the CIFAR-10 benchmark and the Astrophysics use case. As for the basic benchmarks, the 2 figures show that the performances which depend mostly on the hardware i.e. the GPU NVIDIA P100 (IDRIS and CINECA) outperform the GPU NVIDIA K80 (CC-IN2P3). The results show also the huge impact of the batch

sizes with small batch sizes producing poor execution time performance compared to large batch sizes. Unlike the P100, the K80 shows a significant performance gain between one and two GPUs whereas the gain is less clear with four GPUs, that show a slight performance, but far from linear with the number of GPUs. The 2 prototypes at IDRIS with or without containers don't show a significant difference in term of performance.

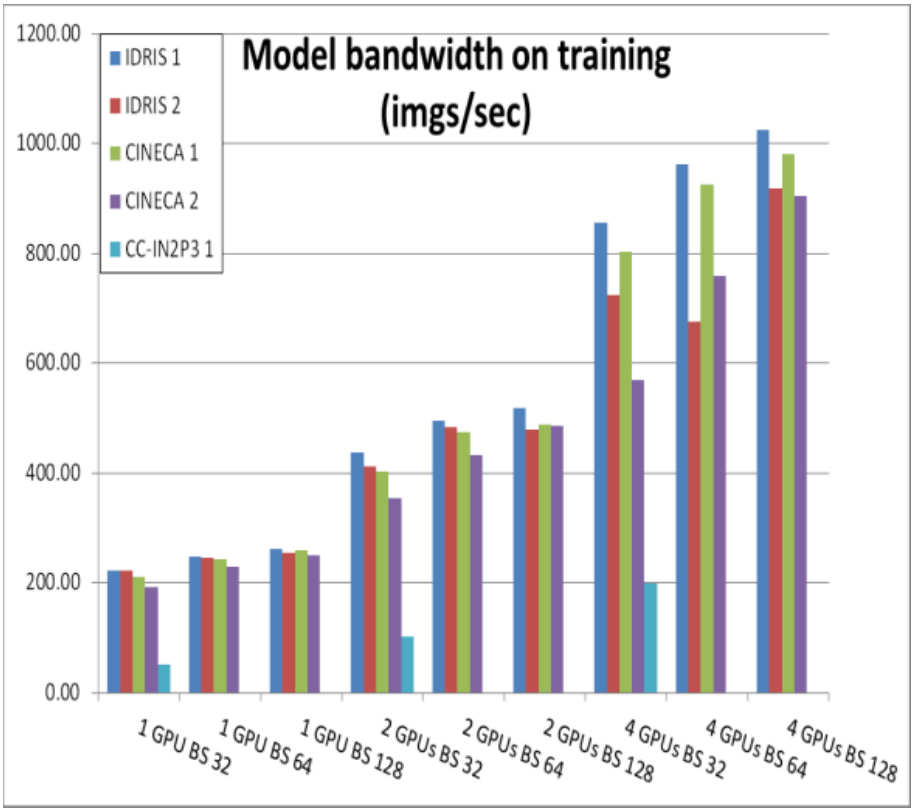


Figure 20: ImageNet - Intra-node model bandwidth

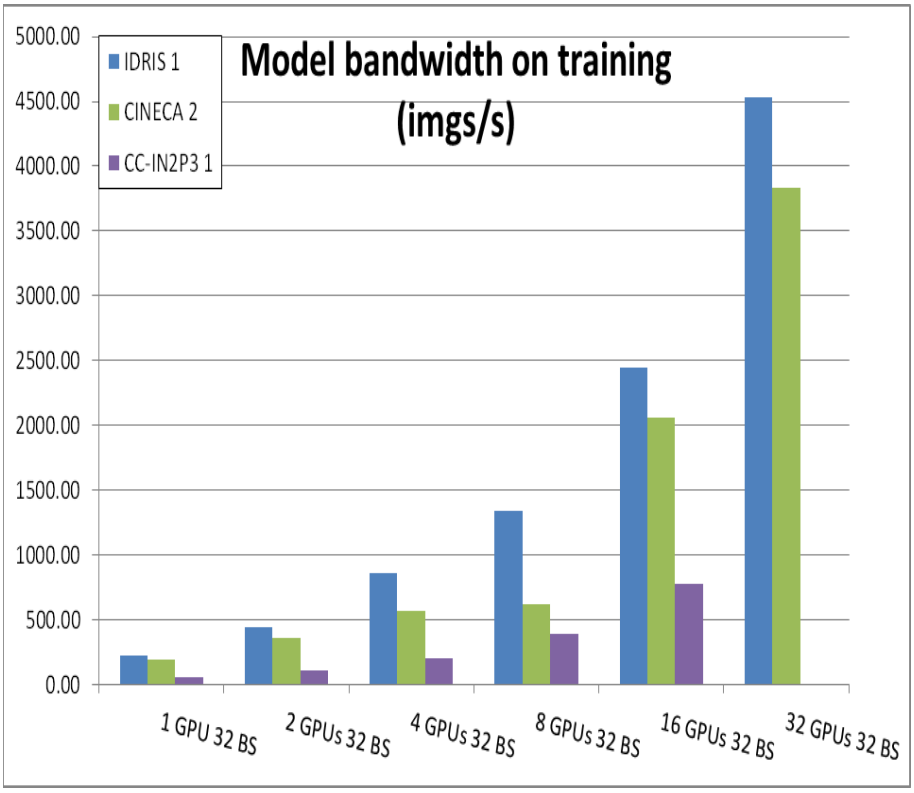


Figure 21: ImageNet-Multi-nodes model bandwidth

Figure 20 and Figure 21 show the performance of Tensorflow (X-axis: batch size and architectures, Y-axis: model bandwidth (images/second)) for the ImageNet benchmark.

The bandwidth increases as the number of GPUs and the batch sizes increases. The NVIDIA K80 runs Out Of Memory, the model becoming too large to manage for a batch size of 64 and 128. The use of container at IDRIS (PowerAI with docker) introduces an overhead in performance which tends to increase with the number of GPUs and larger batch sizes. For the NVIDIA P100, the plot shows a significant performance gain while increasing the number of GPUs from 1 to 32 but far from linear with the number of GPUs. The performance of the NVIDIA K80 is well above with a very limited gain from 1 to 16 GPUs.

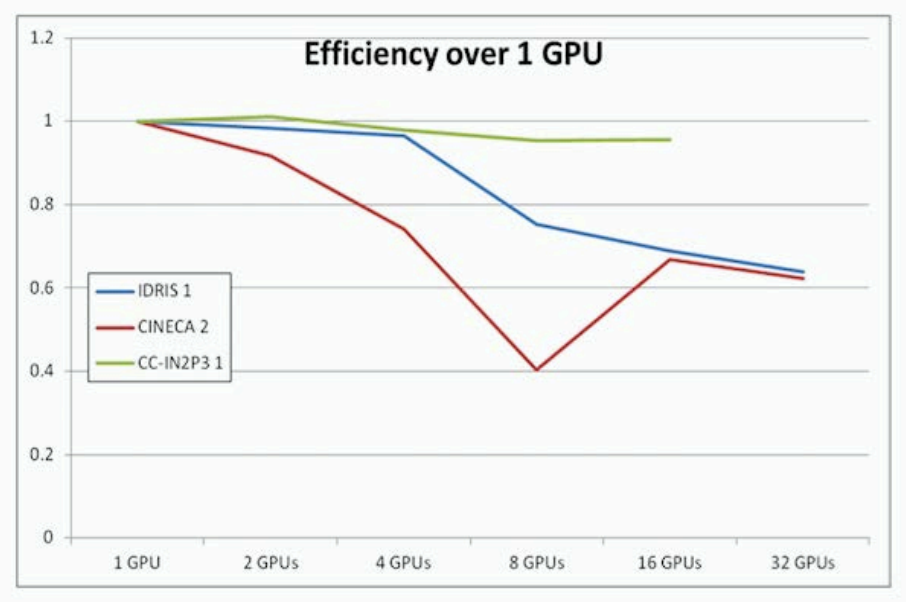


Figure 22: ImageNet - Parallel efficiency over GPUs

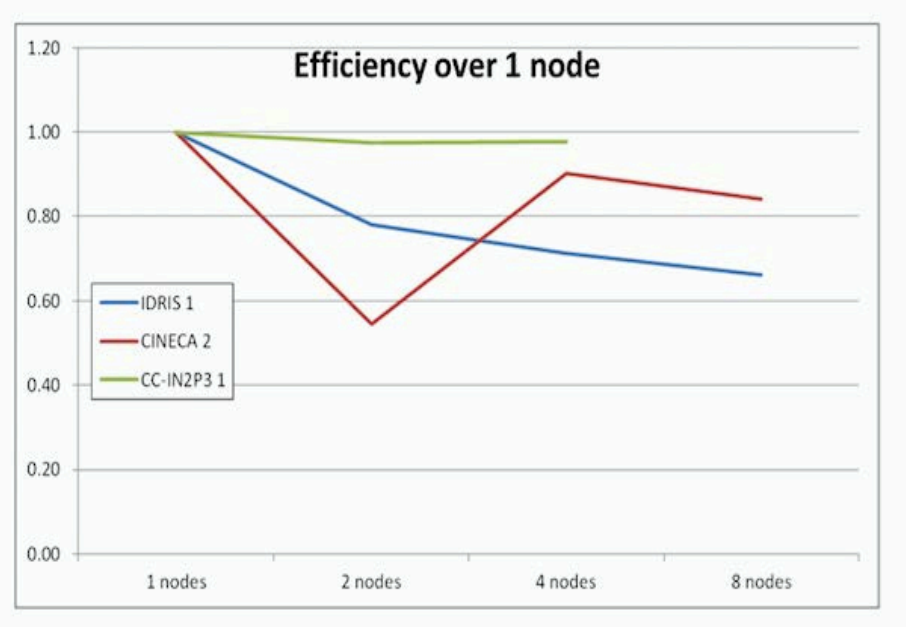


Figure 23: ImageNet - Parallel efficiency over nodes

Figure 22 and Figure 23 show the utilization of the resources over GPUs and nodes for the ImageNet benchmark. The plots show the very good efficiency of the NVIDIA K80 at CC-IN2P3 while increasing the number of GPUs and nodes. The results are worse with the NVIDIA P100 at IDRIS and at CINECA. Whereas IDRIS shows a good efficiency in an intra node environment, the efficiency at CINECA shows an important loss that can be related to memory transfers. In a multi-nodes environment, both IDRIS and CINECA infrastructures show an important loss that could be related to the network limitation.

The following tables show the PRO&CONS of the prototypes:

Tensorflow	Pro	Cons
Adoption	Major deep learning and machine learning framework, with a fast community growth. Very large community	
Flexibility	Low and high levels APIs Allow to create any arbitrary computational graph	Full support for python only
Software management	Frequent releases with new features. As this software is developed and maintained bu Google, continued support and development is ensured in the long-term. Future TF2.0 will have a compatibility module with TF1.x	Some releases can be very difficult to install Frequent changes in the APIs require code migration
Ease of use	Availability of additional layers such as Keras and Horovod	Steep learning phase Difficult to debug Distributed trainings hard to set up with the native distributed feature
Performance	The performances increase with each new release and increase the gap with Caffe/Caffe(2) (from 4 to 10 times depending on the number of GPUs)	
Documentation	A lot of tutorials, examples, codes and notebooks available	

Table 8: TensorFlow - Pro and cons

Caffe/Caffe(2)	Pro	Cons
Adoption	Rather for production environment because of its fast prototyping capabilities Caffe(2) merged with PyTorch to create the research and production platform	Caffe is in maintenance mode. Caffe(2) has a limited community support
Flexibility	Pre-trained models	
Software management	Caffe 2 is now merged into PyTorch	Caffe is not realeased anymore
Ease of use	Easy. Deep learning tasks can be performed without writing no line of codes	Difficult to debug
Performance	Scalable	Poor performance compared to Tensorflow
Documentation	Caffe2 tutorials and examples available	

Table 9: Caffe - Pro and cons

Keras	Pro	Cons
Adoption	Backed by Google engineering Will be fully integrated to Tensorflow 2.0	Support convolutional and recurrent networks only
Flexibility	High level API Layer based with a lot of built-in layer types Modular	Less flexible, less control than Tensorflow Doesn't allow a lot of model customization Not suitable for complex networks
Software management	Frequent releases	
Ease of use	Runs on top on Tensorflow but doesn't require to learn it	

	Intuitive interface, easy experimentation and learning curve Offer also data processing features	
Performance	Support for multi-GPUs	Add a slight overhead to Tensorflow performance
Documentation	Good documentation with examples	

Table 10: Keras - Pro and cons

Horovod	**Pro**	**Cons**
Adoption	High for distributed training, compared to the standard distributed Tensorflow	Tenforflow 2.0 will provide a better integration of Keras and new distributed features that can put Horovod outside of the race
Flexibility		
Software management	Frequent releases	
Ease of use	Easy to install, to implement (some changes in the source code are required) and to launch using mpirun	Running Tensorflow with Keras and Horovod on Power8 provides very poor performance and cannot be used
Performance	Usage of the standard MPI usually available on HPC machines The Horovod performances outperform the ones of the standard distributed Tensorflow	
Documentation	Good documentation with examples	

Table 11: Horovod - Pro and cons

7.3.2. Spark tools

Figure 24 and Figure 25 below show the results for the 2 benchmarks TeraGen and TeraSort for one or two nodes along with the time (in minutes) for two different HDFS data block sizes. HDFS and GPFS break files up into data blocks. The HDFS data block size is usually large compared to the GPFS one. We compared the results with the following values:

1) the HDFS block size equals to the GPFS block size: 512 KB
2) the HDFS block size equals to 32* 512KB (16 MB, maximum value for the GPFS block size)

HDFS means that the HDFS connector for GPFS is used whereas GPFS means a direct access to GPFS.

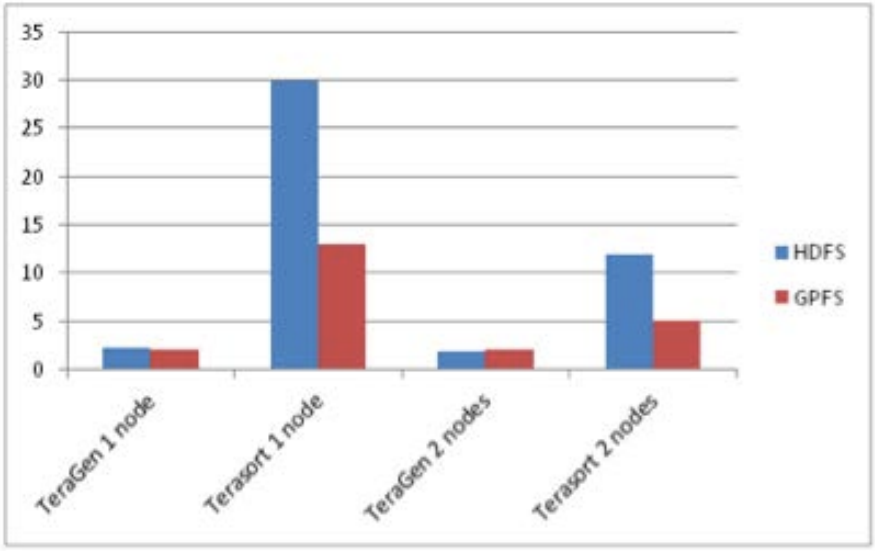


Figure 24: The X-axis describes the TeraGen and TeraSort benchmarks executed on 1 or 2 nodes. The blue colour indicates that the HDFS connector for GPFS is used, whereas the red colour refers to GPFS and it means that there is a direct access to the GPFS file system without any HDFS connector HDFS block size = 32*GPFS block size

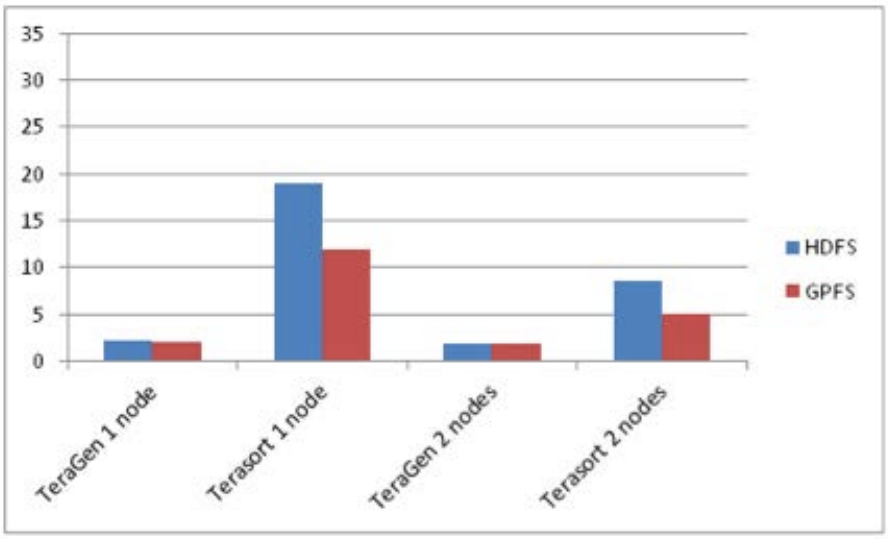


Figure 25: The X-axis describes the TeraGen and TeraSort benchmarks executed on 1 or 2 nodes. The blue colour indicates that the HDFS connector for GPFS is used, whereas the red colour refers to GPFS and it means that there is a direct access to the GPFS file system without any HDFS connector. HDFS block size = GPFS block size

We found that the best I/O performances were obtained using the maximum number of nodes available (2) and when getting the HDFS block size equal to the GPFS block size. This result makes it far from a standard distributed Hadoop cluster where the HDFS block size has to be large (typically 128 MB). Also, the results show that depending on the benchmarks used, the

performance cost introduced by the connector can be negligible such as in the TeraGen case, whereas with some others like Terasort, it shows a high performance cost. So, the HDFS connector for GPFS can facilitate the software deployment as it can avoid to recompile the source code by being able to handle HDFS url but it has an impact of the performance up to a factor 2.

7.3.3. Advanced features

7.3.3.1. The GitLab Data Analytics Projects Group

This service is available at https://repository.prace-ri.eu/git/Data-Analytics. The benchmarks project contains the available benchmarks and all the files needed to run the benchmarks. The following template shows how a benchmark is described using metadata:

Benchmark Name: name of the benchmark
Authors: provided by: name and email address
License: specifies the license type and usage restriction
Description: gives a short benchmark description
Type: scripts or notebooks
Language: can be python 2/3, java, R, Julia, …
Environment: example: python 3.6+, tensorflow, Keras, Horovod
Tag: defines tags that can be used to search for
Output: describes the benchmark output: example: data files, visualization, log files, models, …
Prace url: indicates where the source code can be found in the Data Analytics projects group

The use-case template is similar to the benchmark one and is described above. The datasets project gathers information on datasets and scripts that can be used to download them. These datasets can be used by any Data Analytics user projects. The following template shows how a dataset is described using metadata:

Dataset Name: name of the dataset
Description: gives a short description of the dataset and of its content
Size: indicates the dataset size (training, tests and validation sets)
File type: can be compressed archive, archive file, …
License: specifies the license type and usage restriction
Tag: defines tags that can be used to search for
url: specifies the site url
Dataset download url: specifies the dataset download url and the checksum
Prace dataset location: indicates where to find the dataset in the Prace infrastructure (in the case of large file, the datasets are not stored in the GitLab DA projects group)
Prace download script: indicates where a download script can be found in the DA projects group

7.4. Conclusion and recommendations for the next implementation phase

Within PRACE-5IP/WP6, Task 6.2, we have evaluated a set of pilot services that were likely to cover the needs of users interested in using Data Analytics technics. These pilot services were spread over the following areas:

- Deep learning frameworks and libraries
- Spark enhancements for the HPC environment
- Advanced features such as a Data Analytics GitLab and a dataset download service

Following our experiments, we can conclude and make the following recommendations for PRACE-6IP:
Regarding the deep learning frameworks, Tensorflow remains for now the major deep learning framework with a wild adoption in the AI field. Due to its popularity and the fact that it is backed by Google, it is seen by users as the tool in which it is worth investing time. It was already deployed at almost all of Data Analytics partner sites at the beginning of PRACE-5IP and it is the tool that all centers are willing to adopt. However, there are some key elements to consider before deploying it as a regular service in a production environment:

- Tensorflow is still evolving a lot and we are at a transition period between two major Tensorflow releases: 1.X and the new 2.0 that should be released very soon with a set of new features. Today, the major drawback of Tensorflow 1.X is that it is currently difficult to use. The Tensorflow developers are working on this problem as being the major goal of the next Tensorflow 2.0 release where Keras will be fully integrated with this release. This full integration should bring an ease of use with some Tensorflow specific enhancements. No doubt that user will quickly migrate to this major release given the migration tools that will be available. This can quickly leave Tensorflow 1.X obsolete.
- the way to deploy it at some sites is still evolving. Some sites are using containers, some other not or not only. So it has not reached a common agreement regarding the way to deploy it.

So, we recommend to continue working with this framework and to start the transition to a regular service in PRACE-6IP with Tensorflow 2.X.
The other frameworks that we have considered are Caffe/Caffe(2). They have currently a limited adoption and provide poor performance compared to Tensorflow. Also, in the meantime, Caffe(2) merged with PyTorch and the PyTorch started to grow. So, this software could become an alternative to Tensorflow in the future. Thus, our proposal here is to continue the pilot phase with PyTorch with a look at the Caffe(2) evolution.

Regarding Horovod, it is the most commonly tool used for distributed training with Tensorflow, as Tensorflow doesn't provide efficient and easy to use distributed features for now. However, there should be important changes with Tensorflow 2.0 regarding the distributed part that will have to be evaluated and compared to Horovod. So, we recommend to continue with this prototype in PRACE-6IP.

During PRACE-5IP, we had the opportunity to evaluate the popularity of Spark for the HPC community. We have to conclude that this tool remains for now, not very popular in this field and partners did not show a great interest in evaluating it or in evaluating any Spark performance enhancements for HPC. So, we recommend to stop this prototype for now, while keeping an eye on how HPC and HTC are going to get closer as this tool could resurface.

We recommend to continue with the Data Analytics GitLab project service as regular. This service relies on the PRACE GitLab that works properly, and information sharing can greatly benefit users. The regular status will require a periodic update of the information on the Data Analytics GitLab.
The Dataset download prototype is based on the operational B2SAFE EUDAT service. We recommend to continue with this prototype in order to benefit from the developments that are in progress regarding the http/https API which will allow users to download the files by clicking on a url defined in a html page or use commands such as curl or wget using the http/https protocols.

7.5. First draft of PKIs

In order to keep track of the usage of the data analytics service, here is a first proposal of KPIs that we should track of each service:

7.5.1. Deep Learning SDK

1. Number of projects per framework
2. Number of users or groups size per framework
3. Number of partners that have deployed the framework
4. Number of issues found
5. Service availability

7.5.2. Data Analytics GitLab

These PKIs are drifted from the PRACE GitLab service PKIs, plus the following ones:

1. Total number of access
2. Number of benchmarks
3. Number of benchmark downloads
4. Number of use cases
5. Number of use case downloads
6. Number of reference to datasets
7. Service availability

8 Conclusions

In this deliverable we have presented results of work done within Task 6.2 of Work Package 6 of PRACE-5IP project. We have considered six new services that were the PRACE's answer to raised needs in research or even wider community: (1) the provision of urgent computing services, (2) links to large-scale scientific instruments (i.e. satellites, laser facilities, sequencers, synchrotrons, etc.), (3) smart post processing tools including in-situ visualization, and (4) provisioning of repositories for European open source scientific libraries and applications, (5) Evaluation of lightweight virtualisation technology and (6) Evaluation of new prototypes for Data Analytics services.

The partners that have had assigned PMs in this task were grouped into six groups – one for each service. For service (4) it was decided at the very beginning of the project to migrate it into a regular service of WP6, For the other services, development of at least one pilot/prototype was planned to continue experimenting from PRACE-4IP (for the services 1, 2, 3) or to make first tests in real environment (new services 5 and 6).

Finally, these pilots provided a clear picture about opportunities and weaknesses related to each service and how to continue with it within the next implementation project, PRACE-6IP. Additionally, we have proposed also a first draft of KPI that could be used to monitor the services in the future. This deliverable will be a starting point to upgrade or continue work on these services within PRACE-6IP.